Mixed-Integer Nonlinear Programming Algorithms Based on Discrete Convex Analysis with Applications to Integrated Decision-Making Problems

by

David Alejandro Linan Romero

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Doctor of Philosophy

in

Chemical Engineering

Waterloo, Ontario, Canada, 2024

© David Alejandro Linan Romero 2024

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External examiner	Marianthi G. Ierapetritou Professor Chemical and Biomolecular Engineering, University of Delaware
Supervisor	Luis Ricardez-Sandoval Professor Chemical Engineering, University of Waterloo
Internal members	Hector Budman Professor Chemical Engineering, University of Waterloo Nasser Mohieddin Abukhdeir Associate Professor Chemical Engineering, University of Waterloo
Internal-external member	Fatma Gzara Professor Management Science and Engineering, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Chapters 1, 2, 4, and 6 to 9 were written by me. My supervisor, Luis Ricardez-Sandoval, provided editing and feedback. Some of these chapters were written for publication, as explained below:

- Part of chapter 2 has been incorporated in a review manuscript that has been submitted to a journal and is under review.
- The results in chapter 4 have been published in: Liñán DA, Ricardez-Sandoval LA. A Benders decomposition framework for the optimization of disjunctive superstructures with ordered discrete decisions. *AIChE Journal*. 2023;69(5):e18008. doi: https://doi.org/10.1002/aic.18008
- The results in chapter 6 have been published in: Liñán DA, Ricardez-Sandoval LA. Optimal design and dynamic transitions of multitask catalytic distillation columns: A Discrete-Steepest Descend Framework. *Chemical Engineering and Processing - Process Intensification*. 2022;180:108655. doi: https://doi.org/10.1016/j.cep.2021.108655. Appendix A supplements chapter 6 and has been published in: Liñán DA, Ricardez-Sandoval LA. A Discrete-Steepest Descent Framework for the Simultaneous Process and Control Design of Multigrade Reactive Distillation Columns. *IFAC-PapersOnLine*. 2022;55(7):370-375. doi: https://doi.org/10.1016/j.ifacol.2022.07.472
- The results in chapter 7 have been published in: Liñán DA, Ricardez-Sandoval LA. Discrete-Time Network Scheduling and Dynamic Optimization of Batch Processes with Variable Processing Times through Discrete-Steepest Descent Optimization. *Ind Eng Chem Res*. 2024;63(10):4478-4495. doi: https://doi.org/10.1021/acs.iecr.3c03455.
- The results in chapter 8 have been published in: Liñán DA, Ricardez-Sandoval LA. Multicut logic-based Benders decomposition for discrete-time scheduling and dynamic optimization of network batch plants. *AIChE Journal*. Published online 2024:e18491. doi: https://doi.org/10.1002/aic.18491.

I was the lead author of the remaining chapters of this thesis, meaning that I was responsible of the conceptualization, formulation of models, validation, investigation, visualization, initial draft writing, and implementation of the computer code including models and algorithms. My coauthors provided guidance during each step of the research and provided feedback on draft manuscripts. Publication details of these chapters are as follows:

- The results in chapter 3 have been published in: Liñán DA, Bernal DE, Gómez JM, Ricardez-Sandoval LA. Optimal synthesis and design of catalytic distillation columns: A rate-based modeling approach. *Chemical Engineering Science*. 2021;231:116294. doi: https://doi.org/10.1016/j.ces.2020.116294
- The results in chapter 5 have been published in: Liñán DA, Contreras-Zarazúa G, Sánchez-Ramírez E, Segovia-Hernández JG, Ricardez-Sandoval LA. A hybrid deterministic-stochastic algorithm for the optimal design of process flowsheets with ordered discrete decisions. *Computers & Chemical Engineering*. 2024;180:108501. doi: https://doi.org/10.1016/j.compchemeng.2023.108501
- Appendix B supplements chapter 8 and has been published in: Liñán DA, Reynoso-Donzelli S, Ricardez-Sandoval LA. Optimal scheduling and open-loop control of network batch processes under variable processing times using Generalized Benders Decomposition. *IEEE Xplore, American Control Conference (ACC)*. 2024, pp. 4466-4471, doi: https://doi.org/10.23919/ACC60939.2024.10644393

Abstract

This PhD thesis presents the development and application of new optimization models and algorithms for the solution of relevant Mixed Integer Nonlinear Programming (MINLP) and Generalized Disjunctive Programming (GDP) optimization problems arising in the fields of process design, simultaneous design and control, and simultaneous scheduling and control. The algorithms proposed throughout this thesis rely on a new optimization paradigm: incorporating concepts from discrete convex analysis (DCA) into the development of efficient MINLP and GDP optimization algorithms, rather than relying solely on conventional or general-purpose MINLP or GDP solvers. The main feature of DCA-based algorithms is that they may trigger a more efficient exploration of the integer and some binary or Boolean variables in the formulation known as ordered discrete decisions, leading to alternative local solutions that may not be attained with traditional local deterministic techniques. The advantages of these algorithms are demonstrated through the solution of a variety of optimization problems relevant in chemical engineering.

The first problem considered in this thesis is the rate-based optimal design of catalytic distillation columns. This thesis shows the application of a DCA-based algorithm to address this problem, known as the Discrete-Steepest Descent Algorithm (D-SDA). To the author's knowledge, this is an optimization problem that has not been attempted to be solved using deterministic MINLP optimization strategies in the past. This thesis also explores the application of the D-SDA as a logic-based solver (LD-SDA) to address GDP problems and introduces a new DCA-based technique called logic-based Discrete-Benders Decomposition (LD-BD). This method was designed following logic-based Benders Decomposition (LBBD) principles, which aim to extend the Benders Decomposition (BD) idea to a broader class of optimization problems, by giving the user freedom on the design of Benders cuts tailored for each application. To the author's knowledge, LD-BD is the first algorithm that combines DCA theory with LBBD principles. Another variant of the D-SDA proposed in this work is the Discrete-Steepest Descent Algorithm with Variable Bounding (DSDA-VB), which is used as the core method within a hybrid stochastic (i.e., metaheuristic)-deterministic algorithm for optimal design of process flowsheets. Unlike previous studies that propose hybrid deterministic and stochastic algorithms in sequential and nested arrangements, this thesis proposes a parallel configuration to perform the hybridization.

This thesis also brings novelty by extending the application of DCA-based optimization tools to the Mixed-Integer Dynamic Optimization (MIDO) domain. The first MIDO problem considered in this work is the optimal design and operation of CD units considering discrete and continuous design and operation variables combined with rigorous non-linear dynamic process models. The problem is solved with an enhanced modular D-SDA framework. The key novelty in this study is that optimal process design and dynamic transitions between different product grades in CD units are simultaneously optimized using a deterministic optimization framework. Another MIDO problem considered in this thesis arises in the field of simultaneous scheduling and control. This work proposes a general discrete-time simultaneous scheduling and dynamic optimization (SSDO) formulation based on the State Task Network (STN) representation. This formulation explicitly considers variable processing times, which is a key aspect in the integration of scheduling and control decisions. The resulting problem is solved using LD-SDA. This thesis also presents the first application of a LBBD technique in the field of SSDO applied to STN processes with a discrete-time scheduling formulation, through the herein proposed Multicut LD-BD (MLD-BD) algorithm. MLD-BD builds upon LD-BD and improves its performance by incorporating multiple cuts per iteration, a pruning strategy, and a cut-off technique. Different case studies show that the herein proposed algorithms explore the feasible region of ordered discrete decisions more efficiently than general purpose MINLP solvers, leading to more profitable solutions in shorter computational times. Overall, this thesis brings advances to the field of MINLP optimization by 1) introducing and demonstrating the advantages of new DCA-based algorithms when solving challenging problems in the fields of optimal design, and 2) extending these algorithms to the field of MIDO optimization, with optimal process integration applications.

Acknowledgements

I would like to express my gratitude to my supervisor, Luis A. Ricardez-Sandoval. Over the past few years, Luis has been instrumental in my professional development, providing exceptional guidance and feedback that have shaped me into a better researcher and engineer. I am deeply grateful to Luis for welcoming me into his research group.

I also wish to extend my gratitude to the professors, collaborators, and friends with whom I have had the opportunity to interact over these years, particularly those who have directly contributed to this thesis. Special thanks to the examining committee for their time and feedback through my PhD process.

I want to acknowledge that Chapters 4 and 7 of this thesis present original research that builds upon a methodology and open-source code developed through collaborative research conducted during my PhD: Bernal DE, Ovalle D, Liñán DA, Ricardez-Sandoval LA, Gómez JM, Grossmann IE. Process Superstructure Optimization through Discrete Steepest Descent Optimization: a GDP Analysis and Applications in Process Intensification. In: Yamashita Y, Kano M, eds. *Computer Aided Chemical Engineering*. Vol 49. 14 International Symposium on Process Systems Engineering. Elsevier; 2022:1279-1284. doi: https://doi.org/10.1016/B978-0-323-85159-6.50213-X; Ovalle D, Liñán DA, Lee A, Gómez JM, Ricardez-Sandoval LA, Grossmann IE, Bernal DE. Logic-Based Discrete-Steepest Descent: A Solution Method for Process Synthesis Generalized Disjunctive Programs. Preprint under review in the *Computers & Chemical Engineering* journal, 2024. http://arxiv.org/abs/2405.05358.

Dedication

I believe that those long-lasting experiences and memories that we share with our family are the engine that keeps us moving forward, makes us have a distinctive life position, and ultimately makes us who we are. Fulfilling the goal of finishing my PhD studies is thus a collective result that would not have been possible without such a supportive family. I am fortunate and deeply grateful for having Sonia ("mami"), Orlando ("papito"), Santiago ("Santi"), and my extended family guiding my life.

I am deeply thankful for having the constant love and support of my wife Diana Marcela Quitian Quintero and her family. I am sure this PhD, and life in general, has been much easier than it would have been without her company. Looking forward to continuing having endless adventures at your side.

Table of Contents

Examining Committee Membership ii

Author's Declaration iii

Statement of Contributions iv

Abstract vi

Acknowledgements viii

Dedication ix

List of Figures xv

List of Tables xxii

List of Abbreviations xxvi

List of Symbols xxix

Chapter 1 Introduction 1

1.1 Research Objectives 4

1.2 Contributions 5

1.3 Structure of the Thesis 7

Chapter 2 Background and Literature Review 9

2.1 Optimal process design and simultaneous design and control 10

2.2 Simultaneous Scheduling and Control 12

2.3 Deterministic MINLP optimization 15

2.3.1 General-Purpose MINLP Solvers 15

2.3.2 MINLP Tools to Aid Integrated Decision-making 18

2.4 Relevant Optimization Tools for this Thesis 20

2.4.1 Logic-based optimization of GDP problems 21

2.4.2 The Discrete Steepest Descent Algorithm 22

2.4.3 Discrete Convex Analysis (DCA) 23
2.4.4 Logic Based Benders Decomposition (LBBD) 25
2.4.5 Hybrid deterministic-stochastic MINLP optimization 26
2.5 Summary 28
Chapter 3 Optimal Synthesis and Design of Catalytic Distillation Columns: A Rate-Based Modeling Approach 30
3.1 Overview of PI and Optimal Design of CD Columns 31
3.2 CD nonequilibrium model 34
3.2.1 Microscale phenomena 36
3.2.2 Macroscale phenomena 40
3.3 Mathematical formulation 41
3.3.1 Sets and definitions 42
3.3.2 Structural decisions 43
3.3.3 Logical constraints 44
3.3.4 Catalytic distillation rate-based model 45
3.3.5 Superstructure example 51
3.4 Description of the D-SDA 52
3.5 Case study 55
3.5.1 Performance: Rate-based CD model 57
3.5.2 Rate-based optimization 59
Chapter 4 A Benders Decomposition Framework for the Optimization of Disjunctive Superstructures with Ordered Discrete Decisions 71
4.1 Problem statement 72
4.2 Mathematical framework 74

4.2.1 Background and problem decomposition 74
4.2.2 Generation of Benders cuts 76
4.2.3 Iterative refinement of Benders cuts 78
4.2.4 The Logic-Based Discrete-Benders Decomposition (LD-BD) algorithm 80
4.3 Computational experiments 84
4.3.1 Case study 1: Mathematical illustrative example 84
4.3.2 Case study 2: Distillation superstructure example 88
4.3.3 Case study 3: Reactor network example 94
Chapter 5 A Hybrid Deterministic-Stochastic Algorithm for the Optimal Design of Process Flowsheets with Ordered Discrete Decisions 96
5.1 Problem statement 97
5.2 Mathematical framework 99
5.2.1 Overview of stochastic methods and the DSDA 99
5.2.2 Proposed hybrid deterministic-stochastic algorithm 102
5.3 Computational experiments 109
5.3.1 Case study 1: A thermally coupled system. 112
5.3.2 Case study 2: a sequence of reactive, extractive and conventional distillation columns 118
Chapter 6 Optimal Design and Dynamic Transitions of Multitask Catalytic Distillation Columns 124
6.1 Problem statement 127
6.2 Methodology 129
6.2.1 Objective function 130
6.2.2 Constraints in the feasible region $\boldsymbol{S}$ 133
6.2.3 Problem partitioning and special features 137
6.2.4 Improved D-SDA 141

6.3 Case study 149
6.3.1 Objective function 152
6.3.2 Problem's constraints 154
6.4 Results and discussion 154
6.4.1 Sequential design methodologies: steady state design step 156
6.4.2 Simultaneous design methodology 159
Chapter 7 Discrete-Time Network Scheduling and Dynamic Optimization of Batch Processes with Variable Processing Times through Discrete-Steepest Descent Optimization 168
7.1 Problem statement 170
7.2 Mathematical framework 171
7.2.1 Discrete-time STN with variable processing times for SSDO 171
7.2.2 Mathematical formulation of the SSDO problem 175
7.2.3 Solution of the problem using the LD-SDA: reformulation and decomposition 178
7.2.4 Solution of the problem using the LD-SDA: Algorithm steps 180
7.3 Computational Experiments 183
7.3.1 Case study 1 185
7.3.2 Case study 2 192
7.3.3 Modelling and computational performance analysis 198
Chapter 8 Multicut logic-based Benders decomposition for discrete-time scheduling and dynamic optimization of network batch plants 201
8.1 Problem statement 202
8.1.1 The discrete-time STN model with variable processing times for SSDO 203
8.1.2 Problem reformulation 207
8.2 Algorithm framework 209

8.2.1 Master-subproblem decomposition 209
8.2.2 Generation of multiple benders cuts per iteration 211
8.2.3 Proposed strategies for the solution of subproblems 214
8.2.4 The MLD-BD algorithm for SSDO 219
8.3 Computational experiments 221
8.3.1 Case study 1 222
8.3.2 Case study 2 228
Chapter 9 Conclusions and Future Work 234
9.1 Recommendations for future work 237
Letters of Copyright Permissions 238
References 253
Appendix A A Discrete-Steepest Descent Framework for the Simultaneous Process and Control Design of Multigrade Reactive Distillation Columns 286
Appendix B Optimal scheduling and open-loop control of network batch processes under variable processing times using Generalized Benders Decomposition 301
Appendix C Supplementary Material for Chapter 3 316
Appendix D Supplementary Material for Chapter 4 322
Appendix E Supplementary Material for Chapter 5 339
Appendix F Supplementary Material for Chapter 6 371
Appendix G Supplementary Material for Chapter 7 390
Appendix H Supplementary Material for Chapter 8 401

List of Figures

Chapter 2

Figure 2.1. Illustrative design and control problem..... 11

Figure 2.2. Illustrative scheduling and control problem..... 13

Chapter 3

Figure 3.1. Nonequilibrium models for CD. 35

Figure 3.2. General representation of transport processes in CD. Thermodynamic equilibrium only occurs at the vapor-liquid interphase (♦), with compositions $x^{I}_{n,i}, y^{I}_{n,i}$ and temperature T^{I}_{n}. 36

Figure 3.3. Representation of transport processes for component i at stage n in the selected rate-based model for CD. 38

Figure 3.4. Schematic superstructure representation. $n = 1$ is the condenser and $n = NT$ is the reboiler. 42

Figure 3.5. Example of a design that can be obtained with the proposed MINLP superstructure. 52

Figure 3.6. Motivating D-SDA reformulation example: A) Conventional formulation, B) Reformulated strategy. 53

Figure 3.7. Rate-based simulation (A) and optimization (B) with of design presented in [244]. Symbol ⇋ R used for reactive stages. 58

Figure 3.8. Rate-based simulation (A) and optimization (B) with of design presented in [10]. Symbol ⇋ R used for reactive stages..... 59

Figure 3.9. Representation of a strategy that avoids structural redundancy. Symbol ⇋ R used for reactive stages..... 60

Figure 3.10. Optimization results: discrete decisions. A) Optimal equilibrium-based design [10], B) Optimal rate-based design. Symbol ⇋ R used for reactive stages. 65

Chapter 4

Figure 4.1. The Logic-Based Discrete-Benders Decomposition algorithm (LD-BD). 81

Figure 4.2. Iterations of the LD-BD algorithm for the mathematical example. 86

Figure 4.3. Iterations of the LD-BD and LD-SDA algorithms. The arrows indicate the sequence of iterations. 87

Figure 4.4. (A): Example of a column with 9 internal stages and the feed located at the fifth stage from bottom to top. (B): Example of a column with 8 internal stages and the feed located at the third stage from bottom to top. The remaining Boolean variables in (A) and (B) are False. 90

Figure 4.5. Representation of the discrete decisions of the distillation problem in the Boolean variables domain (A) and the external variables domain (B). 91

Figure 4.6. Feasible region of the distillation problem and enumeration of the initial guesses for external variables. The numbers in the grid are identifiers assigned to each point within the feasible region. . 93

Figure 4.7. Results of running LD-BD, LD-SDA and L-OA from different initializations of external variables. (A): objective function, (B): computational time. 94

Figure 4.8. Effect of increasing the number of randomly sampled points in D^{init} 95

Chapter 5

Figure 5.1. Hybrid arrangements for optimal flowsheet design aided with simulation software. A: sequential hybrid, B: nested hybrid, and C: parallel hybrid. DOF means degrees of freedom. 97

Figure 5.2. Proposed deterministic-stochastic algorithm. Black rectangles at the top and the bottom represent the beginning and the end of the parallelization, respectively. Dashed lines represent the exchange of information. 104

Figure 5.3. Hybrid algorithm implementation and software considered in this work. 111

Figure 5.4. Thermally coupled system that separates a mixture of isobutane, n-butane and n-pentane. 112

Figure 5.5. Convergence plots for the optimal design of the thermally coupled system..... 116

Figure 5.6. Illustrative convergence plots for the number of stages of C1 (A) and the reflux ratio of C1 (B). These plots correspond to the optimization run in Figure 5.5A. 117

Figure 5.7. Sequence of distillation processes designed for the production of MEK from 2,3-BD, and the separation of by-products. 119

Figure 5.8. Convergence plot for the MEK production process. 122

Chapter 6

Figure 6.1. Schematic example of the dynamic multi-scenario formulation. 137

Figure 6.2. Illustration of the pruning procedure 143

Figure 6.3. Proposed methodology for the solution of the simultaneous design and dynamic transitions problem. This results in the optimal transient solution $[v^{SSo}, v^{TRo}, \varphi^{o}, \varphi^{o}]$. 144

Figure 6.4. Illustrative representation of the enhanced D-SDA algorithm. A: Initialization (S1), definition of the initial neighborhood (S2) and steady state neighborhood verification (S3); B: pruning (S3), stopping criterion (S4) and selection of steepest descent direction (S5) C: line search (S6 and S7); and D: the next and last iteration starts at S2. 149

Figure 6.5. Schematic representation of the ETBE superstructure used for the case study. 150

Figure 6.6. A,B,C: optimal design of ETBE columns at different compositions required for the sequential overdesign approach (*Seq. 1*). The design in C is the result from *Seq. 2*. The design in D is the result from *Seq. 1*. 158

Figure 6.7. Results for the ETBE Case Study: Module 1 and Module 2. The initialization of Module 1 was taken from a previous work that considered the design of a ETBE column to produce ETBE at 83%. 159

Figure 6.8. Results of module 3, which implements the enhanced D-SDA. 160

Figure 6.9. Discrete configuration of the design obtained with the sequential (A) and the simultaneous (B) approaches. 162

Figure 6.10. Composition of ETBE product during the production schedule. 164

Figure 6.11. Inputs during dynamic transitions: simultaneous approach. Only the transition period of each scenario is plotted. 166

Chapter 7

Figure 7.1. The LD-SDA applied to the SSDO formulation in eqs 7.5A and 7.5B. 181

Figure 7.2. STN for case study 1. The execution of task T2 in U2 and U3 is described by a semi-batch reactor model. 185

Figure 7.3. Gantt charts illustrating the SIA (A), IIA (B) and LD-SDA solutions (C and D) (case study 1). Colored blocks represent tasks. Batch sizes $b_{i,j,t}$ in m^3 appear as numbers inside colored blocks. 189

Figure 7.4. Manipulated variables and temperature profiles for the second execution of task T2 in unit U3. Subplots A, C and E: SIA solution. Subplots B, D and F: LD-SDA(SIA) solution. 190

Figure 7.5. STN for case study 2. The execution of tasks R1, R2 and R3 in Rl and Rs is described by a semi-batch reactor model. 193

Figure 7.6. Gantt charts illustrating the SIA (A), IIA (B) and LD-SDA solutions (C) (case study 2). Colored blocks represent tasks. Batch sizes $b_{i,j,t}$ in m^3 appear as numbers inside colored blocks. 195

Figure 7.7. Concentration and manipulated variables profiles profiles when executing task R2 in unit Rs. A and C: SIA solution. B and D: LD-SDA solution. 196

Chapter 8

Figure 8.1. The multicut Logic-Based Discrete Benders (MLD-BD) decomposition algorithm. 219

Figure 8.2. A: STN for case study 1. B: STN for case study 2. 222

Figure 8.3. MLD-BD(p,h,co) (A) and SIA (B) Gantt charts (case study 1, $\eta = 100800$ s)..... 228

Figure 8.4. SIA (A), IIA (B) and MLD-BD (p,h) (C) Gantt charts (case study 2)..... 231

Appendix A

Figure A.1. The modular D-SDA approach 291

Figure A.2. The D-SDA. $f_{OBJ}^{SS} = +\infty$ for infeasible problems. 293

Figure A.3. The enhanced D-SDA. $f_{OBJ} = +\infty$ for infeasible problems. 294

Figure A.4. Optimal transitions and PID tuning parameters: proportional gain (K_C), integral time (τ_I), and derivative time (τ_D). (A): period 1, (B): period 2. 298

Figure A.5. (A) Optimal steady-state design (B) Optimal design using the modular D-SDA. $\rightleftharpoons$ R: reactive stages..... 299

Figure A.6. Optimal transition and tuning parameters for the Qr-x_{ETBE} PID loop. (A): x_{ETBE} profile, (B): Qr profile..... 299

Appendix B

Figure B.1. Proposed GBD strategy 311

Figure B.2. STN of the problem 312

Figure B.3. Proposed GBD approaches. A) GBD1, B) GBD2, C) GBD3 313

Figure B.4. A) Best scheduling solution; numbers within the colored bloks represent batch sizes (m^3). B) concentration profiles reaction R3 in Rl. C) Control profiles for reaction R3 in Rl..... 314

Appendix C

Figure C.1. Example of k-neighborhoods. 317

Figure C.2. The D-SDA framework. Black arrows represent the flow of information. 318

Figure C. 3. D-SDA's master problem: exploring external variables' feasible region and recalculating initializations. Grey and blue arrows: algorithm steps.init: initialization.iter: iteration. inner: line

search iterations. E is assumed to be bounded to guarantee convergence with a finite number of iterations. Also, $f_{FNLP} = +\infty$ for infeasible NLP problems. 320

Figure C.4. Time-efficient use of the D-SDA. 321

Appendix D

Figure D.1. RD superstructure. A) original formulation, B) reformulation with external variables. 322

Figure D.2. Steps to optimize a problem when initialized from an infeasible point. 331

Figure D.3. Algorithm to solve the optimization problems in initialization phases 1 and 2. 334

Figure D.4. Representation of 1) $\mathrm{co}\left(\mathrm{dom}_Z(f^*)\right)$ as a dotted blue line, and 2) $\bigcup_{y\in R^2} \mathrm{co}\left(\mathrm{dom}_Z(f^*) \cap N(y)\right)$ as a continuous orange line. The black area represents the source of non-convexity. 335

Figure D.5. Performance of MINLP solvers 336

Figure D.6. Representation of the discrete decisions of the reactor network problem in the Boolean variables domain (A) and the external variables domain (B). 338

Appendix E

Figure E.1. Additional details of the steps in the proposed hybrid algorithm (Figure 5.2). A shared folder allows processors 1 and 2 to communicate with each other. 340

Figure E.2. Step D4 of Figure 5.2: The Discrete-Steepest Descent Algorithm with Variable Bounding (DSDA-VB). Superscript q is the iteration counter used within DSDA-VB. 341

Figure E.3. Illustrative example of the strategy of the algorithm in Table E.2 (i.e., Step B5 in Figure 5.2). A) iteration 1, bound 1, B) iteration 1, bound 2 347

Figure E.4. Illustrative example of the strategy of the algorithm in Table E.2 (i.e., Step B5 in Figure 2). A) iteration 2, bound 1, B) iteration 2, bound 2 348

Figure E.5. Performance of the proposed parallel-hybrid strategy and a nested-hybrid strategy. A: Comparison using a time limit of 5000 seconds. B: augmented version of A. 354

Appendix F

Figure F.1. Solution of NLP dynamic multi-scenario subproblems with e fixed..... 373

Appendix G

Figure G.1. A): analysis of the problem from the traditional convexity perspective, B): analysis of the problem from the discrete-convex-analysis perspective. 399

Appendix H

Figure H.1. Control actions and concentrations for the MLD-BD approach (A and B) and the SIA initialization (C and D). 426

List of Tables

Chapter 3

Table 3.1. Results: solution status, feasibility, and continuous variables using a rate-based model for simulation and economic optimization. 64

Table 3.2. Solution times and objective function values for different solvers and initialization points. 68

Chapter 4

Table 4.1. Steps required to obtain benders cuts at iteration k 80

Chapter 5

Table 5.1. Variables and their bounds for the first case study, and best solution found. 113

Table 5.2. Best solution found by each run of the hybrid algorithm (Figures 5.5B-5.5C). 115

Table 5.3. First iteration of the hybrid algorithm. 116

Table 5.4. Variables and their bounds for the second case study, and best solution found. 121

Table 5.5. Convergence of objective functions f_B^k and f_s^k, number of stages (RD) and reflux ratio (RD) for iterations in which variable bounds are updated. 123

Chapter 6

Table 6.1. Problem variables 152

Table 6.2. Constraints of the problem: feasible region Y 154

Table 6.3. Constraints of the problem: feasible region S 155

Table 6.4. Length to diameter ratio 158

Table 6.5. Optimal design variables and costs: sequential and simultaneous approaches. 161

Table 6.6. Steady state operating variables (inputs) and transition times. 162

Chapter 7

Table 7.1. Comparison of initialization and solution strategies (case study 1) 188

Table 7.2. Comparison of initialization and solution strategies (case study 2) 195

Chapter 8

Table 8.1. Steps to obtain multiple benders cuts at iteration k, and refine previously generated cuts. 214

Table 8.2. Performance of MLD-BD and GBD solution strategies (case study 1, $\eta = 50400$ s). 224

Table 8.3. Performance of MLD-BD solution strategies (case study 1, $\eta = 100{,}800$ s) 227

Table 8. 4. Comparison of the MLD-BD (p,h) approach against the SIA and IIA (case study 2) 230

Appendix C

Table C.1. Notation used through this section..... 317

Appendix E

Table E.1. Iterative strategy to initialize continuous variables bounds $x^{L,M}$ and $x^{U,M}$ from $x^{L,S}$ and $x^{U,S}$ 344

Table E. 2. Bound relaxation using a bisection strategy to update upper and lower bounds of continuous variables..... 345

Table E.3. Initialization of bounds to begin the execution of the SM (iteration $k = 0$)..... 350

Table E. 4. Information required to initialize DSDA-VB (iteration $k = 1$). 351

Table E.5. Solution obtained by DSDA-VB (iteration $k = 1$). 353

Table E. 6. Iterations for case study 1 (Figure 5.5A in chapter 5)..... 355

Table E.7. Iterations for case study 1 (Figure 5.5B in chapter 5). 356

Table E.8. Iterations for case study 1 (Figure 5.5C in chapter 5). 360

Table E.9. Iterations for case study 1 (Figure 5.5D in chapter 5)..... 362

Table E.10. Iterations for case study 2 (Figure 5.8 in chapter 5). 364

Appendix F

Table F.1. Sets and superstructure size parameters. 376

Table F.2. Binary variables (y) and parameters..... 376

Table F.3. Real variables ([φ, χ(t), ζ(t)]) and parameters..... 376

Table F.4. Set subindices..... 378

Table F.5. Superscripts..... 379

Appendix G

Table G.1. Iterative-sequential approach to initialize the D-SDA..... 391

Table G.2. Maximum and minimum capacity of the units (case study 1)..... 392

Table G.3. Parameters related to states $m \in M$ (case study 1)..... 392

Table G.4. Parameters related to pairs $(i, j) \in IJ^{\times}$ (case study 1)..... 392

Table G.5. Parameters in the dynamic models (case study 1)..... 393

Table G.6. Variables in the dynamic models (case study 1)..... 393

Table G.7. Domain of ordered discrete decisions (case study 1)..... 394

Table G.8. Maximum and minimum capacity of the units (case study 2)..... 394

Table G. 9. Parameters related to states $m \in M$ (case study 2)..... 395

Table G.10. Parameters related to pairs $(i, j) \in IJ^{\times}$ (case study 2)..... 395

Table G.11. Parameters in the dynamic models (case study 2)..... 396

Table G.12. Variables in the dynamic models (case study 2)..... 397

Table G.13. Domain of ordered discrete decisions (case study 2)..... 397

Appendix H

Table H.1. Maximum and minimum capacity of the units (case study 1)..... 413

Table H.2. Parameters related to states $m \in M$ (case study 1)..... 413

Table H.3. Parameters related to pairs $(i, j) \in IJ^{\times}$ (case study 1)..... 413

Table H.4. Parameters in the dynamic models (case study 1)..... 413

Table H.5. Variables in the dynamic models (case study 1)..... 414

Table H.6. Domain of ordered discrete decisions (case study 1) 415
Table H.7. Problem statistics (case study 1, $\eta = 50400$ s)..... 417
Table H.8. Problem statistics (case study 1, $\eta = 100800$ s) 417
Table H.9. Maximum and minimum capacity of the units (case study 2)..... 417
Table H.10. Parameters related to states $m \in M$ (case study 2) 418
Table H.11. Parameters related to pairs $(i, j) \in IJ^{\times}$ (case study 2)..... 418
Table H.12. Reactor parameters in the dynamic models (case study 2)..... 418
Table H.13. Reactor variables in the dynamic models (case study 2)..... 419
Table H.14. Distillation parameters in the dynamic models (case study 2) 419
Table H.15. Distillation variables in the dynamic models (case study 2) 420
Table H.16. Domain of ordered discrete decisions (case study 2) 420
Table H.17. Problem statistics (case study 2)..... 424

List of Abbreviations

CD	Catalytic Distillation – Incorporates heterogeneous reaction and separation in the same unit
DAEs	Differential Algebraic Equations – Model involving both algebraic equations and differential equations
DE	Differential Evolution – population-based metaheuristic search algorithm
DETL	Differential Evolution with Tabu List – DE with Tabu list heuristic to improve performance
DOF	Degrees of freedom – Independent variables that, when fixed, transform an optimization problem into a simulation, i.e., primary decision variables.
D-SDA	Discrete Steepest Descent Algorithm – MINLP algorithm that relies on neighborhood exploration
DSDA-VB	D-SDA with variable bounding – DSDA with heuristic for simulation-based optimization
ED	Extractive Distillation – Distillation aided with a solvent feed flow for non-ideal mixture separation
EAC	Equivalent Annual cost – Annualized sum of costs minus sum of revenues of a system
ETBE	Ethyl tert-butyl ether – Organic compound
FNLP	Fixed Nonlinear Programming – NLP that results when discrete variables are fixed
GBD	Generalized Benders Decomposition – Benders decomposition applied to MINLP with cuts based on Lagrangean duality
GDP	Generalized Disjunctive Programming – Alternative representation of MINLP problems
IP	Integer Programming – Unconstrained optimization problem involving integer variables

IPLC	Integer Problem with Linear Constraints – Linearly constrained IP
ITAE	Time-weighted Absolute Error – Controller performance measure
LBBD	Logic-Based Benders Decomposition – Abstraction of the Benders decomposition idea to tackle a larger class of problems
LD-BD	Logic-Based Discrete-Benders Decomposition – Algorithm based on LBBD for problems involving ordered discrete decisions
LD-SDA	Logic-Based Discrete Steepest Descent Algorithm – GDP algorithm that relies on neighborhood exploration
LHHW	Langmuir- Hinshelwood-Hougen-Watson – kinetic model for reaction over a catalyst
L-OA	Logic-Based Outer Approximation – GDP algorithm based on the outer approximation principle
LP	Linear Programming – Continuous variable problem with linear constraints and linear objective function
MIP	Mixed Integer Programming – Mixed integer-continuous variable problem with linear constraints and linear objective function
MINLP	Mixed Integer Nonlinear Programming – Non-convex mixed integer-continuous variable problem with nonlinear constraints and nonlinear objective function
MEK	Methyl Ethyl Ketone – Organic compound
MESH	Mass, Equilibrium, Summation, entHalpy – stage-by-stage mass and energy balance model for distillation
MERSHQ	Mass, Energy, Rate, Summation, Hydraulic, and interface eQuilibrium – non-phase-equilibrium stage-by-stage mass and energy balance model for distillation
MTBE	Methyl tert-butyl ether – stage-by-stage mass and energy balance model for distillation
NLP	Nonlinear Programming – Non-convex continuous variable problem with nonlinear constraints and nonlinear objective function

NRTL	Non-random two-liquid – Nonideal model for vapor-liquid equilibrium calculations
PI	Process Intensification – Field that deals with the improvement of production processes
RD	Reactive Distillation – Incorporates reaction and separation in the same unit
SSDO	Simultaneous Scheduling and Dynamic Optimization – Problem that integrates scheduling decisions with open-loop control
SM	Stochastic method – Optimization algorithm that relies on random search criteria. Not to be confused with stochastic optimization to consider uncertainty
SQP	Sequential Quadratic Programming – NLP algorithm that relies on iterative quadratic problem approximations
STN	State Task Network – Graphical representation and formulation to deal with the scheduling of multiproduct batch production environments
TAC	Total Annual Cost – Annualized sum of the costs of a system
TAME	Tert-amyl methyl ether – Organic compound
1,3-BD	1,3-butadiene – Organic compound
2,3-BD	2,3-butanediol – Organic compound
2MPL	2-methylpropanal – Organic compound
3B2OL	3-buten-2-ol – Organic compound

List of Symbols

Latin symbols

arg min	Arguments of the minima
arg max	Arguments of the maxima
a_n^I	Interfacial area at stage n $[m^2]$
AF	Annualizing factor $[-/year]$
$b_{i,j,t}$	Continuous variable: batch size of task i when processed at unit j starting at time t
$b_{i,j,t}^{shift}$	Continuous variable: Auxiliary continuous variable 1 in the STN formulation
b_j^{max}	Parameter: Maximum capacity of unit j
b_j^{min}	Parameter: Minimum capacity of unit j
BR	Continuous variable: Boil-up ratio
$COMP$	Set of components
$COMP_R$	Set of all components except one
CT	Total number of components
$COND$	Condenser stage set
C	Set of distillation columns in a flowsheet (chapter 5)
CR	Crossover rate
C_A	Continuous variable: concentration of component A (case study 3, chapter 4)
C_B	Continuous variable: concentration of component B (case study 3, chapter 4)
co	Convex hull
c^{fix}	Fixed cost term (SSDO)
$c_{i,j}^{fix}$	Coefficient for fixed cost calculation (SSDO)

c^{var_1}	Variable cost for task-unit pairs without dynamic models (SSDO)
$c_{i,j}^{var_1}$	Coefficient for c^{var_1} cost calculation (SSDO)
c^{var_2}	Variable cost for task-unit pairs with dynamic models (SSDO)
c^{mat}	Cost of raw materials (SSDO)
c_m^{mat}	Coefficient for raw material cost calculation (SSDO)
c^{hot}	Coefficient for hot flow cost calculation (SSDO)
c^{cold}	Coefficient for cold flow cost calculation (SSDO)
c_n^L	Molar density (liquid) $[mol/m^3]$
c_n^V	Molar density (vapor) $[mol/m^3]$
C_0	Fixed annualized investment cost $[\$/year]$
C_{inv}	Annualized investment cost $[\$/year]$
C_{Op}	Annual operating costs $[\$/year]$
C_{ETBE}	Annualized ETBE profit $[\$/year]$
C'_{ETBE}	ETBE product cost $[(\$/year)(s/mol)]$
C_{cat}	Catalyst cost $[\$/kg]$
C_1	Annualized cost of ethanol feed $[(\$/year)(s/mol)]$
C_2	Annualized cost of butenes feed $[(\$/year)(s/mol)]$
C_{Cond}	Annualized condenser overall duty cost $[(\$/year)(s/kJ)]$
C_{Reb}	Annualized reboiler overall duty cost $[(\$/year)(s/kJ)]$
$\bar{C}_i$	Cost/selling price of i $[\$/mol]$
C_{IN}	Time-dependent incomes $[\$/year]$
$\bar{C}_r/\bar{C}_c$	Reboiler/condenser operating cost $[\$/kJ]$
dom	Effective domain

DYN	Feasible set defined by dynamic optimization constraints
D_q	Set of disjuncts for disjunction q
D^k	Set with ordered variables evaluated prior and up to iteration k
$\widetilde{D}^k$	Set with ordered variables evaluated at iteration k
d_c	Variable: diameter of distillation column c $[m]$
D_c	Column diameter $[m]$
$\boldsymbol{e}$	Vector of integer external variables
$e_{i,j}$	External variable that defines the processing time of pair (i,j) in SSDO
E	Set that defines the search space of $\boldsymbol{e}$
f	Objective function
f^*	Optimal subproblem objective function value
f_0^*	Optimal subproblem objective function value in SSDO (standard subproblem)
f_1^*	Optimal subproblem objective function value in SSDO (proposed subproblem 1)
$f_{\widetilde{0}}^*$	Optimal subproblem objective function value in SSDO (proposed subproblem 2)
$f_{\widetilde{1}}^*$	Optimal subproblem objective function value in SSDO (proposed subproblem 3)
$\bar{f}$	Convex envelope of f
$\tilde{f}$	Local convex extension of f
f_B^k	Best objective function at iteration k
f_1	Scheduling-related objective term in SSDO
f_2	Scheduling and control objective term in SSDO
F	Mutation factor (chapter 5); set of feeds (chapter 3)
F_S	Database with feasible solutions
F_S^0	Initialization of F_S

f_S^k	Best objective function value from F_S at iteration k
f_D^k	Locally optimal objective function at iteration k
f_{OBJ}	EAC objective function [$\$/year$]
f_{SUB}	Subproblem objective
FT	Total number of feeds
$\boldsymbol{g}$	Vector of inequality constraints
$\widehat{\boldsymbol{g}}$	Vector of equality constraints
$\boldsymbol{g}^D$	Vector of design inequality and equality constraints over discrete and continuous variables
$\boldsymbol{h}_{d,q}$	Vector of inequality constraints in disjunction d of disjunct q
$\boldsymbol{h}_{i,j,t}$	Indexed vector of constraints that define algebraic equations and inequalities in SSDO
$\widehat{\boldsymbol{h}}_{d,q}$	Vector of equality constraints in disjunction d of disjunct q
$\widehat{\boldsymbol{h}}_1$	Vector of equality constraints 1 (case study 2, chapter 4)
$\widehat{\boldsymbol{h}}_2$	Vector of equality constraints 2 (case study 2, chapter 4)
$\widehat{\boldsymbol{h}}_{i,j,t}$	Indexed vector of constraints that define differential equations in SSDO
$\boldsymbol{h}^{SS}$	Vector of time-dependent constraints (steady state)
$\boldsymbol{h}^{TR}$	Vector of time-dependent constraints (transient)
h_c	Variable: height of distillation column c [m]
h_s	Stage height [m]
h_{total}	Total column height [m]
h_w	Weir height [m]
H_n^L	Enthalpy of liquid leaving stage n [kJ/s]
H_n^V	Enthalpy of vapor leaving stage n [kJ/s]

I	Set of tasks
$I_{i,m}^{-}$	Binary parameter: 1 if task i consumes material from state m
$I_{i,m}^{+}$	Binary parameter: 1 if task i produces material to state m
$I_{i,j}^{\times}$	Binary parameter: 1if task i can be performed at unit j
$IJ^{\times}$	Set with tuples to denote allowed task-to-units mapping
$IJ^{\times,dyn}$	Subset of $IJ^{\times}$ with dynamic models
I_1	Set 1 (case study 1, chapter 4)
IN_f^L	Liquid feed low of input f $[mol/s]$
IN_f^V	Vapor feed flow of input f $[mol/s]$
IN_1	Input feed of pure ethanol $[mol/min]$
IN_2	Input feed of butenes $[mol/min]$
INT	Internal stages set
J	Set of units
J_2	Set 2 (case study 1, chapter 4)
J_k	Set of sampling points (periods) of scenario k
K	Set of scenarios
$k_{n,i,m}^L$	Multicomponent mass transfer coefficient at n between i and m (liquid) $[m/s]$
$k_{n,i,m}^V$	Multicomponent mass transfer coefficient at n between i and m (vapor) $[m/s]$
$K_{n,i}^I$	Interphase equilibrium constant of component i at stage n
k_{rate}	Parameter: kinetic rate constant (case study 3, chapter 4) $[l/mol\ s]$
$LINK$	Feasible set defined by linking constraints
L_n	Liquid flow leaving stage n $[mol/s]$

LR_n	Reflux recirculation flow at stage n $[mol/s]$
L_{ETBE}	Output molar Flow of ETBE $[mol/min]$
min	Minimization
max	Maximization
M	Set of states
M^{in}	Subset of input states
M^{out}	Subset of output states
$m_{cat,n}$	Mass of catalyst at stage n $[kg]$
$M\&S$	Marshall and Swift parameter
ns_u	Number of sites in rate-determining step for reaction u
N	Number of stages (chapter 5), Set that defines the search space of $\boldsymbol{n}$ (chapter 8); set of stages (chapter 3)
NT	Total number of stages of the superstructure
N_∞	Integrally convex neighborhood
N_2	Separable convex neighborhood
N_Θ	Neighborhood of type Θ
$N_{i,j}$	Set that defines the search space of $n_{i,j}$
n_x	Number of variables in $\boldsymbol{x}$
n_y	Number of variables in $\boldsymbol{y}$
n_z	Number of variables in $\boldsymbol{z}$
n_e	Number of variables in $\boldsymbol{e}$
n_{J_k}	Number of sampling points (periods) of scenario k
n_K	Number of time scenarios

n_{φ}	Number of time-independent continuous variables
n_{μ}	Number of inputs
n_{χ}	Number of differential variables
n_{ζ}	Number of algebraic variables
nc_c	Number of catalytic stages of column c
n_T	Number of discretization points in scheduling horizon
$n_{i,j}$	Integer batching variable: number of times task i is processed in unit j
$\boldsymbol{n}$	Vector with batching variables $n_{i,j}$ in SSDO
$\boldsymbol{p}$	Normal vector of a cutting plane
$\boldsymbol{p}^{k,\theta}$	Vector with elements of matrix $\boldsymbol{P}^k$ for central or neighboring point θ at iteration k
$P(\omega)$	Percentage of runs that converged to the global optimum (case study 3, chapter 4)
$\boldsymbol{P}^k$	Matrix with wights of row vector $\boldsymbol{n}$ to calculate cuts obtained at iteration k
P_n	Pressure at stage n $[Pa]$
$P_{ow,n}$	Pressure drop of the liquid over the stage $[Pa]$
ΔP_n	Pressure drop at stage n $[Pa]$
$\Delta P_{C,n}$	Pressure drop due to catalyst $[Pa]$
Q_{Cond}	Condenser duty $[kJ/s]$
Q_{Reb}	Reboiler duty $[kJ/s]$
Q	Set of disjunctions
Q_i	Set of chemical species involved in reaction i
Q_R	Continuous variable: Reboiler duty $[kW]$
Q_C	Continuous variable: Condenser duty $[kW]$
Qr_c	Continuous variable: Reboiler duty of column c $[kW]$

Qc_c	Continuous variable: Condenser duty of column c $[kW]$
$\boldsymbol{Q}^k$	Matrix with wights of row vector $\boldsymbol{e}$ to calculate cuts obtained at iteration k
$\boldsymbol{q}^{k,\theta}$	Vector with elements of matrix $\boldsymbol{Q}^k$ for central or neighboring point θ at iteration k
RR	Continuous variable: Reflux ratio
REB	Reboiler stage set
RX	Set of reactions
r_A	Reaction rate expression with respect to component A (case study 3, chapter 4)
rev	Revenue
$R^L_{n,i,m}$	Elements of inverted mass transfer coefficients matrix (liquid) $[s/m]$
$R^V_{n,i,m}$	Elements of inverted mass transfer coefficients matrix (vapor) $[s/m]$
RX	Total number of reactions
SF	Safety factor
$s_{m,t}$	Continuous variable: inventory of state m at time t
s_m^{max}	Parameter: maximum storage capacity of state m
s_m^{init}	Parameter: initial storage in stage m
$S_{Feasible}$	Feasible region of $[\boldsymbol{x}, \boldsymbol{Y_R}]$
S	Set that represents the feasible region of discrete ($\boldsymbol{y}$) and continuous variables ($\boldsymbol{v}$).
SCH	Feasible set defined by scheduling constraints
$sp_{product}$	Product set point
t	Time
t_k	Final time of scenario k $[min]$
t_k^{TR}	Transition point of scenario k $[min]$
t_p	Operating factor (final optimization time) $[min/year]$

$\Delta t_{k,j}$	Size of sampling interval of sampling point (period) j and scenario k $[min]$
$\Delta \boldsymbol{t}_k$	Vector with sizes of sampling intervals for scenario k $[min]$
T	Set of scheduling time points (chapters 7,8); Temperature $[K]$ (chapter 3)
TLS	Tabu list size
TR	Tabu radius
T_n^L	Liquid bulk temperature at stage n [K]
T_n^V	Vapor bulk temperature at stage n [K]
T_n^I	Interphase temperature at stage n [K]
T_n^{BUBBLE}	Bubble point temperature of liquid bulk at stage n [K]
T_n^{DEW}	Dew point temperature of vapor bulk at stage n [K]
$u_{L,n}$	Liquid velocity through the downcomer at stage n $[m/s]$
$u_{V,n}$	Vapor velocity through the stage at stage n $[m/s]$
$u_{h,n}$	Vapor velocity through the holes at stage n $[m/s]$
$\boldsymbol{v}$	Vector of continuous design and operation variables
$\boldsymbol{v}^{sch}$	Vector with all scheduling variables in SSDO
$\boldsymbol{v}^{sch}_{\tau_{i,j},i,j,m,t}$	Indexed vector of scheduling variables in SSDO
$\boldsymbol{v}^{sch*}_{\tau_{i,j},i,j,k,t}$	Optimized vector of variables $\boldsymbol{v}^{sch}_{\tau_{i,j},i,j,m,t}$ (i.e., current subproblem solution)
$\boldsymbol{v}^{dyn}$	Vector with all dynamic optimization variables in SSDO
$\boldsymbol{v}^{dyn}_{i,j,t}$	Indexed vector of dynamic operation variables in SSDO
$\boldsymbol{v}^{dyn*}_{i,j,t}$	Optimized vector of variables $\boldsymbol{v}^{dyn}_{i,j,t}$ (i.e., current subproblem solution)
V_n	Vapor flow leaving stage n $[mol/s]$
VB_n	Boil-up recirculation flow at stage n $[mol/s]$

$v_{product}$	Composition of desired product (or products)
w	Number of operation cycles
$\boldsymbol{x}$	Vector of continuous variables
$\boldsymbol{x}^*$	Optimized vector of variables $\boldsymbol{x}$ (i.e., current subproblem solution)
$\boldsymbol{x}^{sch}$	Vector of continuous scheduling variables
$\boldsymbol{x}^{sch*}$	Optimized vector of variables $\boldsymbol{x}^{sch}$ (i.e., current subproblem solution)
$\tilde{\boldsymbol{x}}^{sch}$	Optimized vector of variables $\boldsymbol{x}^{sch}$ with cut off (i.e., current subproblem solution)
X	Set that defines the search space of $\boldsymbol{x}$
$x_{D,Benzene}$	Continuous variable: Distillate benzene composition
$x_{B,Toluene}$	Continuous variable: Bottoms toluene composition
x_{ETBE}	Composition of ETBE [$\%\, mol/mol$]
$x_{i,j,t}$	Binary variable: 1 when unit j processes task i starting at time t
$x_{i,j,t}^{sum}$	Continuous variable: Auxiliary continuous variable 2 in the STN formulation
$x_{n,i}$	Liquid bulk composition at stage n form component i
$x_{n,i}^{I}$	Liquid interphase composition at stage n form component i
$y_{n,i}$	Vapor bulk composition at stage n form component i
$y_{n,i}^{I}$	Vapor interphase composition at stage n for component i
$\boldsymbol{Y}$	Vector of Boolean variables
$\boldsymbol{Y_R}$	Vector of ordered Boolean variables
$\boldsymbol{Y_1}$	Vector of ordered Boolean variables 1 (case study 1, chapter 4)
$\boldsymbol{Y_2}$	Vector of ordered Boolean variables 2 (case study 1, chapter 4)
Y_n	Boolean variable that represents the existence of stage n (case study 2, chapter 4)
y_n	Binary variable that represents the existence of stage n (case study 2, chapter 4)

YB_n	Existence of boil-up at stage n
YC_n	Existence of catalyst at stage n
YE_n	Existence of mass transfer at stage n
$YF_{n,f}$	Existence of feed f at stage n
YP_n	Existence of *real stage* at stage n
YR_n	Existence of reflux at stage n
$Y_{\tau_{i,j},i,j}$	Boolean variable: indicates if task i in unit j requires $\tau_{i,j}$ discrete time points
$\boldsymbol{y}$	Vector of ordered discrete binary/Boolean variables. For chapter 5, this represents integer variables, because a reformulation with external variables is not needed
$\boldsymbol{y}^*$	Fixed current value of $\boldsymbol{y}$ (i.e., D-SDA or master problem assignment)
$\boldsymbol{y}^{sch}$	Vector of binary/Boolean variables that do not have an ordered structure
$\boldsymbol{y}^{sch*}$	Optimized vector of variables $\boldsymbol{y}^{sch}$ (i.e., current subproblem solution)
$\tilde{\boldsymbol{y}}^{sch}$	Optimized vector of variables $\boldsymbol{y}^{sch}$ with cut off (i.e., current subproblem solution)
Y	Feasible region of binary variables
$\boldsymbol{z}$	Vector of output variables of interest in simulation-based optimization
$\boldsymbol{z}^*$	Optimized vector of variables $\boldsymbol{z}$ (i.e., current subproblem solution)
$z_{i,f}^L$	Liquid molar composition of component i in feed f
$z_{i,f}^V$	Vapor molar composition of component i in feed f

Greek symbols

α	Offset of a cutting plane
α_1	Continuous variable 1 (case study 1, chapter 4)
$\boldsymbol{\alpha}^k$	Vector with offset term of cuts obtained at iteration k
$\alpha^{k,\theta}$	Offset term of a cut for central or neighboring point θ at iteration k

$\alpha_{k,j}$	Sampling point (period) j of scenario k $[min]$
$\beta_{\boldsymbol{e}^k}$	Cut obtained after solving subproblem at trial point $\boldsymbol{e}^k$
$\beta_{[\boldsymbol{n},\boldsymbol{e}]^{k,\theta}}$	Cut obtained after solving subproblem at trial point $[\boldsymbol{n},\boldsymbol{e}]^{k,\theta}$
$\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^k}$	Vector of cuts obtained after solving subproblem at trial point $[\boldsymbol{n},\boldsymbol{e}]^k$
β_2	Continuous variable 2 (case study 1, chapter 4)
$\boldsymbol{\delta}$	Steepest descent direction
δ	Tolerance for objective function improvement criterion
$\delta_{n,i,m}$	Kronecker delta at stage n
ϵ	Optimality tolerance
ε_1	Tolerance of swichability
ε_2	Tolerance of continuity constraints
$\rho_{i,m}^{-}$	Parameter: fraction of material processed by task i that is coming from state m
$\rho_{i,m}^{+}$	Parameter: fraction of material processed by task i that is dispatched to state m
ρ_{cat}	Catalyst's apparent density $[kg/m^3]$
ω	Number of sampled discrete configurations (case study 3, chapter 4)
ω_{comp}^{str}	Mass fraction of component $comp$ in stream str
Ω	Logic relationships
$\widehat{\Omega}$	Feasible set representing a black-box simulation
η	Duration of scheduling horizon
$\kappa_{n,i,m}^{L}$	Binary mass transfer coefficient at n between i and m (liquid) $[m/s]$
$\kappa_{n,i,m}^{V}$	Binary mass transfer coefficient at n between i and m (liquid) $[m/s]$
$\tau_{i,j}$	Parameter: number of points in the discrete-time grid to perform task i in unit j

$\tau_{i,j,t}^{P}$	Continuous variable: processing time of task i in unit j at time t
$\tau_{i,j}^{P,fixed}$	Parameter: fixed processing time of task i in unit j at time t
$\tau_{i,j}^{P,min}$	Parameter: user-defined processing time lower bound
$\tau_{i,j}^{P,min}$	Parameter: user-defined processing time upper bound
$\Gamma_{n,i,m}^{L}$	Thermodynamic factor at n between i and m (liquid)
$\Gamma_{n,i,m}^{V}$	Thermodynamic factor at n between i and m (vapor)
$\boldsymbol{\mu}$	Vector of control inputs (i.e., manipulated variables)
$\boldsymbol{\mu}_{i,j,t}$	Indexed time-dependent vector of manipulated variables
$\mu_{i,j,t}^{hot}$	Continuous variable: manipulated hot flow of task i in unit j at time t
$\mu_{i,j,t}^{cold}$	Continuous variable: manipulated cold flow of task i in unit j at time t
$\boldsymbol{\nu}_{i}$	Stochiometric coefficients of reaction executed in task i
$\nu_{i,u}$	Stoichiometric coefficient of component i in reaction u
χ	Vector of differential variables (i.e., state variables)
$\boldsymbol{\chi}_{i,j,t}$	Indexed time-dependent vector of state variables
$\dot{\boldsymbol{\chi}}_{i,j,t}$	Indexed time-dependent vector of state variables' derivatives in SSDO
$\boldsymbol{\zeta}$	Vector of algebraic variables
$\boldsymbol{\zeta}_{i,j,t}$	Indexed time-dependent vector of algebraic variables in SSDO
$\boldsymbol{\varphi}$	Vector of time-independent continuous variables
$\boldsymbol{\varphi}_{i,j,t}$	Indexed time-independent vector of continuous variables in SSDO
$\varphi_{i,j}^{P}$	Processing time of task-unit pair (i,j) in the dynamic model of the SSDO formulation
$\varphi_{i,j,t}^{Vol}$	Total hold-up of task-unit pair (i,j) at time t in dynamic models of the SSDO formulation
ψ	Dimensionless time

Other symbols

$[\boldsymbol{a}, \boldsymbol{b}]$	Notation used to indicate that vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ are merged as a single vector
$\lceil a \rceil$	Ceil operator
$\lfloor a \rfloor$	Floor operator
$\emptyset$	Empty set
$\|\boldsymbol{a}\|_\infty$	Infinity norm (ℓ_∞) of vector $\boldsymbol{a}$
$\|\boldsymbol{a}\|_2$	Euclidean norm (ℓ_2) of vector $\boldsymbol{a}$
$\|A\|$	Cardinality of set A
$\leqq$	Equal or less than or equal (depending on modeler choice)
$\exists$	There exists
$\subseteq$	Subset of
$\subset$	Proper subset of
$\in$	Belongs to
$\vee$	Or
$\wedge$	And
$\neg$	Negation
$\veebar$	Exclusive or
∞	Infinity
$\Rightarrow$	Material implication
$\Leftrightarrow$	Equivalent
$\mapsto$	Map arrow
$\rightarrow$	Reaction arrow

$a \leftarrow b$	Update a with the value of b
$\cup$	Union
$\cap$	Intersection
$A \backslash B$	Set A without the elements of B
$\forall$	For all
$\langle \boldsymbol{a}, \boldsymbol{b} \rangle$	Dot product between vector $\boldsymbol{a}$ and $\boldsymbol{b}$
$\mathcal{E}_n$	Interphase heat flow at stage n $[kJ/s]$
$\mathcal{F}_{total}^{str}$	Total molar flow rate in stream str
$\mathcal{M}_{comp}^{str}$	Mass flow of component $comp$ in stream str
$\mathcal{N}_{n,i}$	Interphase molar flow at stage n for component i $[mol/s]$
M	Big-M
$\mathbb{R}$	Set of real numbers
$\mathbb{R}^n$	n-dimensional real space
$\mathcal{R}_{n,u}$	Reaction rate of chemical equation u at stage n $[mol/kg]$
$\mathrm{T}_{i,j}$	Set with processing time options $\tau_{i,j}$
χ_{comp}^{str}	Molar fraction of component $comp$ in stream str
$\mathbb{Z}$	Set of integers
$\mathbb{Z}^n$	n-dimensional integer space
z	Real variable: master problem objective function

Set subindices

c	Index of set C (chapter 5); Index of sets $COMP$, $COMP_R$ (chapter 3)
d	Index of set D_q
f	Index of set F

i	Index of set I (chapters 7,8); Index of sets $COMP$, $COMP_R$ (chapter 3)
i_1	Index of set I_1 (case study 1, chapter 4)
j	Index of set J (chapters 7,8); Index of sets $COMP$, $COMP_R$ (chapter 3); Index of set J_K Denoting a sampling point or period (chapter 6)
j_2	Index of set J_2 (case study 1, chapter 4)
k	Index of set K denoting a scenario or time slot (chapter 6)
l	Index of sets $COMP$, $COMP_R$
q	Index of set Q
t	Index of set T (chapters 7,8); Index of sets N, $COND$, REB, INT (chapter 3)
u	Index of set RX
m	Index of set M(chapters 7,8); Index of sets $COMP$, $COMP_R$ (chapter 3)
n	Stage index (chapter 5); Index of sets N, $COND$, REB, INT (chapter 3)

Subscripts

inv	Investment
I	Independent
Op	Operating
OBJ	Objective
$Cond$	Condenser
Reb	Reboiler
$user$	User-defined

Superscripts

$+$	Denotes consumption
$-$	Denotes production

$\times$	Used to denote allowed task-unit pairs
θ	Benders cut identifier when generating multiple cuts per iteration
$best$	Used to denote the best objective/variables identified so far
$BUBBLE$	Bubble point
DEW	Dew point
dyn	Dynamic
$init$	Initialization
in	Input
$iter$	D-SDA iteration
$inner$	Line search iteration
inv	Investment
I	Variable initialization from stochastic optimization (chapter 5); Interphase (chapter 3)
k	Main Iteration counter
$\hat{k}$	Auxiliary iteration counter 1
$\tilde{k}$	Auxiliary iteration counter 2
L	Lower algorithmic bound (chapter 5); Liquid (chapter 3)
$L, 0$	User-defined lower bound
L, M	Modified Lower bound
L, S	lower bound initialization from stochastic optimization
LS	Line search
o	Used to denote ordered variables values that define a minimizer within a neighborhood
out	Output
sch	Scheduling

SS	Steady state
TR	Transient state/ transition
T	Matrix and vector transpose notation
U	Upper algorithmic bound
$U, 0$	User-defined upper bound
U, M	Modified upper bound
U, S	Upper bound initialization from stochastic optimization
V	Vapor

“Tantas idas y venidas, tantas vueltas y revueltas
(quiero, amiga, que me diga), ¿son de alguna utilidad?
Yo me afano, mas no en vano.
Sé mi oficio, y en servicio de mi dueño tengo empeño de lucir mi habilidad.”

“Comings and goings, turns and twists,
idle freaks, heed who lists. All this no useful purpose speaks.
Not so futile, my endeavor, in my duty faithful ever.
My master to serve, I strain every nerve to be always prompt and clever.”

- *Tomás de Iriarte. English interpretation by Geo. H. Devereux*

Chapter 1
Introduction

Nowadays, in the context of Industry 4.0, process plants are transitioning from traditional manufacturing practices to automated computer-supported solutions able to integrate decision-making within manufacturing enterprises. This paradigm shift aims to increase profits, optimize resource utilization efficiency, promote long-term sustainability, and enhance responsiveness under uncertainties. Accordingly, the development of reliable Mixed Integer Nonlinear Programming (MINLP) and Generalized Disjunctive Programming (GDP) algorithms capable of simultaneously handling design, planning, scheduling, or control decisions are crucial to achieving Industry 4.0 integration goals. The lack of integration between these layers has resulted in sub-optimal and even infeasible operation strategies[1]. However, the integrated optimization of any of these layers usually results in large-scale optimization problems when applied to traditional and intensified chemical processes. These problems represent a challenge because finding optimal solutions is non-trivial due to their non-convex nature and model size.

One of the main difficulties faced by the existing optimization strategies is the adequate treatment of the discrete variables that emerge in design and scheduling problems, and their inherent interaction with the nonlinear equations of the control problem. Discrete variables in design and scheduling naturally introduce non-convexities. These non-convexities, when combined with non-linear control factors, lead to complex problems that make it hard to find not just the global optimum, but even local optima. Moreover, they pose convergence difficulties when trying to find feasible solutions, due to their sensitivity to variables' initialization and search bounds. Algorithmic and theoretical developments are thus needed to alleviate the computational complexity of integrated decision-making problems. In particular, the current literature suggests the exploitation of special characteristics in scheduling, design and control problems (e.g., decomposable structures, hidden convexity, sets of binary variables with ordered structures, tight inequalities, lower bounds, etc.) for the development of tailored algorithms[2]. A common characteristic between scheduling, design and control problems is that they often involve ordered discrete decisions, e.g., number of units in series/parallel in a reactor superstructure; selection of the most sensitive stage for distillation temperature control applications; number of times a task is executed during a production schedule. The ordered nature of some of these variables is lost when embedded through binary or Boolean variables in MINLP or GDP formulations, respectively. Recently, it has been found that ordered structures can be recaptured through the reformulation of binary/Boolean

variables into integer *external variables*, allowing ordered discrete decisions to be optimized in a master optimization layer[a]. Other advantages of external variables include decreasing the number of degrees of freedom and decreasing the number of discrete variables, e.g., a group of multiple binary variables can be reformulated into a single external variable. Then, the decomposed problem can be solved efficiently using an adequate algorithm such as the Discrete-Steepest Descent Algorithm (D-SDA)[3,4]. Note that the groups of binary/Boolean variables from which external variables are obtained resemble restricted special ordered set of type 1 (restricted-SOS1) variables, i.e., a group of variables where exactly one of the variables is non-zero in the final solution[5]. Similarly, the ordered binary/Boolean variables herein considered enforce exactly $k \in \mathbb{N}$ non-zero variables in the final solution. Ordered binary/Boolean variables also impose the additional requirement that the set over which a group of binary/Boolean variables is defined has an intuitive order in the sense that it must contain ordinal, discrete interval or discrete ratio data, e.g., discrete locations in space, discrete points in time, etc.

The D-SDA is the only MINLP algorithm in the literature that has its roots on Discrete Convex Analysis (DCA). The latter algorithm aims to provide theoretical foundations for the deterministic optimization of Integer Programming (IP) problems. The key characteristic of DCA-based algorithms such as the D-SDA is that they rely on neighborhood explorations of ordered discrete decisions, e.g., the neighborhood for the number of stages in the design of a distillation column is defined by increasing and decreasing the number of stages by one unit. Although the search regions of integer variables are inherently non-convex, DCA provides notions of convexity for integer variables, such as integral convexity. Accordingly, if a given problem satisfies a DCA-based convexity requirement (e.g., integral convexity), then global optimality can be ensured by local exploration with a suitable DCA-based algorithm. In practice, local optimization techniques are commonly used to solve challenging MINLP or GDP optimization problems, despite their convexity characteristics. Due to their reliance on neighborhood verification, a key benefit of DCA-based optimization techniques is that they may converge to a local solution with respect to ordered discrete decisions, which may not be attained by general-purpose MINLP solvers[6]. Nevertheless, the D-SDA has some practical limitations, e.g., it requires the user to provide a feasible initialization, it is limited to local optimization, it is not suitable to handle large-scale MIDO problems, it is currently limited to address problems that result in NLP

[a] Through this thesis, ordered discrete decisions generalize the notion of integer variables, by considering both the integer variables present in the original problem formulation and the integer external variables obtained from the reformulation of a GDP or MINLP problem.

subproblems, and it limits is search to a vicinity of an user-provided initialization[7]. Furthermore, its performance has not been investigated when handling flowsheet design problems involving multiple unit operations, scheduling problems, or integrated decision-making problems. Thus, algorithmic developments and research studies are still needed to understand the benefits that DCA-based algorithms may bring to the optimization of chemical engineering problems.

Another promising tool to explore the domain of ordered discrete decisions in integrated decision-making chemical engineering problems involves the inclusion of cuts throughout the solution procedure. Cuts are constraints that allow to exclude variable's combinations that are undesirable (non-optimal or infeasible), without evaluating them explicitly. The generation of cuts become necessary in algorithms based on Benders decomposition, where cuts allow to update decision variables in an efficient manner, thanks to the decomposition of the problem into *complicating* and *non-complicating* variables in a *master problem* and a *subproblem*, respectively [8]. *Complicating variables* are those that reduce the complexity of the optimization problem when they are fixed in the analysis, e.g., when fixing discrete variables in an MINLP, the resulting problem is a Nonlinear Programing (NLP) problem. Without loss of generality, existing Benders based algorithms, e.g., Benders Decomposition (BD) for Linear Programing (LP) subproblems, can be categorized into the more general Logic-Based Benders Decomposition (LBBD), which does not establish a particular requirement over the form of the subproblem [9]. Besides, the general LBBD has demonstrated reductions in solution times with respect to traditional Mixed Integer Programing (MILP) implementations, when applied to chemical process scheduling and hybrid control problems[10]. Nonetheless, a feature of LBBD is that its cuts must be designed on a case-by-case basis. Hence, its application as a DCA-based algorithm to address integrated decision-making problems is not straightforward, thus leaving the door open for further research.

Despite the main interest of this thesis is to tackle integrated decision-making problems, many optimal design problems in chemical engineering are regarded as challenging MINLP problems, e.g., when incorporating multiscale modeling features, or when considering the optimal design of multiple interconnected units simultaneously. Part of this thesis is focused on single-unit or multiple-unit flowsheet design optimization problems, which are regarded as Mixed-Integer Nonlinear Programming (MINLP) problems. Formulating and initializing MINLP flowsheet design problems may require modeling expertise, and constructing accurate mechanistic models for systems with complex thermodynamic equilibrium and hydrodynamic relationships can be challenging and time-consuming. Typically, these problems are non-convex and challenging to solve due to the nonlinearity of non-ideal

thermodynamic models, mass, and energy balances in chemical processes; the inherent non-convexity of discrete variables, and the nonlinearities that arise from the interactions of discrete and continuous variables. Hence, optimizing the design of rigorous flowsheet models can turn into a computationally expensive and time-consuming process. Note that this thesis is limited to the solution of MINLP/GDP whose continuous part involves smooth functions with smooth derivatives where first and second derivatives are available for an NLP optimizer such as CONOPT[11].

1.1 Research Objectives

Although ordered discrete variables are common in the formulation of design, scheduling and control optimization problems, only a few works in the literature have studied strategies based on DCA to explore the domain of these variables efficiently, when dealing with single-unit superstructure design problems. Therefore, this work aims to make significant progress in the development of MINLP and GDP optimization algorithms based on DCA to efficiently address process design, simultaneous scheduling and control and simultaneous design and control problems of high relevance in chemical engineering. To achieve this goal, this thesis focuses on the following specific objectives:

1. Develop new DCA-based optimization techniques that alleviate the limitations of the conventional D-SDA, by combining DCA and LBBD theory.
2. Broaden the scope of DCA-based algorithms to tackle applications in chemical engineering by addressing problems in unexplored fields including multiscale intensified systems, design problems involving multiple unit operations and integrated scheduling-control and design-control problems.
3. Develop tailored search strategies (e.g., algorithm hybridization, decomposition techniques and heuristics) that exploit special problem structures and incorporate them within the DCA-based algorithms investigated in this work to improve their efficiency.
4. Propose optimization models for the integrated decision-making problems studied through this thesis including the optimal transient design of a catalytic distillation column undergoing dynamic transitions, and the simultaneous scheduling and dynamic optimization of a network batch plant with variable processing times.

1.2 Contributions

The research conducted in this PhD thesis provides the following contributions:

1. The application of the D-SDA to solve a general multiscale MINLP problem for the synthesis and design of catalytic distillation (CD) columns, using a type of nonequilibrium formulation called rate-based model. The herein proposed model can be easily extended to similar case studies, such as other stage separation processes, e.g., reactive distillation, conventional distillation, extractive distillation, among others. The D-SDA has already demonstrated its superiority to find optimal configurations for the design of CD columns with equilibrium models when compared to existing MINLP solvers[12]. Therefore, the D-SDA is used to solve the resulting MINLP problem, demonstrating its potential against conventional MINLP solvers to deal with more complex multiscale and multiphase systems.

2. The Logic-based Discrete-Benders Decomposition (LD-BD) algorithm, which incorporates a novel strategy to obtain cuts. Overall, LD-BD combines the advantages of a logic-based algorithm for GDP (e.g., solution of subproblems in reduced space); a LBBD strategy[b] (e.g., customizable Benders cuts), and DCA (e.g., exploitation of ordered structures). In contrast to standard MINLP techniques, solving GDP subproblems in reduced space means that only the constraints and variables of active disjunctions are considered at a given iteration of the algorithm, which results in faster subproblem convergence and avoids potential numerical issues that may arise when constraints are inactive. Furthermore, this thesis proposes the first cutting strategy that relies on DCA principles by using information from a solution candidate and its neighborhood to generate cuts that approximate the convex envelope of the objective function. The cuts generated at each iteration are constantly refined within the proposed LD-BD algorithm, which guarantees convergence to a local optimum within a discrete neighborhood.

3. The work also presents the first flowsheet design methodology that executes a deterministic algorithm and a stochastic algorithm (e.g., differential evolution, genetic algorithm, etc) in parallel, which is a feature that has not been exploited by the available hybrid strategies for

[b] Different definitions of "Logic-based" are used in the literature depending on the context. In the context of GDP, logic-based algorithms are those able to retain the logic structure of a GDP formulation by solving subproblems with fixed Boolean variables in reduced space[13]. In the context of LBBD, logic-based refers to the idea of using logical inference in optimization methods through the generation of cuts that exploit special problem structures[10]. LD-BD can be categorized as both a logic-based GDP and a LBBD algorithm.

optimal design of process flowsheets that makes use of chemical process simulators. This ultimately leads to solutions that satisfy local optimality for both discrete and continuous degrees of freedom (DOF), i.e., concepts from DCA are used to verify the local optimality of discrete DOF, while the local optimality requirement of continuous variables is assessed by an NLP solver. These optimality criteria are not guaranteed by the available methodologies for flowsheet design available in the literature; hence the novelty of this work. This results in the first application of a DCA-based algorithm to optimize problems involving flowsheet simulators.

4. A general MIDO formulation for the economic optimal design of multitask CD units that enables the simultaneous consideration of process dynamics, e.g., set-point changes and disturbances. This formulation is flexible and adaptable to other types of distillation process or intensified systems. An enhanced D-SDA is used as a deterministic optimization methodology that allows to systematically solve this demanding optimization problem, by taking advantage of the special structure of the MIDO problem. This enhanced D-SDA has a modular structure, allowing the designer to decide if process dynamics should be considered or not at the design stage depending on the computational resources available to perform the optimal process design.

5. A modeling framework that generalizes the discrete-time State Task Network (STN) scheduling model to simultaneous scheduling and dynamic optimization (SSDO) problems with variable processing times by introducing a new GDP formulation. The solution of this problem brings novelty by improving and expanding the D-SDA to consider MINLP/GDP applications other than superstructure optimization. This shows how the D-SDA can be used to take advantage of a special structure often found in discrete-time SSDO formulations, namely the presence of ordered discrete decisions such as those related to variable processing times and batching variables.

6. The first application of LBBD for SSDO through the implementation of a customized Multicut Logic-based Discrete Benders Decomposition (MLD-BD) algorithm. MLD-BD incorporates a multicut generation strategy based on DCA that allows to tackle the non-convexities introduced by the relaxation of ordered discrete decisions. The MLD-BD method is further enhanced by incorporating a pruning strategy that can be combined with heuristics. The pruning strategy reduces computational time, without affecting the local properties of the solution obtained. Further reductions in solution time can be obtained when combining the pruning strategy with

heuristics, which may be advantageous in situations when the computational time is a limiting factor, e.g., online applications.

1.3 Structure of the Thesis

The thesis presented herein is structured as follows:

Chapter 2 is divided into three parts. First, it provides introductory definitions and examples of design, design and control and scheduling and control problems. Second, it provides literature review on the available deterministic MINLP optimization techniques and tools for integrated decision-making in chemical engineering. Third, this chapter provides an overview of the main optimization algorithms that serve as foundation for the MINLP algorithms proposed and studied in this thesis. This chapter is part of a review that has been submitted for publication to the Reviews in Chemical Engineering journal[14].

Chapter 3 studies the optimal synthesis and design of a rigorous CD column that explicitly considers the multiscale and multiphase nature of this intensification process. A rate-based model that couples micro and macroscale events taking place inside the CD column is proposed. To solve this problem, the D-SDA optimization strategy enables the designer to improve an initial feasible design, while providing local optimality guarantees. A case study involving the production of ethyl tert-butyl ether (ETBE) is considered. This study was published in the Chemical Engineering Science journal[7].

Chapter 4 introduces the LD-BD algorithm for GDP optimization. The key idea behind LD-BD is to obtain Benders cuts that use neighborhood information of a reformulated version of Boolean variables. A mathematical case study, the optimization of a network with CSTRs in series, and a test problem involving the design of a distillation column are considered to demonstrate the features of LD-BD. This study was published in the AIChE Journal[15].

Chapter 5 deals with the simulation-based optimal design of multiple interconnected separation units. A hybrid algorithm that combines a stochastic method (SM) with the D-SDA in parallel is proposed. The D-SDA is improved with a Variable Bounding strategy (VB) aimed to address de difficulties of simulation-based optimization, resulting in the DSDA-VB algorithm. The proposed method is illustrated using a thermally coupled system and a sequence of reactive, extractive, and traditional distillation columns. This study was published in the Computers & Chemical Engineering journal[16].

Chapter 6 considers the optimal design of multitask CD columns in the transient domain. The proposed optimization method is based on the D-SDA, which is improved in this study to enable dynamic optimization efficiently. The production of ETBE is considered as a case study, where dynamic transitions between different grades of ETBE coupled with design considerations are investigated. This study was published in the Chemical Engineering and Processing-Process Intensification journal[17].

Chapter 7 extends the application of DCA-based algorithms to address the SSDO of network batch plants with variable processing times using a discrete-time formulation. Given that this is an optimization problem that has not been previously studied in the literature, this chapter introduces a GDP model of the problem. This GDP formulation is exploited by a LD-SDA that directly solves the GDP problem without the need of a GDP-to-MINLP reformulation. The performance of the LD-SDA solution strategy is illustrated using case studies adapted from the literature. This study was published in the Industrial & Engineering Chemistry Research journal[18].

Chapter 8 builds upon the modeling and optimization advances from chapter 7 in the field of simultaneous scheduling and dynamic optimization. This chapter introduces MLD-BD as a more efficient strategy to address the SSDO of network batch plants with variable processing times. The proposed algorithm relies on the generation of multiple cuts per iteration thus producing accurate approximations of the objective function in shorter computational times. This study was published in the AIChE journal[19].

Chapter 9 presents the concluding remarks achieved by the present research work. This chapter also provides recommendations for potential future work.

Appendices A and B provide supporting results for chapters 6 and 8, respectively. These two appendices where published in the conference proceedings of the 13th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems (DYCOPS 2022)[20], and the 2024 American Control Conference (ACC)[21]. Appendices C to H provide supporting information for chapters 3 to 8, respectively.

Chapter 2
Background and Literature Review

This thesis proposes algorithms that can be extended across the different decision-making and integrated decision-making stages in a production facility that deals with MINLP problems. However, this thesis only considers applications in the fields of optimal process design, optimal design and control and optimal scheduling and control. Accordingly, this chapter begins by providing background and illustrative examples of optimal design and simultaneous design and control (section 2.1), as well as simultaneous scheduling and control (section 2.2). Optimization strategies to solve these problems can be generally categorized into deterministic and stochastic (i.e., metaheuristic) techniques. Stochastic methods are not able to recognize a locally optimal solution, and their stopping criteria are based on the fulfilment of requirements based on heuristics, e.g., stop when there is no significant improvement during the last iterations. On the other hand, deterministic methods are based on mathematical theorems that ensure that the solution obtained is better than any other neighboring solution, e.g., local optimality guarantees are ensured by the necessary and sufficient conditions in continuous optimization. Most of the algorithmic developments presented through this thesis belong to the field of deterministic MINLP optimization, hence this chapter reviews the most widely used deterministic MINLP optimization algorithms in chemical engineering in section 2.3. A review of general-purpose MINLP solvers (i.e., those that are readily available in algebraic modeling systems such as GAMS and Pyomo) is presented in section 2.3.1, while section 2.3.2 provides background on MINLP tools that are particularly relevant in the field of optimal integrated decision-making. Section 2.4 then provides an overview of the optimization tools that are the backbone of the algorithms presented through this thesis, namely the logic-based optimization of GDP problems, the Discrete-Steepest Descent Algorithm (D-SDA), Discrete Convex Analysis (DCA), Logic-Based Benders decomposition, and the hybridization of deterministic algorithms with stochastic algorithms. Section 2.5 provides a summary of this chapter. It is important to clarify that, throughout this thesis, stochastic optimization methods are referred to as black-box (often population-based) heuristic methods that rely on randomness in their iterations, e.g., genetic algorithms, particle swarm optimization, differential evolution. These algorithms are regarded as metaheuristics, in the sense that they are supported by arguments of convergence to a global optimum in infinite time, however, their success is not well grounded in a mathematically proven stopping criterion[22]. Furthermore, the optimization problems considered through this thesis do not incorporate any type of uncertainty.

2.1 Optimal process design and simultaneous design and control

Optimal design problems generally rely on the postulation of a superstructure model, i.e., a formulation that includes binary or Boolean variables to model a variety of design candidates, each with a varying number of unit operations and/or multiple flow interconnection options between these unit operations, e.g., number of reactors in series/parallel, number of stages in multistage heat exchanger networks, number of stages and feed stage locations within distillation sequences, among others. Furthermore, each unit operation involved has continuous design variables (e.g., reactor size, distillation column diameter, etc.), and continuous operation variables (e.g., heat duties, operation pressure and temperature, etc.). The constraints in the formulation of these problems include nonlinear mass and energy balances, product quality requirements and feasible operation regions, e.g., the temperature of a reactor cannot exceed a certain limit to avoid undesired reactions. The objective function of these problems is usually economic such as equivalent annual cost or total annual cost minimization, but safety and sustainability measures can also be incorporated in the objective. When an economic objective is chosen, optimal design is expected to offer economic improvements with respect to traditional design methods that rely on heuristics and trial-and-error strategies that do not consider all the constraints of the problem simultaneously, e.g., sequential adjustments of heat exchanger designs until pressure drops are below the maximum allowable values and design requirements are satisfied; or the sequential design of distillation columns using shortcut methods.

Optimal design problems can be extended to consider the transient operation of the dynamic systems involved. Design and control problems include control considerations in logical and operational constraints through closed-loop or open loop dynamic models. Control and process dynamic considerations are often included in the objective function through various measures, e.g., the integral squared error of a controlled variable with respect to its setpoint or by assigning an economic cost to the closed-loop transient operation[23,24]. Thus, simultaneous consideration of process design and control decisions results in an economic and sustainable design with improved dynamic responsiveness under disturbances, uncertainties or set-point changes. Figure 2.1 illustrates a design and control problem relevant to MINLP optimization, which is the optimal sizing of a multiproduct continuous distillation column, which in this case has a bottoms product with different purities A and B, e.g., see [25–28]. Both nonlinearities and discrete decisions appear in the design part of this problem due to the consideration of non-linear mass/energy balances and the presence of discrete variables such as the number of trays and feed location; thus making the design problem by itself a MINLP. Moreover, the consideration of

set-point change A→B and B→A or other disturbance scenarios through discretized differential equations make the NLP part of this problem more complex than an optimal design problem at steady state. The solution of this problem with an economic objective function through simultaneous design and control may return a profitable design that is able to accommodate the production of both A and B, while keeping dynamic transitions A→B and B→A feasible and inexpensive.

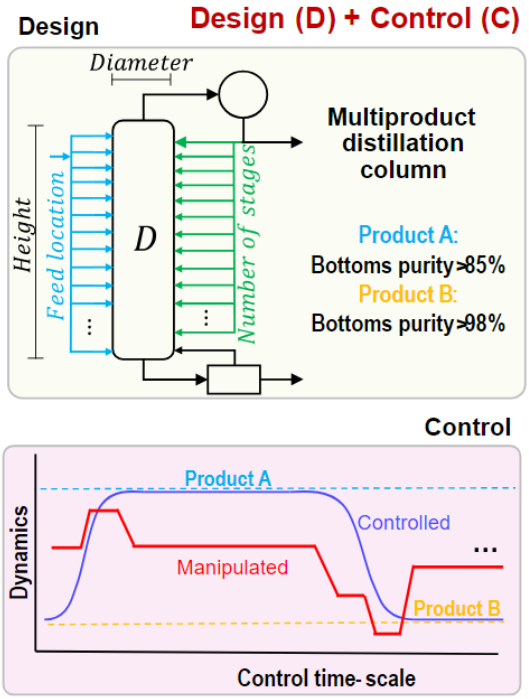


Figure 2.1. Illustrative design and control problem

Both stochastic and deterministic optimization methods have been widely used to solve MINLP flowsheet design problems[29,30]. On the one hand, the resulting problem can be optimized using deterministic mathematical programming techniques such as those based on branch and bound, outer approximation, logic-based solvers, NLP techniques based on continuous reformulations, among others[31] (see section 2.3.1 for further details). The key feature of deterministic strategies is that they are designed to converge to a locally or globally optimal solution, provided that a well-behaved algebraic representation of the functions of the problem is available, e.g., the objective function and constraints typically should be differentiable when using an NLP solver. On the other hand, stochastic optimization techniques typically generate a set of solution candidates based on random sampling. Then, they iteratively improve the objective function of the solution candidates, by performing black-box

simulations and evaluations of the objective function. They are useful for complex optimization problems with non-convex relationships, high dimensionality, and ill-behaved functions where traditional optimization methods may not be efficient[32]. Among the available stochastic techniques, evolutionary algorithms such differential evolution and genetic algorithms are the most prominent metaheuristics in the literature for optimal flowsheet design, although many others are available, e.g., particle swarm optimization, simulated annealing, and tabu search[33]. Other emerging techniques include reinforcement learning approaches[34,35].

Both deterministic and stochastic optimization techniques exhibit limitations when solving process flowsheet optimization problems. Local optimization strategies highly depend on the initialization of continuous and discrete decisions, due to "zero flow" numerical issues [15]. Also, finding a global solution to non-convex problems is not guaranteed with local solvers [36]. Conversely, traditional global deterministic MINLP methods may be computationally prohibitive when dealing with large-scale flowsheet design problems that consider continuous and discrete variables and rigorous nonlinear models for multiple processing units [37,38]. Furthermore, the performance of both local and global deterministic solvers may be impacted by the level of detail of the process models and the formulation selected to represent the interactions between continuous and discrete variables, e.g., Big-M, convex hull, etc. In contrast, stochastic methods that use randomized search strategies and exploration techniques that are not tied to specific model assumptions or simplifications. This makes stochastic methods especially suitable to consider the simulation software as a black box. Despite their advantages, black box stochastic approaches have other limitations such as a large number of function evaluations, which may result in slow convergence, or refined parameter tuning to achieve optimal performance [39]. Further research is thus needed in different fields to develop more efficient solution techniques for optimal design problems in chemical engineering.

2.2 Simultaneous Scheduling and Control

The intensive research in modeling and optimization algorithms for chemical production scheduling is allowing an ongoing transition from manual scheduling to automated computer-supported scheduling solutions that offer multiple advantages, e.g., cost savings, reduction in waste production, and efficient utilization of resources [40]. In particular, the combination of scheduling and control decisions is a problem that has been studied over the last two decades, both in the forms of simultaneous scheduling and dynamic optimization (SSDO) and integrated scheduling and control (ISC). SSDO is generally

regarded as the open-loop optimization problem that incorporates both scheduling and control decisions in the formulation, while ISC addresses the closed-loop iterative optimization of scheduling and control variables [41]. ISC and SSDO aim to incorporate the closed-loop/open-loop dynamic models of unit operations into the scheduling problem, which aims to find the optimal start times, durations, and processed volumes of different tasks within a time horizon with respect to an objective function that is usually economic. ISC and SSDO thus allow the generation of economic production plans that comply with the dynamic operation capabilities of the unit operations involved in the process. Figure 2.2 illustrates a SSDO problem where a continuous multiproduct distillation column is preceded by two batch operations: reaction and filtration. The scheduling part of this problem introduces discrete decisions when modeling the batching, sequencing, and timing of reaction and filtration tasks, while finding the optimal production sequence of products A and B based on a demand scenario. Furthermore, the consideration of dynamic models to optimize the control actions of key unit operations such as the reactor and the distillation column introduces nonlinear constraints to the formulation, resulting in a MINLP formulation. The SSDO of this problem is expected to yield a production schedule that is more likely to satisfy the demand requirements for products A and B, by avoiding the potential inconsistences that may arise with traditional sequential scheduling and control solutions, e.g., not being able to satisfy a reaction production target, due a selection of batch sizes and processing times in the scheduling layer that is blind to reactor control considerations.

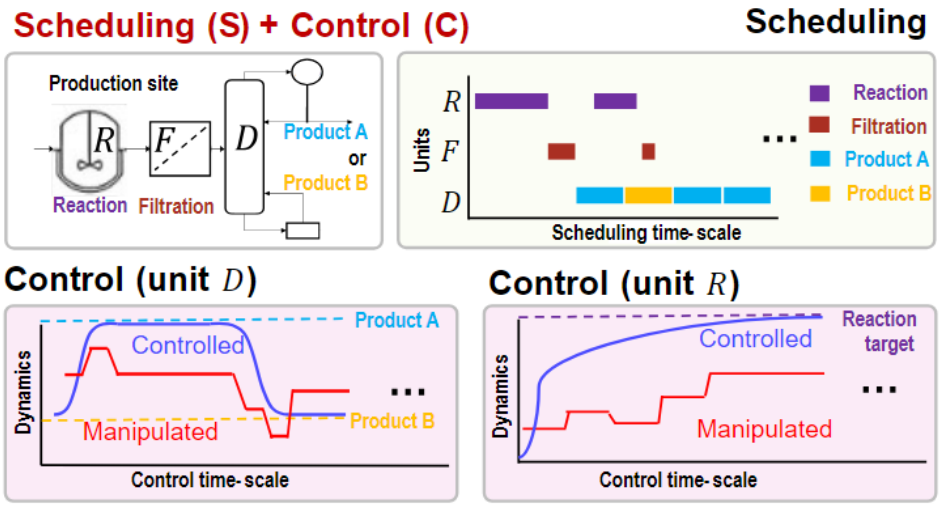


Figure 2.2. Illustrative scheduling and control problem

The available strategies to interlink scheduling and control decisions are often classified as *top-down* and *bottom-up* approaches. Top-down approaches incorporate non-linear constraints that represent the dynamic operation of the system into the scheduling problem, whereas bottom-up strategies aim to incorporate scheduling and economic decisions at the decision-control layer[42,43]. On the one hand, bottom-up approaches aim to avoid discontinuities between scheduling and process control decisions, by explicitly incorporating scheduling and economic considerations at the supervisory or regulatory control level, e.g., through economic nonlinear model predictive control. One of the main limitations of bottom-up implementations is their inability to adequately handle discrete decisions for large-scale systems (e.g., sequencing variables), due to restrictions in providing adequate control actions in real-time, which hinder the solution of a potentially non-convex MINLP problem at every sampling instant. On the other hand, top-down approaches exhibit more flexibility to handle discrete variables, as their computational time requirements are less tight. Also, top-down strategies can benefit from the wide variety of existing scheduling models for their implementation[44]. Nevertheless, top-down strategies maintain a hierarchical separation between scheduling and control that introduces non-uniformity, i.e., although the scheduling layer is augmented with nonlinear dynamic considerations, the supervisory and regulatory control layers remain in the hierarchical scheme[43]. This may result in discrepancies between the scheduling and control layers in online top-down implementations, e.g., the dynamic model in the scheduling layer may not accurately represent the closed-loop dynamics of the plant. Different strategies to mitigate these discrepancies have been proposed in the literature, e.g., allowing the scheduling layer to access plant measurements[45], and approximating closed-loop dynamics in the scheduling layer through MPC using first principles or data-driven surrogates[46,47].

Regardless of the modeling strategy (e.g., top-down, bottom-up) or feedback strategy (e.g., SSDO, ISC) selected, combined scheduling and control problems share a common challenge, i.e., the solution of a MINLP problem, which results from the integration of integer variables and nonlinear dynamic models [48]. The solution of this computationally-demanding MINLP often requires advanced and specialized solution strategies. For example, both top-down approaches (i.e., include nonlinear process dynamics in the scheduling model) and bottom-up methods (i.e., include scheduling decisions at the control level) should cope with computational difficulties of the underlying MINLP optimization problems[42,49]. In the case of top-down strategies, the computational time is limited by the frequency at which production scheduling is performed, e.g., production scheduling may be performed on a daily basis[50]. Time limits are even more restrictive for bottom-up approaches, which typically require solving

an optimization problem within the controller sampling time; thus making bottom-up approaches challenging to implement[43] (e.g., simultaneous economic model predictive control and scheduling). Thus, the development of efficient MINLP optimization algorithms is still needed to facilitate the implementation of simultaneous scheduling and control solutions for real-time applications, and to increase the commercial adoption and implementation of ISC and SSDO solutions [2,42]. Note that, despite the efforts to deal with integrated scheduling and control in academia (e.g., see [51,52]), such integration is not widely accepted by industrial practitioners yet[53]. While this thesis focuses on optimization performance as a fundamental (and still challenging) limitation to enable the integration, other relevant factors that are also holding back the integration include 1) closed-loop strategies that adequately use feedback from uncertain environments to handle set-point changes and disturbances, 2) effective strategies to handle large volumes of historical/current data coming from different sources in complex real-world systems, 3) continuous synchronization between the real system and predictions coming from models in a simulated environment, 4) changes in organizational and human behavior to implement new decision-making tools, and 5) development of commercially available software prototypes for optimal process integration[54].

2.3 Deterministic MINLP optimization

2.3.1 General-Purpose MINLP Solvers

According to Liberti et al.[55], the motivation to develop general-purpose deterministic MINLP solution algorithms in chemical engineering originated from seminal studies such as those by Geoffrion[56], who proposed the Generalized Benders Decomposition (GBD) algorithm and Grossman[57,58], who first outlined the handling of nonlinearities in the optimal synthesis of process flowsheets through Branch and Bound (BB) and Outer Approximation (OA) techniques. The reader is referred to the review by Kronqvist et al.[36] for further details about the working principle of these strategies. What these strategies have in common is that they were originally conceived as local optimization algorithms able to guarantee global optimality under certain convexity conditions, e.g., BB and OA guarantee global optimality for convex MINLPs, i.e., when continuously relaxing binary or integer requirements, the resulting problem is convex. GBD, BB and OA have thus served as foundation to develop many local and global MINLP optimization algorithms and solvers through the years.

Nowadays, there are a variety of commercial and open source general-purpose MINLP solvers available to the public, e.g., see [36,59]. the solvers that have been substantially applied in the context of

integrated decision-making include DICOPT, which is based on the OA concept; SBB, which is based on the BB principle, and BARON, which is based on BB, constraint propagation and duality techniques that make it suitable for global optimization. Note that GBD (as a general-purpose solver) is closely related and usually requires more iterations than OA; hence, GBD is not widely used in practice for general MINLP optimization. Nonetheless, GBD has been successfully used to exploit special problem structures and tackle integrated decision-making problems[60–62]. Customized GBD methods leverage the special structure of integrated decision-making problems by separating them into a master problem that considers one of the decision layers (e.g., scheduling) and a subproblem that considers the remainder of the problem (e.g., control decisions). The resulting decomposed problem can then be solved iteratively, i.e., at each iteration the variables of the master problem are fixed in the subproblem, whose solution is used to update the master problem with a new constraint called a Benders cut. Other MINLP general-purpose solvers include those based on extended cutting planes[63] and extended supporting hyperplanes[64], however these have not been widely applied in the context of integrated decision-making.

Early attempts to address integrated MINLP decision-making problems appeared in the context of optimal design and scheduling of multipurpose batch plants that consisted of finding the optimal number of processing units and their capacities to minimize the total capital cost of the plant, while simultaneously accommodating the production of multiple products within a fixed time horizon[65,66]. These problems were addressed by relying on custom formulations that allowed solving the problems through standard NLP techniques. Later, with the advancement of MINLP solution techniques, studies started to emerge in the in the fields of design and control, planning and scheduling, scheduling and control and their combinations. Some of these applications are summarized below.

Generalized Benders Decomposition: GBD has been used for the simultaneous design and control of separation units including conventional[67–69] and heat-integrated[68] separation systems. GBD has also been used to solve the optimal design and scheduling of multipurpose batch plants[70]. More recently, GBD has been used to address the simultaneous scheduling and control of multipurpose batch plants with network structures[61,62], and the design and control of a reactive distillation column[71].

Branch and Bound: BB-based strategies have been mostly used to address problems in the field of simultaneous scheduling and control, with applications in multiproduct continuous and batch manufacturing processes[72,73], single-unit multiproduct reactors[74], and heat exchanger networks under

fouling[75]. BB has also been used in other fields, e.g., to support the simultaneous design and control of reactor networks[76], or the planning and scheduling of supply chains in the oil and chemical industries[77].

Outer Approximation: OA techniques have been applied to a broader range of applications, thanks to availability of OA-based solvers such as DICOPT in GAMS, AOA in AIMMS, OAERAP in gPROMS, among others. These applications include the design and control of conventional distillation columns[67], semi-batch polymerization reactors[78], a system of interconnected continuous stirred tank reactors (CSTR)[79], and a semi-continuous distillation columns[80]. For scheduling and control, OA applications include the cyclic scheduling and control of a multiproduct CSTR[81] and a polymerization reactor[82]. OA has also been applied to multiple design-planning and design-scheduling problems such as the optimal strategic planning and design of a network of petrochemical production processes[83]; the design and scheduling of desalinization systems[84,85], among others. Furthermore, OA has allowed the integration of three or more decision layers, e.g., the simultaneous planning, scheduling and control of single-unit systems[86,87], the simultaneous design, scheduling and control of a polymerization reactor[88], and the simultaneous design, planning, scheduling and control of multiproduct batch plants[89].

Despite the success of general-purpose MINLP solvers, several studies have demonstrated the drop in performance or even failure of general-purpose MINLP solvers and highlighted the need of more efficient solution strategies when dealing with integrated decision-making problems, e.g., see [61,90–93]. General-purpose MINLP solvers may require restrictively long computational times, identify low quality local solutions (e.g., get stuck at the initialization point), or even fail to find feasible solutions. For example, Flores-Tlacuahuac and Biegler[79] compared the performance of SBB and DICOPT when tackling the simultaneous design and control of two CSTRs in series with up to 2647 continuous variables and 3905 constraints. That study showed that DICOPT led to solver failure in some instances, due to poor linearization of non-convex functions. Valdez-Navarro and Ricardez-Sandoval[94] compared the performance of SBB, DICOPT and BARON to address the simultaneous scheduling and control of network batch processes under uncertainty and found that increasing the number of uncertain scenarios has a negative impact on traditional solver performance. The most challenging case study they tested involved 361 realizations in 2 uncertain parameters, 446271 continuous variables, 66 binary variables and 450365 constraints. The authors found that SBB, DICOPT and BARON failed to identify a feasible solution to the problem after one day of computation, even though the problem was known to be feasible. Lozano-Santamaria and Macchietto[75] found that standard general-purpose MINLP solvers may be time intensive and require solution times in the order of magnitude of days when solving

realistic simultaneous scheduling and control problems of heat exchanger networks under fouling; thus, more efficient strategies were suggested to find solutions in shorter computational times. Typically, one of the key factors affecting general-purpose solvers' performance is the convexity/non-convexity characteristics of the problem at hand and how these characteristics may lead to different local solutions depending on the initialization. A second factor is the type and number of nonlinear expressions that are part of the formulation, and how these may impact the solver's success to detect feasible solutions or run into numerical instabilities. A third factor is the number of discrete variables in the problem and their interactions with continuous variables, leading to problems with challenging combinatorial structures. These issues have motivated the development of alternative deterministic MINLP solution techniques for integrated decision-making, this is discussed next.

2.3.2 MINLP Tools to Aid Integrated Decision-making

This section provides an overview of the MINLP algorithms and frameworks for integrated decision-making (i.e., other than those mentioned in the previous section) including Mixed Integer Programming (MIP) reformulations, continuous reformulations, and Lagrangean Relaxation. Other relevant tools not included in this section are multiparametric programming (e.g., see[95,96]), bilevel optimization (e.g., see[97]), and back-off algorithms (e.g., see [94,98]). These three strategies do not necessarily focus on facilitating the solution of an underlying single-level MINLP problem resulting from the integration; instead, these techniques aim to facilitate the incorporation of other features that arise in integrated decision-making, such as the presence of conflicting objectives, uncertainties, and disturbances.

2.3.2.1 Mixed Integer Programming Reformulations

In MIP reformulations, nonlinearities are reformulated as linear relationships between integer and continuous variables. MIP reformulations of MINLP problems may involve exact linearization, e.g., when linearly reformulating the multiplication of two binary variables, or the multiplication of binary and continuous variables. Exact linearization have been successfully applied to integrated scheduling and planning problems involving batch processing plants[99,100]. In more general scenarios, exact linearization transformations may not be available and MIP input-output approaches that only provide an approximation of the original MINLP may be needed. These approaches rely on the approximation of nonlinearities as linear operators, leading to a formulation that can be readily solved with MIP methods, at the expense of having a model that is valid within a reduced space where the linearization was performed. This approach has been successfully applied to capital investment planning and

scheduling of an oil refinery[101]. To overcome the limitations of MIP input-output approaches, piecewise linear reformulations of data-driven low-order models (e.g., Hammerstein-Wiener models, and piecewise affine models) were proposed to represent nonlinear process dynamics in simultaneous scheduling and control problems. This strategy has been tested for instance on a continuous CSTR with cyclic production[102], a continuous multiproduct polymerization reactor[103] and an air separation unit[104]. A similar strategy was employed to simultaneously consider planning, scheduling and dynamic optimization of batch processes, where dynamic models were replaced by linear surrogate models, resulting in a MIP formulation[105]. Piecewise linear approximations that transform a MINLP into an MIP have also been considered for the planning and scheduling of flexible process networks under uncertainty[106]. Data-driven and surrogate methods may simplify the effort of model development; however, they require training and validation of data sets that are assumed to cover/predict the entire operating range of the plant.

2.3.2.2 Continuous Reformulations

In continuous reformulations, discrete decisions are expressed as continuous variables and discrete requirements are enforced via additional constraints. Continuous reformulations have been developed to reformulate MINLPs as Nonlinear Programs (NLPs) to reduce the high computational costs often faced by MINLP and MIP techniques. Examples of reformulations include complementarity constraints, circle constraints, nonlinear complementarity problem functions, among others[31,107,108]. These techniques often require the iterative optimization of the resulting NLP with varying tolerance parameters that either enforce integer requirements as they go to zero, or the consideration of integer requirements through objective function penalization. Despite this, they are usually faster than traditional MINLP solvers. Integrated decision-making MINLP applications include the integration of optimal scheduling and control of heat exchanger networks under fouling[109], scheduling and NMPC of a multiproduct continuous reactor[110], simultaneous design and control of plug flow and CSTR reactor networks[111], design and control of extractive distillation column systems[112]. The main drawback of continuous reformulations is that they inevitably lead to a nonconvex optimization problem that generally requires educated initial guesses to achieve convergence[107].

2.3.2.3 Lagrangean Relaxation

Lagrangean relaxation corresponds to the general idea of penalizing a subset of complicating constraints in the objective function of an optimization problem, i.e., complicating constraints are

dualized. The nonnegative weights used to penalize these constraints are known as Lagrangean multipliers[113]. An application of Lagrangean Relaxation that has become relevant in the context of MINLP integrated decision-making in chemical engineering is known as Lagrangean Decomposition (LD), which is a useful approach to exploit optimization problem structures where two (or more) subset of constraints interconnected through common variables appear in the problem formulation. For instance, in simultaneous scheduling and control, some constraints and variables are used to model scheduling decisions, and others are used to capture transient operation, i.e., dynamic path constraints. These two groups of constraints are often connected through common variables. In the case of two subsets of constraints, if common variables are duplicated, the linking constraints that connect these variables can be dualized resulting in a decomposable formulation where each subset of constraints considers independent variables[113].

Note that the value of Lagrangean multipliers used to penalize constraints is initially unknown. Hence, once a decomposable problem is obtained, an iterative approach is used to update the Lagrangean multipliers. For example, chemical engineering applications in the field of planning, scheduling and control of a multiproduct CSTR[114], or the scheduling and control of polymerization reactors[115] have implemented the sub-gradient method developed by Fisher[116] to iteratively find and update Lagrangean multipliers. Other relevant works in the field include the simultaneous supply chain design and planning in the electric motor industry[117], and the integration of refinery planning and crude-oil scheduling[118,119]. In general, LD helps to reduce the computational cost of solving integrated decision-making problems, against the direct solution without decomposition techniques[42]. Note that there are some challenges in the application of LD to MINLP integrated decision-making. For instance, an adequate strategy should be used to update Lagrangean multipliers, e.g., the algorithm may not converge if the sub-gradient step size is not defined or updated properly. Furthermore, problem-specific heuristics may be needed to solve the resulting MINLP subproblems efficiently[118].

2.4 Relevant Optimization Tools for this Thesis

As discussed above, significant progress has been made in the development of MINLP optimization tools for integrated decision-making. Despite these efforts, the existing techniques still exhibit limitations, e.g., convergence issues in continuous reformulations and Lagrangean decomposition, or accuracy issues with simple linearization-based MIP reformulations. Thus, despite the advances in the field, more research is still needed in the development of MINLP algorithms that can efficiently handle

integrated decision-making problems. This section provides a brief overview of the optimization tools that support the MINLP algorithms developed through this thesis. Each subsection refers the reader to relevant works that provide further background on each topic.

2.4.1 Logic-based optimization of GDP problems

Generalized Disjunctive Programming (GDP) modeling and logic-based solution strategies were proposed to address some limitation of MINLP techniques, i.e., dependence on a predefined algebraic formulation such as big-M, and numerical issues encountered due to the interactions of continuous and discrete decisions[120]. GDP provides a logical modeling framework, where switching decisions between groups of constraints are represented through Boolean variables that activate/deactivate constraints within disjunction terms[13]. Logic-based solution approaches take advantage of the GDP form of the problem and its underlying logic relationships. Logic-based solvers that are freely available through the recently developed Pyomo.GDP plataform[13] include Logic-Based Outer Approximation (L-OA), Global Logic-based Outer Approximation (GL-OA), and Logic-Based Branch and Bound (L-BB). The defining characteristic of these algorithms is that they solve the nonlinear subproblems that result from fixing Boolean variables in a reduced space, i.e., nonlinear constraints and variables related to inactive disjunctions are omitted from the subproblem model. This avoids numerical issues and reduces subproblems' sizes when compared to MINLP solvers.

Logic-based algorithms such as L-OA, L-BB and GL-OA have not been extensively teste in the literature, given that they only became available open-source very recently. Also, logic-based strategies may still exhibit some of the limitations observed in MINLP algorithms. For instance, the quality of the solution obtained may depend on the user-specified initialization when a local solver such as L-OA is considered, while global solvers may lead to high computational times to reach global optimality. Despite the recent advances in GDP and logic-based optimization strategies, methodologies that can take advantage of problems involving special structures (e.g., ordered Boolean variables) that may be present in Boolean variables have not been widely studied. Furthermore, to the author's knowledge, logic-based solvers have not been studied in the field of integrated decision-making. The reader is referred to review[121] and research work[13], which provide further background in the area.

2.4.2 The Discrete Steepest Descent Algorithm

Ordered discrete decisions may appear in MINLP or GDP problems either as integer variables that appear explicitly in the formulation, or as groups of binary/Boolean variables that follow an ordered structure. For instance, the total number of times a reaction task is executed within a fixed scheduling horizon, or the number of reactors in series or parallel in a superstructure for process design are ordered discrete decisions. These special variables have only been studied in the context of optimal process design[4]. It has been shown that general-purpose MINLP solvers are unable to effectively capture these ordered structures in their optimization procedures, leading to suboptimal solutions and lengthy computational times. To overcome this limitation, the D-SDA was developed as a specialized decomposition strategy to effectively optimize ordered discrete decisions in MINLP superstructure design problems[4]. From a theoretical perspective, the main difference between the D-SDA and general purpose MINLP/GDP solvers is that D-SDA does not rely on traditional optimality and convexity requirements for ordered discrete decisions, e.g., a MINLP is "convex" if the relaxed-MINLP is convex. Instead, D-SDA has its roots on discrete convex analysis, which provides alternative definitions of convexity for Integer Programming (IP) and assesses the local optimality of ordered discrete decisions through neighborhood verification. A solution is declared as locally optimal when the subproblem's objective function does not improve within a neighborhood of the external variables. This feature has allowed the D-SDA to report better solutions than those obtained by local solvers such as SBB or DICOPT, when applied to optimal superstructure design[4].

The D-SDA was originally proposed for IP problems[6], and was latter adapted to solve MINLPs [4]. This strategy decomposes the problem in a master problem that optimizes ordered discrete variables and a subproblem that handles the remaining variables using an NLP solver. The value of the *external variables* in the *master problem* is updated using an algorithm that is analogous to the Steepest Descent Algorithm for continuous optimization. Thus far, using the D-SDA for the solution of MINLP problems that involve *external variables* has shown advantages in terms of solution quality and computational time, with respect of traditional MINLP algorithms [4,122]. The D-SDA has been tested in the context of PI, allowing the optimization of CD columns using equilibrium models[3,7].

In some cases, the local optimum found by the D-SDA may be a global optimum, even if traditional convexity assumptions do not hold[4]. For instance, this situation occurs when the objective function of the problem is *integrally convex*[6]. In this situation, a local optimization algorithm such as D-SDA avoids the need to have educated initial guesses for discrete decisions and decreases the computational effort

needed by standard MINLP optimization algorithms. The D-SDA is currently the only MINLP algorithm based on DCA. In the course of my PhD studies, the D-SDA has also been extended to directly solve GDP problems, resulting in the logic-based D-SDA (LD-SDA) variant[3]. However, the application of DCA-based techniques in other decision layers (e.g., scheduling), integrated decision-making, or optimal flowsheet design involving multiple interconnected units remains an open research area. Thus, further research is needed to understand the benefits of DCA-based algorithms in the broader field of chemical engineering optimization. The reader is referred to book[6] for background on DCA, and to research works[4] for the application of D-SDA to MINLP problems. The extension of D-SDA to GDP (LD-SDA) can be found in[3,123]. Given the relevance of DCA in this thesis, the next section shows important definitions retrieved from[6] that will be used through this thesis.

2.4.3 Discrete Convex Analysis (DCA)

DCA aims to propose convexity notions for the optimization of an integer-valued functions, where the vector if integer variables is denoted by $\boldsymbol{e}$. To get started, the *effective domain* of a function f with integer arguments $\boldsymbol{e}$ is shown in definition 2.1. This notion is relevant, as the effective domain defines the subset of the n-dimensional integer-lattice ($\mathbb{Z}^n$) of interest, where potential integer solutions are finite, i.e., neither unbounded ($f \neq -\infty$) nor infeasible ($f = \infty$).

Definition 2.1 The *effective domain* of function $f: \mathbb{Z}^n \to \mathbb{R} \cup \{\pm\infty\}$ is given by:

$$\mathrm{dom}_{\mathbb{Z}} f = \{\boldsymbol{e} \in \mathbb{Z}^n : -\infty < f(\boldsymbol{e}) < +\infty\} \tag{2.1}$$

Henceforth, assume that $f: \mathbb{Z}^n \mapsto \mathbb{R} \cup \{+\infty\}$ and that $\mathrm{dom}_{\mathbb{Z}} f$ is nonempty. Now, it is possible to define the *convex closure* of f and the *local convex extension of* f, through definitions 2.2-2.4. These definitions allow verifying if a function f is *integrally convex* (definition 2.5), which is a notion that allows defining stopping conditions in DCA-based algorithms such as the D-SDA, as well as providing the requirements needed to guarantee global optimality from the local search performed by the D-SDA. Functions f, $\bar{f}$ and $\tilde{f}$ satisfy the following properties: 1) $\tilde{f}(\boldsymbol{x}) \geq \bar{f}(\boldsymbol{x}), \forall \boldsymbol{x} \in \mathbb{R}^n$ and 2) $\tilde{f}(\boldsymbol{e}) = f(\boldsymbol{e}), \forall \boldsymbol{e} \in \mathbb{Z}^n$. Note that the equality $\bar{f}(\boldsymbol{e}) = f(\boldsymbol{e}), \boldsymbol{e} \in \mathbb{Z}^n$ is not always true.

Definition 2.2 The *convex closure* of f is a function $\bar{f}: \mathbb{R}^n \mapsto \mathbb{R} \cup \{\pm\infty\}$ given by:

$$\bar{f}(\boldsymbol{x}) = \sup_{\boldsymbol{p} \in \mathbb{R}^n, \alpha \in \mathbb{R}} \{\langle \boldsymbol{p}, \boldsymbol{x} \rangle + \alpha : \langle \boldsymbol{p}, \boldsymbol{e} \rangle + \alpha \leq f(\boldsymbol{e}), \forall \boldsymbol{e} \in \mathbb{Z}^n\}, \quad \boldsymbol{x} \in \mathbb{R}^n \tag{2.2}$$

Definition 2.3 The *integral neighborhood* of a point $\boldsymbol{x} \in \mathbb{R}^n$ is defined by:

$$N(\boldsymbol{x}) = \{\boldsymbol{e} \in \mathbb{Z}^n : \|\boldsymbol{x} - \boldsymbol{e}\|_\infty < 1\}, \qquad \boldsymbol{x} \in \mathbb{R}^n \tag{2.3}$$

Definition 2.4 The *local convex extension* of f is a function $\tilde{f}: \mathbb{R}^n \mapsto \mathbb{R} \cup \{\pm\infty\}$ given by:

$$\tilde{f}(\boldsymbol{x}) = \sup_{\boldsymbol{p} \in \mathbb{R}^n, \alpha \in \mathbb{R}} \{\langle \boldsymbol{p}, \boldsymbol{x} \rangle + \alpha : \langle \boldsymbol{p}, \boldsymbol{e} \rangle + \alpha \leq f(\boldsymbol{e}), \forall \boldsymbol{e} \in N(\boldsymbol{x})\}, \quad \boldsymbol{x} \in \mathbb{R}^n \tag{2.4}$$

Definition 2.5 A function $f: \mathbb{Z}^n \mapsto \mathbb{R} \cup \{+\infty\}$ is said to be *integrally convex* if and only if $\bar{f}(\boldsymbol{x}) = \tilde{f}(\boldsymbol{x})$, $\forall\, \boldsymbol{x} \in \mathbb{R}^n$.

Definition 2.5 is relevant for this thesis, as it is able to provide a definition of convexity for the optimization of (generally nonlinear) integer-valued functions. In contrast to the standard convexity notion for MINLPs (i.e., the continuous relaxation of the MINLP is convex), integral convexity does not rely on the relaxation of the integer requirement of discrete variables $\boldsymbol{e}$. A function may still retain integrally convex properties in a case where its continuous relaxation shows a highly non-convex (according to the usual definition of convexity) and multimodal shape. Accordingly, an optimization algorithm such as the D-SDA that relies on the integrally convex notion may avoid the nonlinearities that arise when traditional local optimization methods (e.g., DICOPT) relax integer requirements within their iterative procedures, and ultimately converge to a different (and potentially better) solution.

It is noteworthy that definition 2.5 does not explicitly impose any convexity requirement over the search space of interest for integer variables $\boldsymbol{e}$. This requirement is rather implicit in definition 2.5, in the sense that a function cannot be integrally convex if its effective domain (i.e., $\mathrm{dom}_{\mathbb{Z}} f$) is not integrally convex. The definition of an integrally convex effective domain is provided below (definition 2.6), where $\mathrm{co}(\mathrm{dom}_{\mathbb{Z}} f)$ is the convex hull of $\mathrm{dom}_{\mathbb{Z}} f$.

Definition 2.6 Consider the effective domain $\mathrm{dom}_{\mathbb{Z}} f$ of function $f: \mathbb{Z}^n \mapsto \mathbb{R} \cup \{+\infty\}$. If for any $\boldsymbol{x} \in \mathbb{R}^n$ having $\boldsymbol{x} \in \mathrm{co}(\mathrm{dom}_{\mathbb{Z}} f)$ implies that $\boldsymbol{x} \in \mathrm{co}\big(\mathrm{dom}_{\mathbb{Z}} f \cap N(\boldsymbol{x})\big)$, then $\mathrm{dom}_{\mathbb{Z}} f$ is integrally convex.

The definitions described above are key because they allow to define local and global optimality conditions for the optimization of integer-valued objective functions. To the author's knowledge, these optimality conditions are not implemented by existing local deterministic MINLP methods. To test local optimality, DCA theory proposes evaluating function f within an integer neighborhood $N_\infty(\boldsymbol{e}^*)$, where $\boldsymbol{e}^*$ is the local minimizer candidate and $N_\infty(\boldsymbol{e}^*)$ contains those discrete points whose ℓ_∞ -norm

(infinite norm) with respect to $\boldsymbol{e}^*$ is less or equal to 1. Based on this neighborhood, theorem 2.1 provides local and global optimality definitions.

Theorem 2.1 *For* $f: \mathbb{Z}^n \mapsto \mathbb{R} \cup \{+\infty\}$ *and* $\boldsymbol{e}^* \in dom_{\mathbb{Z}} f$*, if*

$$f(\boldsymbol{e}^*) \leq f(\boldsymbol{e}), \quad \forall \boldsymbol{e} \in N_\infty(\boldsymbol{e}^*), \text{ with } N_\infty(\boldsymbol{e}^*) = \{\boldsymbol{e} \in \mathbb{Z}^n : \|\boldsymbol{e} - \boldsymbol{e}^*\|_\infty \leq 1\} \tag{2.5}$$

then, $\boldsymbol{e}^*$ *is a local (or i-local) minimizer. If* f *is integrally convex, the following statement is true:*

$$f(\boldsymbol{e}^*) \leq f(\boldsymbol{e}), \forall \boldsymbol{e} \in \mathbb{Z}^n \Leftrightarrow f(\boldsymbol{e}^*) \leq f(\boldsymbol{e}), \quad \forall \boldsymbol{e} \in N_\infty(\boldsymbol{e}^*) \tag{2.6}$$

meaning that $\boldsymbol{e}^*$ *is the global minimum of* f.

Note that DCA theory goes beyond integrally convex functions, and defines other types of convexity, e.g., separable convexity, M-convexity, L-convexity, among others. For each of the existing convexity definitions, there is an associated exploration neighborhood $N_\Theta(\boldsymbol{e}^*)$ for which local optimality with respect to the neighborhood implies global optimality. For example, through this thesis some problems are solved using the $N_2(\boldsymbol{e}^*)$ exploration neighborhood associated to separable convexity, which contains those discrete points whose ℓ_2 -norm (Euclidean norm) with respect to $\boldsymbol{e}^*$ is less or equal to 1. Nonetheless, most of the existing convexity classes (, e.g., separable convexity, M-convexity, L-convexity, etc) are subclasses of the broader integrally convex notion, which has been widely studied in the literature. Therefore, the integrally convexity notions were adopted as the DCA foundation for the contributions presented through this thesis. Note that the extension of the aforementioned theory to MINLP optimization involving ordered discrete decisions is provided in a previous work[4], which relies on the decomposition of the problem into integer variables $\boldsymbol{e}$ in a master problem and objective function evaluations obtained through subproblem optimization, i.e., rather than having an analytic function, $f(\boldsymbol{e})$ becomes a black-box function when dealing with MINLP optimization.

2.4.4 Logic Based Benders Decomposition (LBBD)

One promising and widely used solution method to exploit special features in optimization formulations is Logic-Based Benders Decomposition (LBBD), which iterates between a Mixed-Integer Linear Programming (MIP) master problem that solves for a subset of complicating variables, and a subproblem (typically a linear optimization problem) which keeps complicating variables fixed[10]. The solution of each subproblem returns a Benders cut that is added as a new constraint in the master problem. LBBD provides an algorithmic framework that generalizes the Benders Decomposition (BD)

method allowing the user to design customized logic-based benders cuts that can be used to exploit special problem structures[124]. Also, subproblems do not need to be of a specific type in LBBD as long as they allow the generation of new Benders cuts[125].

Most applications of LBBD can be found in the fields of scheduling, planning and their integration[10,124–127]. Their results show that customized LBBD strategies generally outperform state-of-the-art solvers. Juvin et al.[125] showed that LBBD results in better optimality gaps and objective function values when compared to constraint programming techniques in a job-shop scheduling problem. Similarly, Elçi and Hooker[127] studied the first LBBD application in two-stage stochastic planning and scheduling, which resulted in tighter cuts and faster convergence than the conventional L-shaped BD method. Despite this progress, the application of LBBD in other integrated decision-making applications has not been reported in the literature. To the authors' knowledge, the Benders-based solution strategies proposed in the literature for MINLP integrated decision-making have been developed under the traditional Generalized Benders Decomposition (GBD) framework. The reader is referred to reference [128] for further details on LBBD and its applications.

2.4.5 Hybrid deterministic-stochastic MINLP optimization

A key benefit of stochastic (i.e., metaheuristic) optimization algorithms compared to deterministic algorithms is that they can optimize black-box models, such as those emerging from chemical engineering simulations for design optimization. Nevertheless, stochastic methods require user expertise to determine tunning parameters, do not provide optimality guarantees, and tend to require lengthy computational times. Previous works have suggested different arrangements of hybrid deterministic-stochastic MINLP algorithms that counterbalance the benefits and drawbacks of each strategy. The simplest hybrid deterministic-stochastic technique consists of applying these techniques sequentially, where stochastic optimization is used first, and the best solution found is further refined using a deterministic optimization solver, see e.g., Srinivas and Rangaiah [129], Munawar and Gudi [130], Staudt and de P. Soares [131], Chia et al. [132], and Herrera Velázquez et al.[133]. Given that there is no iterative interaction between the deterministic and the stochastic step, the performance of neither of the algorithms improves. Moreover, this sequential methodology would fail at providing local optimality guarantees in flowsheet optimization problems implemented within a commercial simulator without MINLP capabilities. More advanced strategies hybridize stochastic and deterministic strategies in a nested fashion, by keeping the stochastic method in an outer loop and the deterministic strategy in an

inner loop, see e.g., Urselmann et al. [134–136], Zhou et al. [137], Gómez et al. [138], and Holtbruegge et al.[139]. Most of those works implement a memetic algorithm, i.e., an evolutionary algorithm that optimizes the discrete (or a mix of discrete and continuous) degrees of freedom (DOF) in an outer loop, while NLP optimization with discrete variables fixed is used for each solution candidate in an inner loop. Although these memetic algorithms have demonstrated improvements in solution quality and computational performance, they fail to guarantee local optimality for discrete DOF in the inner loop. To address this issue, Kruber et al. [38] and Skiborowski et al. [33] used GAMS to investigate the inclusion of both discrete and continuous DOF when evaluating solution candidates in the inner loop. To avoid the rigorous solution of an MINLP for each solution candidate, Skiborowski et al. [33] recommend a continuous reformulation of discrete variables such as feed locations and number of stages, which approximates the solution of the MINLP as a series of NLPs.

Considering a simulation software to define the flowsheet design optimization problem exhibits a few attractive features. For instance, they do not require the explicit formulation of the model equations and they allow to modify the underlying property methods, e.g., test different thermodynamic packages. Also, models embedded in chemical engineering software can be applied to a wide range of flowsheets with multiple unit operations and DOF while preserving the robustness of the models, thanks to their readily available unit operation, thermodynamic, and physical property models. Nevertheless, a practical limitation of a simulation tool is the lack of integration between deterministic mathematical programming techniques and process simulators. Chemical engineering simulation software are typically able to perform local NLP optimizations; nonetheless, most simulation tools do not have MINLP optimization capabilities [37,140,141]. This hinders the application of hybrid algorithms that rely on MINLP optimization to flowsheet design optimization with simulation software. For instance, the nested hybrid algorithm proposed by Skiborowski et al.[33] relies on the continuous reformulation of a MINLP problem; however, traditional simulation tools do not allow the continuous reformulation of discrete decisions within their corresponding NLP optimizer. Based on the above, a hybrid deterministic-stochastic optimization tool for simulation-based MINLP optimal design that provides an optimality certificate and addresses the computational time issues of traditional stochastic optimization has not been proposed in the literature. The reader is referred to [59] for further details on stochastic and hybrid methods for MINLP.

2.5 Summary

The first two sections of this chapter introduced the type of problems that will be considered through this thesis, namely optimal design problems, optimal design and control problems and simultaneous scheduling and control problems. The second section provided a literature review that covers the existing deterministic MINLP optimization algorithms, focusing on those that are relevant for optimal process integration in chemical engineering. The third section summarized the optimization tools that will be used through this thesis to propose alternative MINLP optimization algorithms. Based on the literature review presented through this section, the process systems engineering community (PSE) has been transitioning from the direct execution of general-purpose solvers (e.g., DICOPT, SBB) to the development of ad-hoc optimization tools able to efficiently solve increasingly complex MINLP problems (e.g., LD, NLP reformulations). In this context, there are still some problem features and special structures that have not been widely studied and exploited in the literature, particularly in the field of optimal process integration.

GDP has been widely accepted as a modeling framework more general than MINLP; however, optimization algorithms that are able to solve GDP problems directly (i.e., without a transformation such as Big-M) are still limited and have not been widely tested and studied. To the author's knowledge, direct GDP solvers have not been studied in the field of integrated decision-making. Furthermore, a common feature of GDP and MINLP problems in chemical engineering is the presence of Integer/binary/Boolean variables that follow ordered structures, and that have been traditionally optimized using search techniques such as OA or BB. Recently, an MINLP algorithm based on DCA known as the D-SDA has been proposed to search the space of these variables more efficiently; nonetheless, the solution of MINLP problems using DCA concepts is still an emerging research field. Thus far, the D-SDA has only been tested to tackle the optimal design of single-unit problems. Hence, studying the performance of the D-SDA to tackle a wider variety of MINLP problems in chemical engineering such as those found in optimal process integration is still an open research area. Moreover, the D-SDA is the only DCA-based optimization algorithm available in the literature to date. Some optimization tools that can be used to propose and extend the applicability of DCA-based algorithms include LBBD and the hybridization of deterministic and stochastic optimization techniques. On the one hand, LBBD is a tool that enables the development of Benders-based algorithms, by exploiting special problem structures for cut generation. Accordingly, DCA concepts can be incorporated into the generation of these cuts, which is an idea that has not been previously explored in the literature. In the

context of integrated decision-making, LBBD has shown improvements with respect to general purpose MIP solvers and traditional BD techniques in scheduling and planning, but the application of LBBD in MINLP integrated decision-making is still an open research area. On the other hand, hybridizing stochastic techniques with deterministic DCA-based methos brings an opportunity to improve the performance of purely stochastic optimization techniques, which are often used in simulation-based flowsheet optimization. To the author's knowledge, a hybrid deterministic-stochastic algorithm that can guarantee local optimality for both discrete and continuous variables is not currently available to optimize a process design using chemical engineering simulation software.

Chapter 3
Optimal Synthesis and Design of Catalytic Distillation Columns: A Rate-Based Modeling Approach

This chapter aims to perform the optimal synthesis and design of staged CD columns modeled with rate-based and hydrodynamic equations that take into account the complex multiscale and multiphase behavior of this intensified process. This optimization problem can be summarized as follows: find the optimal operating variables (e.g. reflux ratio and reboiler duty), continuous design variables (e.g. diameter, stage height and weir height), and discrete design decisions (e.g. number of stages, location of feeds and distribution of reactive stages) that minimize an economic objective function for a CD column. The optimization of the objective function is subject to microscale constraints related to multicomponent mass/energy transfer, reaction kinetics and interphase equilibrium, as well as macroscale constraints associated with mass/energy balances, pressure drop, hydrodynamic inequality constraints, and a feasible region for discrete variables. The direct solution of such intensive optimization problems is challenging due to nonlinearities introduced by heterogeneous reactions, transport phenomena, and interactions between discrete and continuous variables. Also, combinatorial complexity involved in the selection of multiple structural decisions complicates the problem, e.g., distribution of reactive stages along the column, location of feed stages, and total number of stages. Thus, the discrete-steepest descent algorithm (D-SDA) is used to address the optimal synthesis and design of rate-based CD columns. To the author's knowledge, MINLP deterministic optimization techniques such as the D-SDA have not been studied in the past to address the optimal design of CD modeled with rate-based equations.

This chapter is organized as follows: Section 3.1 provides an overview of PI and the optimal design of CD columns. In Section 3.2, nonequilibrium models are introduced, emphasizing their multiscale nature and their differences and similarities compared to the equilibrium MESH (Mass, Equilibrium, Summation, entHalpy) model. Next, a set of MINLP equations that combine the selected rate-based model, macroscale phenomena, and a discrete superstructure is presented in Section 3.3. Section 3.4 summarizes the D-SDA methodology. Section 3.5 presents the case study used in this work to test the rate-based CD model's performance. Two computational experiments are presented in this section to motivate the synthesis of CD columns with rate-based models. Also, the ETBE CD column's optimal design is performed with the D-SDA to consider discrete variables and their interaction with the

multiscale model. Furthermore, general-purpose MINLP solvers are compared with the D-SDA at the end of this section.

3.1 Overview of PI and Optimal Design of CD Columns

Process intensification (PI) is a rapidly evolving area that is gaining considerable attention to develop new efficient, environmentally friendly, and sustainable chemical operations that can produce valuable products at a low cost. The current trend in chemical engineering is to investigate and use the multiscale structure of processes and products to further intensify and optimize chemical systems [142–148]. Although the combination of multiscale considerations with PI may result in higher computational requirements, recent developments in modeling tools and algorithms make these combined problems tractable for optimization. For instance, multiscale optimization approaches have been successfully applied to the intensification of processes such as milli-scaled fixed bed reactors [149,150] as well as the implementation of radial microchannel reactors for the multiscale intensification of gas-to-liquid technologies in the oil industry [151]. Multiscale frameworks have also been applied to other processes such as hydraulic fracturing [152]; crystallization [153,154]; kraft pulping [155]; lithium-ion batteries [156,157]; thin film deposition [158]; polymerization [159]; and catalytic surface reactions [160–164]. These studies are motivated by the fact that coupling fine-scale events with macroscale events is critical to capture the actual behavior of systems that evolve at multiple spatial and temporal scales [154,163,165–168]. The development of multiscale models is not an easy task since it may require special mathematical tools that would enable accurate prediction of the system's observables. For instance, tailored coupling and model reduction techniques may be needed to efficiently transfer information between different time domains and across different spatial dimensions. In general, multiscale models are complex; hence, low-order models are developed to capture the process behavior under various operating conditions and then used to perform control and optimization for these systems [169–171]. A recent trend in this field is to develop hybrid models that couple mechanistic multiscale models with low-order models, which replace a section of the actual model that can be challenging to compute [155,159,172–176].

In the context of the distillation and reaction systems, PI is accomplished by efficiently integrating processes into a single operation; this is referred to as a reactive distillation (RD) or catalytic distillation (CD)[c] processes. PI alternatives that have recently been investigated in this field include columns with

[c] RD is an intensified process that combines both reaction and separation into a single unit. CD is a particular case of RD where a heterogeneous catalyst is used to accelerate the reactions taking place within the reactive column.

multiple reactive zones, reactive dividing wall columns, and catalytic cyclic distillation [145,177–181]; and additional applications are provided in [182]. Despite the progress in this area, PI solutions for RD/CD are still emerging, but multiscale interactions are seldom considered, hence the need to advance this field.

A catalytic distillation (CD) column is a PI unit that combines heterogeneous catalysis and multiphase separation for the cost-effective production of valuable chemical products. Despite the significant progress in the optimal synthesis and design of CD columns, most of the available optimization studies for this PI unit assume phase/thermal equilibrium [183–185]. This assumption neglects complex micro and nanoscale heat and mass transfer events taking place in the vapor-liquid interphase within the column, which will impact the unit's overall performance. The rate-based model, also referred to as the MERSHQ model (Mass, Energy, Rate, Summation, Hydraulic, and interface eQuilibrium), allows one to perform distillation calculations without the assumption that the vapor and liquid bulks are in equilibrium[186]. Thus, this modeling approach accounts for the mass and energy transfer rates from the bulks to their interphase, where thermodynamic equilibrium is assumed. To account for these microscale phenomena, two approximations are usually considered: (1) a differential approximation, where partial differential equations are directly used to perform mass and energy balances over finite elements; and (2) an integral approximation that usually results in a set of algebraic equations, with products of transfer coefficients and driving forces. These characteristics suggest that the rate-based model predicts a real column's behavior with better accuracy than the traditional equilibrium models, which have been discussed in detail in previous reports. For instance, Taylor et al. [187] showed that the rate-based model is better than the equilibrium model at predicting composition trajectories for multicomponent systems such as water-ethanol-acetone and water-cyclohexane-ethanol. Furthermore, it has been previously reported that an equilibrium model may not provide accurate descriptions of the actual system’s behavior. Hence, rate-based models are more suitable to specify reliable CD designs [188]. Similar observations have also been made for CD and RD units [189–199]. Some of these works also highlighted the need to use comprehensive nonequilibrium models that incorporates both continuous and discrete variables to intensify both laboratory and industrial-scale units[195,196]. Hence, nonequilibrium models are essential to specify CD designs that remain feasible during operation.

Optimization-based process synthesis and design of CD columns with nonequilibrium models are required to specify realistic optimal process designs. Optimization procedures seek to replace heuristics, step-by-step procedures, and overdesign factors by automatic algorithms that rely on the objective

function and the superstructure model provided by the user[200]. These optimization models are often posed as Mixed-Integer Nonlinear Programming (MINLP) problems, which are challenging to solve given the presence of discrete variables and nonlinear equations. The first obstacle posed by MINLP models for CD synthesis and design is that they introduce multiple nonlinear equations required to specify multi-component diffusion, thermodynamic, hydrodynamic, and reaction models. Moreover, using a rate-based model increases the size of the problem when compared to an equilibrium model. For example, if design variables are ignored, and both equilibrium and rate-based models are expressed in terms of their input and state variables, then the rate-based model has approximately two times the number of variables of the equilibrium model [201]. Hence, both the model complexity and the corresponding computational costs increase while solving those problems. Accordingly, the selection of an adequate nonlinear solver is critical to guarantee convergence. The second obstacle is the introduction of discrete decisions such as the total number of stages, feeds location, and reaction stages location, which add combinatorial complexity. The interaction between discrete and continuous variables is another major obstacle that makes the optimal synthesis and design of CD columns a non-trivial task. The issue with these discrete-continuous interactions is that both convergence and optimality may be compromised if an adequate MINLP formulation is not selected, e.g., variables being forced to zero by the discrete choices can lead to numerically undefined nonlinear expressions.

Optimization studies for the optimal design of rate-based CD units are still lacking in the literature. The optimal synthesis and design of nonequilibrium RD/CD columns have been previously studied by Gómez et al.[201]. A decomposition strategy was used in that work, which resulted in a *master* MINLP model solved using simulated annealing and nonlinear subproblems solved with successive quadratic programming. Another study addressed the rate-based design of a CD column for the synthesis of tert-amyl ethyl ether by combining a heuristic procedure with a modified differential evolution algorithm[202]. More recently, the stochastic approach has been extended to intensified distillation sequences by linking a genetic algorithm to the simulator, such as Aspen Plus[203]. In contrast to stochastic techniques, there is a gap in the literature regarding the deterministic[d] optimal design of heterogeneously catalyzed or reactive distillation columns using rate-based models. Deterministic optimization techniques provide optimality guarantees, yield replicable results, and do not require system-dependent parameters to be tuned[204]. To the authors' knowledge, the only relevant studies that used deterministic optimization for

[d] Deterministic means a solution method that does not rely on random number generation and would therefore yield the same answer if run twice under the same computational conditions.

optimal nonequilibrium CD design applied orthogonal collocation on finite elements, which leads to the reformulation of the original MINLP model into a large-scale continuous nonlinear problem[205,206]. The orthogonal collocation approximation may be a suitable representation of the original process if an adequate number of collocation points is considered. While orthogonal collocation is an alternative to the design of distillation processes, the relaxation of the discrete decisions is an approximation that inherently results in solutions that may not necessarily comply with the discrete nature of the corresponding decision variables[205,207]. Contrary to the studies reviewed above, the D-SDA explicitly considers discrete choices and deterministic procedures to optimize the design of nonequilibrium CD columns.

3.2 CD nonequilibrium model

Rate-based models fall into the broader category of nonequilibrium models, which do not assume that the multiple phases inside the internal stages of distillation columns are in thermodynamic equilibrium (see the available nonequilibrium models for CD in Figure 3.1). The most generic nonequilibrium model should consider non-thermal, non-mechanic, non-phase, and non-chemical equilibrium. This is possible through Computational Fluid Dynamics (CFD) simulations; however, this approximation is often found impractical for design optimization because it requires high CPU times and it is currently limited to process simulation of laboratory-scale columns or subsections of distillation units, e.g. a single stage of a column[208]. Another strategy that considers the nonequilibrium nature of the process consists of using hydrodynamic analogies obtained from experimental observations of flow patterns[208,209]. These analogies can be used to simplify the conservation differential equations that represent the process. Although this strategy has been widely used for various applications in reactor design[210–213], it has only been studied for a few CD packed columns. In contrast, the equilibrium MESH model has been traditionally used for CD, but it is limited to consider only non-chemical equilibrium[214]. Consequently, a tradeoff is obtained with rate-based models, making multiple assumptions that allow one to obtain simple though realistic representations of the non-thermal, non-phase, and non-chemical equilibrium behavior inside a column. The main assumption of rate-based models is that they do not consider momentum balances in their equations. Instead, simple flow patterns are defined *a priori*; in particular, rate-based models for mixed vapor-mixed liquid, plug flow vapor-mixed liquid, plug flow vapor-dispersion flow liquid, and plug flow vapor-plug flow liquid are available in the literature[215].

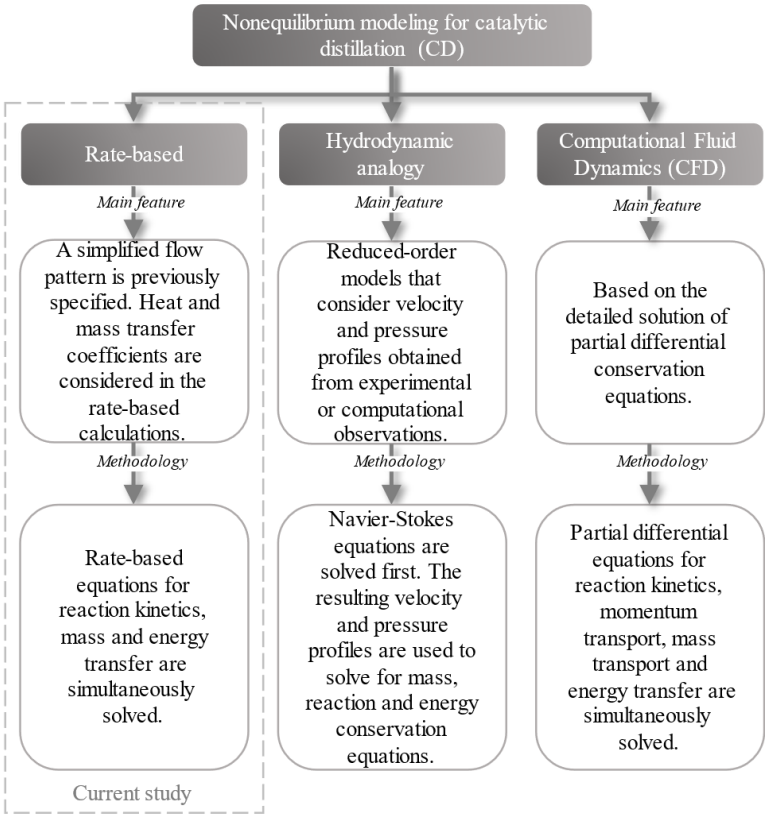


Figure 3.1. Nonequilibrium models for CD.

Contrary to equilibrium MESH models, rate-based models do not include multiphase balances in a single equation. Instead, each phase is modeled separately, and additional terms are included to account for molar ($\mathcal{N}_{n,i}$) and energy ($\mathcal{E}_n$) exchange between phases for every component i at separation stage n [186]. In the specific case of heterogeneous CD, length and time scales vary from nanometers to meters and from nanoseconds to hours, respectively. For instance, catalyst, film, and intraparticle diffusion occurs at small scales (nanometers-millimeters and milliseconds-minutes). In contrast, macroscale events associated with column dynamics, dimensions, and undesired hydrodynamic effects occur at larger scales (millimeters-meters and seconds-hours) [216]. Accordingly, another distinguishing feature of rate-based models is that they can be combined with both small and large temporal and spatial scale events, while the equilibrium MESH model cannot consider small events different from reaction kinetics and nonideal equilibrium interactions between components.

3.2.1 Microscale phenomena

The microscale phenomena considered in this chapter happen over the continuum domain; thus, random interactions in the molecular domain are ignored. Multiple modeling strategies are available to account for small scale interactions in rate-based models [186,215–218]. The most common strategy consists of using a film model, which assumes that mass and energy transfer rates are controlled by a resistance concentrated in a layer adjacent to the interphase [186]. When using the film model, mass transfer coefficients are often used to account for each film's diffusive mass transfer rate. When the system involves chemical reaction(s), extra considerations need to be made. If the reaction is heterogeneously catalyzed, the most general case occurs when: (1) simultaneous reaction and diffusion through catalyst pores is considered; (2) there is film diffusion from/to the liquid to/from the catalyst, from/to the liquid to/from the vapor-liquid interphase, and from/to the vapor to/from the vapor-liquid interphase; and (3) the flow pattern has not been previously established. A schematic representation of this general case is shown in Figure 3.2, where illustrative composition profiles ($x_{n,i}$ and $y_{n,i}$) and temperature profiles (T_n^L and T_n^V) are shown for component i at stage n.

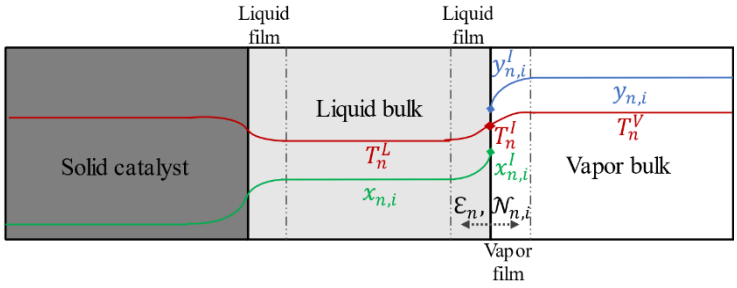


Figure 3.2. General representation of transport processes in CD. Thermodynamic equilibrium only occurs at the vapor-liquid interphase (♦), with compositions $x_{n,i}^I, y_{n,i}^I$ and temperature T_n^I.

Major assumptions are made in this study:

1. The reaction can be treated pseudo-homogeneously, i.e., there is no transfer resistance between the fluid and the catalyst particles, because the catalyst surface is completely exposed to the liquid bulk [219]. Therefore, diffusion through catalyst pores and catalyst-liquid film diffusion are lumped in a single term [216,220].

2. The mixed-flow pattern represents the velocity profile of the liquid and vapor phases inside the column. Equivalently, the vapor and liquid bulks are ideally mixed and the bulk properties may represent the outlet properties of the streams leaving the stages.

These assumptions simplify the steady-state rate-based model by reducing it to a nonlinear set of algebraic equations with a reduced number of variables, i.e. further reformulations to deal with Differential Algebraic Equations (DAEs) and complex operations over matrices like exponentials and square roots are avoided. For example, modeling of intraparticle diffusion would require the implementation of the differential dusty-fluid model or generalized Maxwell-Stefan differential equations; while dealing with plug flow or dispersion flow models involve the computation of exponentials, and square roots of matrices, which may impose computational penalties for systems with multiple components [186,192,215,221,222]. Although these assumptions are beneficial for simulation and optimization, they need to be verified for the system under consideration. For instance, CD processes are usually applied to the synthesis of octane-enhancing ethers like methyl tertiary butyl ether (MTBE), tertiary amyl methyl ether (TAME), and ETBE through etherification reactions over acid catalysts [223,224]. By comparing the dusty-fluid and pseudo-homogeneous models in a CD column, Higler et al.[192] concluded that mass transfer resistance shall not be ignored for MTBE production, while the production of TAME is kinetically controlled and intraparticle diffusion can be neglected. Those comparisons have not been performed yet for ETBE; however, studies for typical reaction conditions have shown that diffusion limitations may be absent or present, depending on parameters such as catalyst size, alcohol concentration, and temperature [221,222]. Regarding the flow pattern, assuming ideal mixing often underpredicts Murphree efficiencies since this simple flow model does not allow to obtain efficiencies greater than the unity, which are common in real practice with large diameter columns. On the other hand, plug flow and disperse flow models can predict efficiencies higher than the unity thus predicting the equilibrium model's deviation more accurately [215].

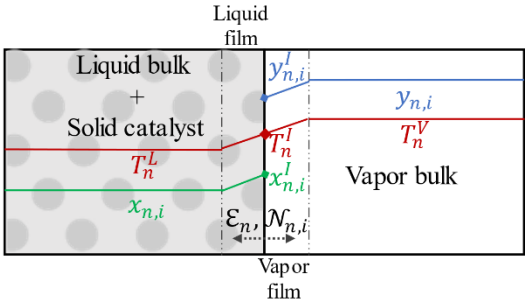


Figure 3.3. Representation of transport processes for component i at stage n in the selected rate-based model for CD.

A schematic representation of the rate-based model considered in this study is presented in Figure 3.3. In order to represent the multicomponent nature of the process, both the diffusive and convective contributions are included in Eqs. (3.1.A) and (3.1.B) to calculate the interphase flow $\mathcal{N}_{n,i}$ for component i at stage n, where $COMP$ is the set of components, c is a reference component, and $COMP_R$ is the set of all components except c. Diffusive contributions are calculated using film theory, which introduces multicomponent mass transfer coefficients between components i and m ($k_{n,i,m}^V$ and $k_{n,i,m}^L$) that must be estimated. As indicated by Taylor and Krishna [186], Maxwell-Stefan equations for mass transfer in multicomponent systems are presented in Eqs. (3.1.C)-(3.1.F). These equations can be used to calculate multicomponent coefficients when the film thickness is unknown. This results in a separation between diffusive contributions through terms $R_{n,i,m}^V$ and $R_{n,i,m}^L$, and thermodynamic contributions through thermodynamic factors $\Gamma_{n,i,m}^V$ and $\Gamma_{n,i,m}^L$. On the one hand, diffusive contributions depend on binary mass transfer coefficients $\kappa_{n,i,m}^V$ and $\kappa_{n,i,m}^L$, which are estimated from physical models or empirical correlations, using Maxwell-Stefan diffusion coefficients [217,225]. On the other hand, thermodynamic factors are needed because the fundamental driving force for mass transfer is the chemical potential [215]. For the vapor phase, thermodynamic factors are usually assumed to be ideal, i.e. $\Gamma_{n,i,m}^V = \delta_{n,i,m}$, where $\delta_{n,i,m}$ is the Kronecker delta at stage n. For the liquid phase, $\Gamma_{n,i,m}^L$ is calculated using activity coefficient models such as Margules, Van Laar, Wilson, Non-Random Two-Liquid (NRTL), etc. [226]. Regarding interfacial areas (a_n^I), these can be grouped with mass transfer coefficients on a single term, because binary mass transfer coefficient expressions are inversely proportional to the area, i.e., $\kappa_{n,i,m}^{Phase} \sim \frac{1}{a_n^I}, \forall Phase \in \{V, L\}$. Additionally, molar densities (c_n^V and c_n^L) are computed using appropriate mixing rules or equations of state [227]. For the interphase energy flow $\mathcal{E}_n$, the methodology

adopted in this study is based on thermodynamic calculations and it is described in Section 3.3.4.2. Note that flow, temperature and composition-dependent correlations for transfer and equilibrium coefficient calculations are not computed and fixed *a-priori*; instead, they are explicitly included within the optimization formulation.

With regards to the reaction kinetics, there are several models available for heterogeneously catalyzed reactions. Typically, these models consist of a series of elementary steps comprising adsorption, surface reaction, and desorption from the catalytic surface [228]. When applied to etherification reactions over acid catalytic surfaces, the most common models are the Langmuir-Hinshelwood-Hougen-Watson (LHHW) and Eley-Rideal type models [221,229]. In this study, the LHHW kinetic model shown in Eq. (3.1.G) is considered. At stage n, and for every reaction u in the set of reaction RX, this model defines a kinetic reaction rate $\mathcal{R}_{n,u}$ through: a *kinetic factor* that contains a rate constant and adsorption equilibrium constants multiplied by species activities raised to their corresponding exponents; a *driving force* that stablishes the chemical equilibrium limitation of the reaction, and an *adsorption group* representing the available number of active sites for catalytic reactions, with ns_u representing the number of sites in the rate determining step of reaction u [228].

Interphase flows: diffusive and convective contributions

$$\mathcal{N}_{n,i} = c_n^V \sum_{m\in COMP_R} a_n^I k_{n,i,m}^V \left(y_{n,m} - y_{n,m}^I\right) + y_{n,i} \sum_{i\in COMP} \mathcal{N}_{n,i} \ , \forall i \in COMP_R \tag{3.1.A}$$

$$\mathcal{N}_{n,i} = c_n^L \sum_{m\in COMP_R} a_n^I k_{n,i,m}^L \left(x_{n,m}^I - x_{n,m}\right) + x_{n,i} \sum_{i\in COMP} \mathcal{N}_{n,i} \ , \forall i \in COMP_R \tag{3.1.B}$$

Multicomponent mass transfer coefficients

$$\sum_{j\in COMP_R} k_{n,i,j}^V R_{n,j,m}^V = \Gamma_{n,i,m}^V \ , \forall i \in COMP_R, \forall m \in COMP_R \tag{3.1.C}$$

$$\sum_{j\in COMP_R} k_{n,i,j}^L R_{n,j,m}^L = \Gamma_{n,i,m}^L \ , \forall i \in COMP_R, \forall m \in COMP_R \tag{3.1.D}$$

$$R_{n,j,m}^V = \begin{cases} \frac{y_{n,j}}{\kappa_{n,j,c}^V} + \sum_{l\in COMP\setminus\{j\}} \frac{y_{n,l}}{\kappa_{n,j,l}^V}, j = m \\ -y_{n,j}\left(\frac{1}{\kappa_{n,j,m}^V} - \frac{1}{\kappa_{n,j,c}^V}\right), j \neq m \end{cases} , \forall i \in COMP_R, \forall m \in COMP_R \tag{3.1.E}$$

$$R^{L}_{n,j,m} = \begin{cases} \frac{x_{n,j}}{\kappa^{L}_{n,j,c}} + \sum_{l \in COMP \setminus \{j\}} \frac{x_{n,l}}{\kappa^{L}_{n,j,l}}, j = m \\ -x_{n,j}\left(\frac{1}{\kappa^{L}_{n,j,m}} - \frac{1}{\kappa^{L}_{n,j,c}}\right), j \neq m \end{cases}, \forall i \in COMP_R, \forall m \in COMP_R \quad (3.1.F)$$

LHHW reaction model

$$\mathcal{R}_{n,u} = \frac{(Kinetic\ factor_{n,u})(Driving\ force_{n,u})}{(Adsorption\ group_{n,u})^{ns_u}}, \forall u \in RX \quad (3.1.G)$$

3.2.2 Macroscale phenomena

Macroscale variables such as pressure drop, vapor and liquid flowrates, and internal dimensions (e.g., stage height, column diameter, weir length, and weir height) need to be coupled with hydrodynamic operation constraints. Aside from geometric design and pressure drop calculations, these constraints allow us to keep the column's internal flows within their capability limits by using inequality constraints, i.e., these hydraulic phenomena limit the feasible search range of design and operating variables by limiting vapor and liquid flows over the column. These undesired phenomena can be avoided by considering the following constraints [230]:

- *Entrainment flooding*: Vapor momentum pushes the froth level to reach the stage above. This is eliminated by imposing a maximum vapor velocity limit through the stage and maximum liquid velocity through the downcomer.
- *Downflow flooding*: Occurs when the liquid stacks up in the downcomer and the liquid froth in the downcomer reaches the stage above. This can be prevented by minimizing the stage pressure drop, thus avoiding excessive liquid flows.
- *Weeping*: This happens when the vapor velocity through the holes is too low. A lower bound on the vapor velocity through the holes needs to be enforced to avoid this condition.
- *Catalyst flooding*: This occurs due to excessive pressure drop at high vapor flows, and thus, a maximum pressure drop due to catalyst must be considered.

The coupling between microscale and macroscale phenomena, including hydrodynamic inequality constraints, discrete decisions, mole balances, and energy balances, is presented in Section 3.3.4.2. Note that mesoscale phenomena associated with differential conservation equations of mass, momentum, chemical species, and energy were not considered in this work. Considering mesoscale

phenomena would require implementing the problem in software for CFD simulation; however, this would be quite challenging for process design optimization. To the authors' knowledge, the integration of macro-, meso-, and microscales is currently limited to simulation studies for reactive distillation; see, e.g.[231].

3.3 Mathematical formulation

The formulation that results from combining both micro and macroscale phenomena with macroscale discrete decision variables is presented here. The proposed model can consider continuous variables and discrete choices in CD design, e.g., total number of stages, location of previously specified feed streams, number of catalytic stages, and location of catalytic stages. Keeping in mind that our goal is to perform optimal synthesis and design of a rate-based CD model, the resulting mathematical formulation leads to a Mixed-Integer Nonlinear Programming (MINLP) model. The modeling assumptions considered in this work are:

- Steady-state operation.
- Adiabatic operation.
- Total condenser and partial reboiler.
- Pseudo-homogeneous reaction.
- Vapor-liquid contact is allowed in reactive stages.
- The mixed-flow model represents the cross-flow of vapor and liquid, i.e., ideal mixing.
- The film model is used to represent interphase diffusive mass transfer rates.
- One-dimensional transport normal to the interphase.
- Only single reflux and boil-up stages are allowed.
- Both vapor and liquid bulks' temperatures satisfy saturation constraints.

In this case, an MINLP with binary multiplication variables is selected since it allows a straightforward interpretation of the interactions between discrete and continuous variables. However, these constraints can be reformulated using other approaches, e.g., Generalized Disjunctive Programming (GDP), Big-*M* or hull reformulations [232–235]. The model presented in this work is an adaptation of that proposed in our previous work for equilibrium stage-by-stage models [12]; however, this work aims to solve a more challenging multiscale and multiphase system. Consequently, the model follows the same logic for discrete variables described in [12]. The proposed rate-based model is described next.

3.3.1 Sets and definitions

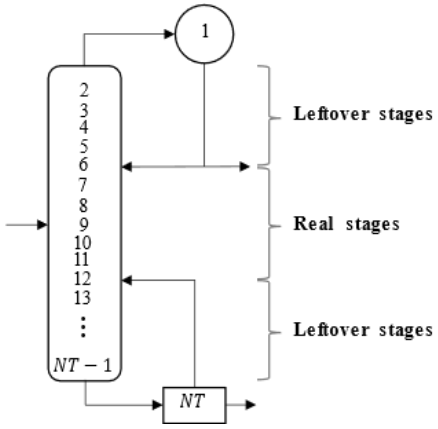


Figure 3.4. Schematic superstructure representation. $n = 1$ is the condenser and $n = NT$ is the reboiler.

The proposed MINLP model is a superstructure that contains all possible configurations of discrete decisions. Thus, a sufficiently large set of stages N is used, where some of these are *real stages* because they will be part of the resulting CD design. The remaining stages, i.e., *leftover stages*, are not considered part of the distillation column; however, it is necessary to impose constraints on these stages since they act as artificial bypasses that conserve mass and energy of single-phase flows (see Figure 3.4). Additional modeling descriptions include the set of chemical components ($COMP$), the set of feeds (F), the set of chemical reactions (RX), and those sets that define the three main sections of the superstructure, i.e., condenser ($COND$), reboiler (REB) and internal stages (INT). These sets are defined as follows:

Set of components (index i, j, l, m, c)

$$COMP = \{1, \dots, CT\} \tag{3.2.A}$$

Set of stages in the superstructure (index n, t)

$$N = \{1, \dots, NT\} \tag{3.2.B}$$

Set of feeds (index f)

$$F = \{1, \dots, FT\} \tag{3.2.C}$$

Set of reactions (index u)

$$RX = \{1, \dots, XT\} \quad (3.2.D)$$

Condenser stage

$$COND = \{1\} \subseteq N \quad (3.2.E)$$

Reboiler stage

$$REB = \{NT\} \subseteq N \quad (3.2.F)$$

Internal stages of the superstructure

$$INT = N \backslash REB \backslash COND \quad (3.2.G)$$

3.3.2 Structural decisions

This section defines independent and dependent binary variables used to explore the discrete decisions in distillation design. Four independent binary variables are considered in this study:

Existence of catalyst at stage n.

$$YC_n \in \{0,1\}, \forall n \in INT \quad (3.3.A)$$

Existence of reflux at stage n

$$YR_n \in \{0,1\}, \forall n \in INT \quad (3.3.B)$$

Existence of boil-up at stage n

$$YB_n \in \{0,1\}, \forall n \in INT \quad (3.3.C)$$

Existence of feed f at stage n

$$YF_{n,f} \in \{0,1\}, \forall n \in INT, \forall f \in F \quad (3.3.D)$$

Eq. (3.3.A) specifies if a stage is catalytic or not; Eq. (3.3.B) determines the reflux location, Eq. (3.3.C) defines the boil-up location (Eq. 3.3.C), whereas Eq. (3.3.D) establishes the location of feeds. If any of these binary variables is equal to 1 at stage n, it means that the defined process exists at that location.

Dependent binary variables can be calculated as functions of the independent binary variables. Dependent binary variables are used to decide whether a stage is a *real stage* or a *leftover stage* and establish when vapor-liquid contact occurs. A stage is a *real stage* if it is located below the reflux and above the boil-up, which is mathematically stated in Eq. (3.4.A), i.e. $YP_n = 1$ means that stage n is real. Regarding the contact between vapor and liquid phases, this study assumes that the catalyst is placed along the column so that vapor-liquid mass transfer exists in reactive stages [216]. This is enforced by Eq. (3.4.B), where $YE_n = 1$ indicates that there is vapor-liquid contact in stage n.

Definition of real and leftover stages n.

$$YP_n = 1 - \left(1 - \sum_{j=2}^{n} YR_j\right) - \left(\sum_{j=2}^{n} YB_j - YB_n\right), \forall n \in INT \quad (3.4.A)$$

Definition of mass transfer in real stages n

$$YE_n = YP_n, \forall n \in INT \quad (3.4.B)$$

3.3.3 Logical constraints

The constraints described in this section enforce structural requirements of superstructure distillation design. Eqs. (3.5.A)-(3.5.C) ensure countercurrent vapor-liquid contact by forcing one reflux and one boil-up to exist in the final design. Note that these equations allow the variable YP_n in Eq. (3.4.A) to take binary values only. Eq. (3.5.D) ensures single feeds, i.e., distributed feeds are not being considered. Eqs. (3.5.E)-(3.5.H) keep feed stages and catalytic stages inside the section of *real stages*.

The boil-up stage is below the reflux stage

$$\sum_{t=2}^{n} YR_t \geq YB_n \,, \forall n \in INT \quad (3.5.A)$$

There is one reflux stage

$$\sum_{n \in INT} YR_n = 1 \quad (3.5.B)$$

There is one boil-up stage

$$\sum_{n \in INT} YB_n = 1 \quad (3.5.C)$$

There is one feed stage per feed stream

$$\sum_{n \in INT} YF_{n,f} = 1, \forall f \in F \quad (3.5.D)$$

The feed stages are below the reflux stage

$$\sum_{t=2}^{n} YR_t \geq YF_{n,f}, \forall f \in F, \forall n \in INT \quad (3.5.E)$$

The boil-up stage is below the feed stages

$$\sum_{t=2}^{n} YF_{t,f} \geq YB_n, \forall f \in F, \forall n \in INT \quad (3.5.F)$$

The catalytic stages are below the reflux stage

$$\sum_{t=2}^{n} YR_t \geq YC_n, \forall n \in INT \quad (3.5.G)$$

The catalytic stages are above the boil-up stage

$$\sum_{t=2}^{n} YB_t - YB_n \leq 1 - YC_n, \forall n \in INT \quad (3.5.H)$$

3.3.4 Catalytic distillation rate-based model

The model herein represents the logic propositions of the problem by multiplying continuous variables with binary variables that turn on/off the constraints within each case, e.g., binary variables and constraints defined in the previous sections are combined with mole and energy balances. Both the mass and energy balance equations depend on the microscale modeling equations presented in section 3.2.1, thus enabling the coupling of macro- and microscale phenomena in the CD column. Condenser and reboiler equations are presented in Section 3.3.4.1. Then, the multiscale modeling events described in Section 3.2 are combined with binary variables, mole, and energy balances in Section 3.3.4.2 to complete the description of the mixed-integer rate-based CD model.

3.3.4.1 Condenser and reboiler

These equations are independent of binary variables because these stages hold a fixed position at stages 1 and NT for the condenser and reboiler, respectively. The notation for continuous variables is established as follows: V_n and L_n are molar vapor and liquid flows leaving stage n, respectively; $K_{n,i}^{I}$ is the interphase equilibrium constant for component i at n; Q_{Cond} and Q_{Reb} are condenser and reboiler duties, whereas H_n^V and H_n^L are vapor and liquid enthalpies at stage n, respectively. To avoid the presence of binary variables in condenser and reboiler equations, mole and energy balances must be expressed in terms of the reflux ratio (RR), the boil-up ratio (BR), and the output distillate (V_1) and bottoms (L_{NT}) liquid flows, i.e.

Total mole balance (condenser)

$$0 = V_2 - V_1(1 + RR) \quad (3.6.A)$$

Partial mole balance (condenser)

$$0 = V_2 y_{2,i} - V_1 x_{1,i}(RR + 1), \forall i \in COMP \quad (3.6.B)$$

Summation equation (condenser)

$$0 = \sum_{i \in COMP}\left(y_{1,i} - x_{1,i}\right) \quad (3.6.C)$$

Thermodynamic equilibrium (condenser)

$$0 = \left(y_{1,i}^I - K_{1,i}^I x_{1,i}^I\right), \forall i \in COMP \quad (3.6.D)$$

Energy balance (condenser)

$$0 = V_2 H_2^V - V_1(RR + 1)H_1^L - Q_{Cond} \quad (3.6.E)$$

Total mole balance (reboiler)

$$0 = L_{NT-1} - L_{NT}(1 + BR) \quad (3.6.F)$$

Partial mole balance (reboiler)

$$0 = L_{NT-1} x_{NT-1,i} - L_{NT}\left(x_{NT,i} + BR y_{NT,i}\right), \forall i \in COMP \quad (3.6.G)$$

Summation equation (reboiler)

$$0 = \sum_{i \in COMP}\left(y_{NT,i} - x_{NT,i}\right) \quad (3.6.H)$$

Thermodynamic equilibrium (reboiler)

$$0 = \left(y_{NT,i}^I - K_{NT,i}^I x_{NT,i}^I\right), \forall i \in COMP \quad (3.6.I)$$

Energy balance (reboiler)

$$0 = Q_{Reb} + L_{NT-1} H_{NT-1}^L - L_{NT} H_{NT}^L - (BR) L_{NT} H_{NT}^V \quad (3.6.J)$$

Moreover, both reboiler and condenser stages are modeled as equilibrium stages; thus, bulk compositions and temperatures are equal to those in the interphase, where thermodynamic equilibrium occurs [201], i.e.

Liquid bulk in thermodynamic equilibrium (condenser and reboiler)

$$0 = x_{n,i} - x_{n,i}^{I}, \forall n \in COND \cup REB, \forall i \in I \quad (3.7.A)$$

$$0 = T_n^L - T_n^I, \forall n \in COND \cup REB \quad (3.7.B)$$

Vapor bulk in thermodynamic equilibrium (condenser and reboiler)

$$0 = y_{n,i} - y_{n,i}^{I}, \forall n \in COND \cup REB, \forall i \in I \quad (3.7.C)$$

$$0 = T_n^V - T_n^I, \forall n \in COND \cup REB \quad (3.7.D)$$

Condenser and reboiler stages interact with the rest of the superstructure, i.e., internal stages, through Eqs. (3.8.A)-(3.8.D). These equations determine the location of the reflux liquid flow ($V_1(RR)$) and the boil-up vapor flow ($L_{NT}(BR)$).

Definition and location of reflux flow

$$LR_n = V_1(RR)YR_n, \forall n \in INT \quad (3.8.A)$$

$$L_1 = 0 \quad (3.8.B)$$

Definition and location of the boil-up flow

$$VB_n = L_{NT}(BR)YB_n, \forall n \in INT \quad (3.8.C)$$

$$V_{NT} = 0 \quad (3.8.D)$$

3.3.4.2 Internal stages

Internal stages determine if a stage is a *real stage* or a *leftover stage*. The constraints considered must include the presence/absence of reaction, feed flows, reflux flow, and boil-up flow for a *real stage*. Note that set $COMP_R$ stands for "every component except one" since some rate-based equations are independent for exactly $CT - 1$ components. The main rate-based equations that directly interact with the binary variables are as follows:

Partial model balance in the vapor phase

$$0 = \sum_{f\in F} YF_{n,f} IN_f^V z_{i,f}^V + VB_n y_{NT,i} + V_{n+1} y_{n+1,i} - V_n y_{n,i} - \mathcal{N}_{n,i}\,, \forall n \in INT, \quad \forall i \in COMP \tag{3.9.A}$$

Partial mole balance in the liquid phase

$$0 = \sum_{f\in F} YF_{n,f} IN_f^L z_{i,f}^L + LR_n x_{1,i} + L_{n-1} x_{n-1,i} - L_n x_{n,i} + \mathcal{N}_{n,i} + YC_n m_{cat,n} \sum_{u\in RX} \nu_{i,u} \mathcal{R}_{n,u}\,, \forall n \in INT, \forall i \in COMP \tag{3.9.B}$$

Summation equation in the vapor phase

$$0 = 1 - \sum_{i\in COMP} y_{n,i}\,, \forall n \in INT \tag{3.9.C}$$

Summation equation in the liquid phase

$$0 = 1 - \sum_{i\in COMP} x_{n,i}\,, \forall n \in INT \tag{3.9.D}$$

Energy balance in the vapor phase

$$0 = \sum_{f\in F} YF_{n,f} IN_f^V H_{IN,f}^V + VB_n H_{NT}^V + V_{n+1} H_{n+1}^V - V_n H_n^V - \mathcal{E}_n\,, \forall n \in INT \tag{3.9.E}$$

Energy balance in the liquid phase

$$0 = \sum_{f\in F} YF_{n,f} IN_f^L H_{IN,f}^L + LR_n H_1^L + L_{n-1} H_{n-1}^L - L_n H_n^L + \mathcal{E}_n\,, \forall n \in INT \tag{3.9.F}$$

Mass transfer mechanism

$$0 = (1 - YE_n)\mathcal{N}_{n,i}\,, \forall n \in INT, \forall i \in COMP \tag{3.9.G}$$

$$0 = YE_n\left(\mathcal{N}_{n,i} - c_n^V \sum_{m\in COMP_R} a_n^I k_{n,i,m}^V \left(y_{n,m} - y_{n,m}^I\right) - y_{n,i} \sum_{i\in COMP} \mathcal{N}_{n,i}\right), \quad \forall n \in INT, \forall i \in COMP_R \tag{3.9.H}$$

$$0 = YE_n\left(\mathcal{N}_{n,i} - c_n^L \sum_{m\in COMP_R} a_n^I k_{n,i,m}^L \left(x_{n,m}^I - x_{n,m}\right) - x_{n,i} \sum_{i\in COMP} \mathcal{N}_{n,i}\right), \quad \forall n \in INT, \forall i \in COMP_R \tag{3.9.I}$$

$$0 = YE_n\left(y_{n,i}^I - K_{n,i}^I x_{n,i}^I\right), \forall n \in INT\,, \forall i \in COMP \tag{3.9.J}$$

$$0 = YE_n\left(1 - \sum_{i\in COMP} y_{n,i}^I\right), \forall n \in INT \tag{3.9.K}$$

$$0 = YE_n\left(1 - \sum_{i \in COMP} x_{n,i}^I\right), \forall n \in INT \quad (3.9.L)$$

Energy transfer mechanism

$$0 = (1 - YE_n)\mathcal{E}_n, \forall n \in INT \quad (3.9.M)$$

$$0 = YE_n(T_n^V - T_n^{DEW}), \forall n \in INT \quad (3.9.N)$$

$$0 = YE_n(T_n^L - T_n^{BUBBLE}), \forall n \in INT \quad (3.9.O)$$

Pressure drop

$$P_n = (P_{n-1} + YP_n \Delta P_n), \forall n \in INT \quad (3.9.P)$$

Overall, Eqs. (3.9.A)-(3.9.P) are used to connect microscale diffusion and reaction interactions with macroscale mole and energy balances in vapor and liquid bulks and discrete variables, thus highlighting the multiscale nature of this CD column model. Eqs. (3.9.A)-(3.9.F) are mole and energy balances for vapor and liquid phases, which are interconnected through variables $\mathcal{N}_{n,i}$ and $\mathcal{E}_n$. Both variables can take positive, zero, or negative values; thus, flows going from the vapor to the liquid are assumed positive. Variables $\mathcal{N}_{n,i}$ were already introduced in Section 3.2.1, in Eqs. (3.1.A) and (3.1.B), and their definition is included in the model through Eqs. (3.9.H) and (3.9.I) acknowledging that microscale vapor-liquid interactions only occur when mass transfer is allowed, i.e., when $YE_n = 1$. The presence or absence of input flows at stage n is characterized by the multiplication between $YF_{n,f}$ and its associated feed stream IN_f^L or IN_f^V, with molar compositions $z_{i,f}^L$ and $z_{i,f}^V$ respectively. Moreover, reaction terms appear in partial mole balances for the liquid phase, were binary variables (YC_n) indicate the existence of reaction if their value is equal to 1. If a reaction occurs at stage n, stoichiometric coefficients ($\nu_{i,u}$) and reaction rates ($\mathcal{R}_{n,u}$) per mass of catalyst ($m_{cat,n}$) are given by a chemical equation and a suitable kinetic model, e.g., the LHHW expression in Eq. (3.1.G). Note that reaction terms are not included in energy balances because they are implicitly incorporated in the enthalpy calculations.

The rate at which energy ($\mathcal{E}_n$) and molar ($\mathcal{N}_{n,i}$) transfer occurs is governed by Eqs. (3.9.G)-(3.9.O), which depends on the binary term YE_n. Given the assumption of mass transfer in reactive stages, when a stage n is located above the reflux stage or below the boil-up stage (i.e. $YE_n = 0$), the interphase

mole and energy transfer, the reboiler and reflux flows, and the possibility of having reactions or feeds are all set to zero. This results in the input-output process required for *leftover stages*. Alternatively, rate-based equations are considered for *real stages* when $YE_n = 1$. Furthermore, additional variables must be calculated by adding new equations in terms of the problem's independent variables, e.g., vapor and liquid flows, compositions, bulk temperatures, interphase properties, etc. For example, Eqs. (3.9.N) and (3.9.O) calculate bulk temperatures by considering that the vapor and liquid bulk phases inside the stages are saturated; thus, additional K-values with their respective saturation pressures and activity coefficients must be added for each phase. Also, Eqs. (3.9.N) and (3.9.O) avoid having liquids and vapors above and below their bubble and dew temperatures, respectively; this phenomenon was originally reported by Gómez[201] for nonequilibrium CD models. The new variables introduced by these equations are the bubble temperature of the liquid T_n^{BUBBLE} and the dew temperature of the vapor T_n^{DEW}, which are calculated at the state of the liquid and vapor bulk, respectively. Concerning mole transfer rates $\mathcal{N}_{n,i}$, the diffusive contribution, e.g. $c_n^L \sum_{m \in COMP_R} a_n^I k_{n,i,m}^L \left(x_{n,m}^I - x_{n,m}\right)$, introduces the product of multicomponent mass transfer coefficients ($k_{n,i,m}^L$) with interfacial areas (a_n^I) and total molar concentrations, which are calculated as described in Section 3.2.1.

Eqs. (3.10.A)-(3.10.E) represent macroscale hydrodynamic feasibility constraints described in Section 3.2.2, which depend on bulk pressures, temperatures, compositions, and vapor/liquid flows. These equations avoid the undesired effects previously discussed, i.e., entrainment flooding, downflow flooding, weeping, and catalyst flooding. More details about these constraints can be found in [12,236].

Entrainment flooding: maximum vapor velocity

$$YP_n\left(u_{V,n} - u_{V,n}^{max}\right) \leq 0, \forall n \in INT \tag{3.10.A}$$

Entrainment flooding: maximum liquid velocity

$$YP_n\left(u_{L,n} - u_{L,n}^{max}\right) \leq 0, \forall n \in INT \tag{3.10.B}$$

Downflow flooding: maximum liquid's pressure drop

$$YP_n\left(P_{ow,n} - P_{ow,n}^{max}\right) \leq 0, \forall n \in INT \tag{3.10.C}$$

Weeping: Minimum vapor velocity through the holes

$$YP_n(u_{h,n}^{min} - u_{h,n}) \leq 0, \forall n \in INT \tag{3.10.D}$$

Catalyst flooding: maximum catalyst's pressure drop

$$YC_n\left(\Delta P_{C,n} - \Delta P_{C,n}^{max}\right) \leq 0, \forall n \in INT \quad (3.10.E)$$

3.3.5 Superstructure example

An illustrative design that can be obtained with the proposed superstructure is shown in Figure 3.5. This superstructure has $NT = 5$ stages, with one condenser, one reboiler, two *leftover stages*, and one *real stage*, where all the independent and dependent binary variables are activated. Consequently, the rate-based equations that combine multiphase and multiscale phenomena are turned on at $n = 3$. In the upper section of the superstructure, there is a *leftover stage* ($n = 2$) that connects the *real stage* with the condenser; thus, vapor flow and energy must be conserved when it passes through this stage. Mass and energy are also conserved for the liquid entering at $n = 2$; therefore, the input liquid flow at this stage must vanish as described in Eq. (8.B). The equivalent situation occurs at stage $n = 4$, where mass and energy are conserved and the vapor input flow must be specified at 0, i.e., see Eq. (8.D).

Note that the binary variables in the formulation above are not independent of each other, i.e., the existence of the stage (through YE_n) supersedes the choice of whether it is a reactive or non-reactive stage (YC_n), or if there are input flows on that stage ($YF_{n,f}$). Hence, the proposed MINLP formulation for this CD column, which includes all the constraints and variables, might not transmit this precedence information to the MINLP solvers. For instance, the binary formulation conveys the single discrete decision of selecting the total number of stages by using multiple binary variables that decide if a candidate stage of the superstructure will be used in the final design. This issue motivated the development of a reformulation and a solution strategy that allows the optimization over single discrete variables (e.g., the actual total number of stages), instead of a large number of binary variables (e.g., one for each candidate stage). This method, referred to as the D-SDA, is explained below.

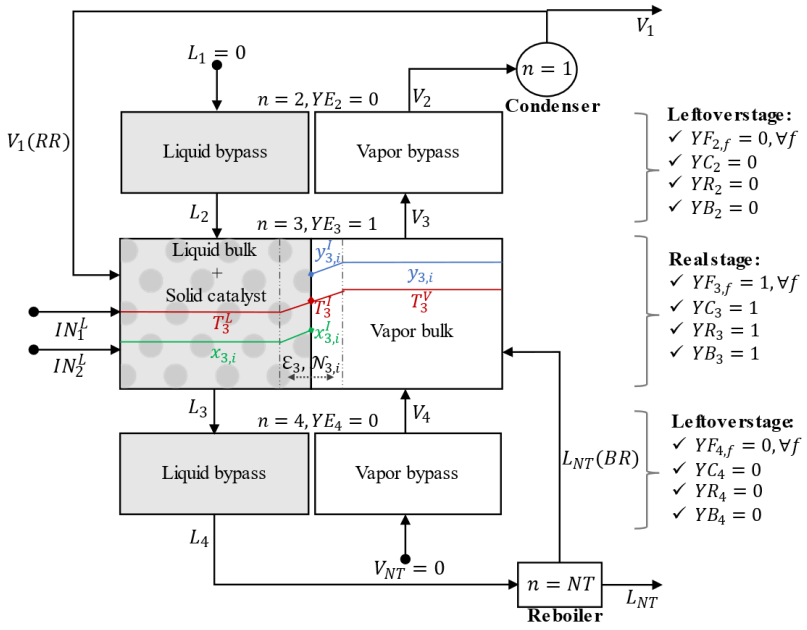


Figure 3.5. Example of a design that can be obtained with the proposed MINLP superstructure.

3.4 Description of the D-SDA

The rate-based CD model described in Section 3.3 can be used to optimize intensified systems. Recently, a Discrete-Steepest Descent Algorithm (D-SDA) was proposed to optimally place units and streams over superstructures with ordered locations[12]. Given an initial superstructure for a distillation system, D-SDA executes a local search to optimize the process according to an objective function. This algorithm is particularly suited for the optimal synthesis and design of distillation columns since it can handle the difficulties commonly associated with this problem, i.e., the inherent bilinear terms that appear in MESH/MERSHQ models, the presence of polynomial and transcendental functions in thermodynamic, reaction, hydrodynamic and mass transfer constraints, and the complexity that arises when discrete and continuous variables are involved, e.g., lack of convergence with streams of zero flows. These issues often do not allow existing MINLP solvers to converge or attain the same local solution for different initializations [12]. On the other hand, the D-SDA is based on discrete convex analysis[6], which provides a different local optimality criterion, referred from heretofore as *integrally local* optimality (*i-local* optimality). This feature allows the D-SDA to perform better than the available MINLP solvers when performing the optimal design of distillation columns [12]. A brief description of this optimization framework is provided next.

The D-SDA relies on the fact that multiple binary variables can be reformulated as a single integer variable when the objective is to place a stream over a superstructure. To illustrate the idea, Figure 3.6 compares the conventional formulation (Figure 3.6A) with an equivalent reformulated formulation (Figure 3.6B) for a column with 14 internal stages. The conventional formulation sets a feed over the superstructure by considering 14 binary variables. Instead, the proposed reformulation uses a single integer variable x_E associated with a single feed. This reformulation of the integer variables is referred to as *external variables* because they cannot be explicitly written in an algebraic optimization model; hence, they must be handled by a master optimization layer. This results in a master-subproblem decomposition, where *external variables* are manipulated in a master Integer Problem with Linear Constraints (IPLC) and the vector of continuous variables $\boldsymbol{x}$ is optimized in Nonlinear Programming (NLP) subproblems with the discrete variables fixed, i.e., Fixed Nonlinear Programming (FNLP) subproblems. The D-SDA introduced by Murota[237] is used to explore the domain of *external variables* in the master IPLC, while the available NLP solvers can be used for the FNLP subproblems. For the example shown in Figure 3.6, the IPLC selects the best decisions for the optimization variable e within its feasible domain ($2 \leq e \leq 15$), i.e. increase or decrease the feed's position to minimize the column's economic function. Accordingly, the objective function will be calculated by the FNLP subproblems at each value of the *external variable* defined by the IPLC. To do so, each FNLP subproblem computes the objective function by optimizing all the continuous variables with e fixed, where independent and dependent binary variables are calculated correspondingly, e.g., $YF_n = 1\ if\ e = n$ and $YF_n = 0$ otherwise.

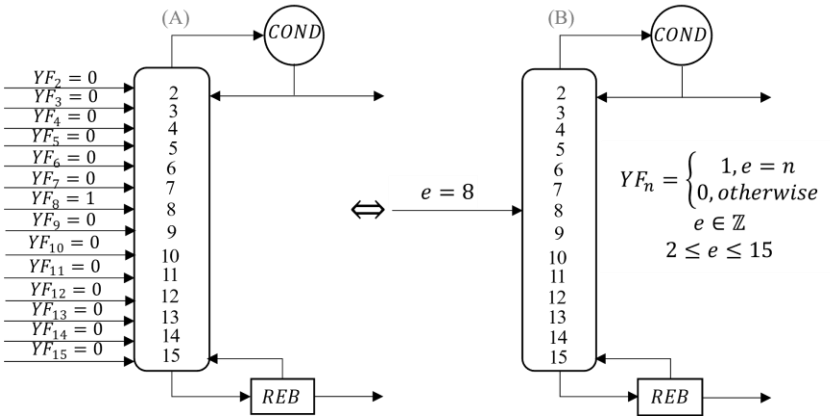


Figure 3.6. Motivating D-SDA reformulation example: A) Conventional formulation, B) Reformulated strategy.

The D-SDA algorithm can be summarized as follows: first, an initial design selected from the superstructure is specified by the user to initialize the D-SDA's local search. Then, a *k-neighborhood* algorithm is used to verify if there is no other solution better than the current design; where a *k-neighborhood* comprises nearby discrete points within an ℓ_k-distance smaller or equal to 1. If there is no direction of improvement, then stop, as an optimal solution over the *k-neighborhood* has been found; otherwise, the best direction of improvement is selected as the steepest-descent direction $\boldsymbol{\delta}$. Next, a line search evaluates each candidate over $\boldsymbol{\delta}$ one by one, until there is no improvement in the objective function. The best solution along the line search is selected, and a new iteration is performed. Another issue addressed by the D-SDA is the difficulty in the convergence of the NLP solvers. Note that the D-SDA method performs a custom initialization of the NLP subproblems at each iteration step. This initialization routine considers that the continuous variables are reinitialized to keep providing good starting points for the FNLP subproblems once a new candidate solution is found. Furthermore, the line search is an optional step, in the sense that the D-SDA can also explore the *external variables'* search space by relying solely on neighborhood exploration; nonetheless, it has been observed in practice that consecutive successful iterations in a line search may improve the objective function significantly with a small number of subproblem evaluations[22]. Monotonicity in the objective function is thus desired (but not required) to efficiently search the space of *external variables* when performing the line search. More details about the D-SDA are provided in section C.1 of appendix C.

Based on the above, the D-SDA framework is suitable to optimize CD columns with rate-based nonlinear models since it can deal with integer and continuous variables. This is because the D-SDA efficiently explores the feasible region of discrete variables by using *external variables*, avoiding multiple binary variables that are instead replaced by a single discrete decision. This allows the implementation of an alternative stopping criterion that is not executed by any of the existing MINLP or logic-based local solvers, demonstrating to attain better local solutions for CD design [12]. The proposed framework also implements a recalculation procedure for the continuous variables' initialization while executing the main search over the *external variables*' domain. This allows to keep updating the initialization for the nonlinear subproblems, and thus improving the search performed by the NLP solver selected to achieve convergence. Furthermore, the D-SDA's IPLC considers discrete decisions only; this offers two advantages: (1) additional reformulations in the upper-level problem are not required, e.g., the linearization of products between continuous and discrete variables used in the Logic-Based Outer Approximation [238]; and (2) all discrete decisions found by the IPLC are integer

solutions; thus, any formulation can be used for the subproblems, e.g., a problem with discrete variables or a black-box simulation. A case study that applies the D-SDA to the optimal design and intensification of a rate-based CD column is presented next.

3.5 Case study

A CD column that produces ETBE as a product from the exothermic reaction between isobutene and ethanol is considered. The chemical reaction is modeled using the Langmuir-Hinshelwood-Hougen-Watson (LHHW) model available in [239]. The chemical species involved are isobutene, ethanol, ETBE, and n-butene, where the first three appear in the main reaction, while the n-butene acts as an inert component. Nonideal equilibrium interactions between these species are considered by using the NRTL model for activity coefficients in the liquid and the Soave-Redlich-Kwong equation for vapor fugacity coefficients [236]. The feedstreams into the CD unit are specified assuming a split feed mode, i.e., a feed of pure ethanol ($0.029\ mol/s$ at $T = 342\ K$) and a second feed of n-butene ($0.096\ mol/s$ at $T = 323\ K$) with isobutene ($0.3\ mol/mol$). This information is summarized in Eqs. (3.11.A)-(3.11.C). The split flow mode results in a column with three main zones: the rectifying zone, the reactive zone, and the stripping zone. The rectifying section is used to remove light inerts, e.g. n-butene, while preventing excessive loss of ethanol in the distillate by lowering ethanol's feed and allowing mass transfer of this component from the vapor to the liquid phase. The reactive zone is located below the rectifying zone, from the ethanol's feed to the butenes' feed; this comprises the sections where the acid resin Amberlyst 15® is placed to catalyze the exothermic reaction. The stripping stages located below the butenes' feed are used to purify the ETBE product and to keep the light reactant isobutylene inside the column by allowing the mass transfer of this component from the liquid to the vapor phase [239,240]. Since the ETBE is the desired product, a constraint is enforced to ensure the desired product quality. In this case, the ETBE bottom's composition is expected to be at least $x_{NT,ETBE}^{min} = 0.83$ (see Eq. (3.11.D)). Additional design considerations are: the condenser's operating pressure is set to $9.5 \times 10^5\ Pa$ in Eq. (3.11.E); the column must guarantee enough space to place the catalyst according to Eq. (3.11.F), where ρ_{cat} is the catalyst's apparent density; and the column must respect a typical length (h_{total}) to diameter (D_C) ratio specified in Eq. (3.11.G). For more information about these constraints the reader is referred to a previous work[12].

Set of components

$$COMP = \{Isobutene, Ethanol, n-butene, ETBE\} \quad (3.11.A)$$

Set of feeds

$$F = \{1,2\} \tag{3.11.B}$$

1: Ethanol feed. 2: Butenes feed (isobutene $0.3\ mol/mol$ *and n-butene* $0.7\ mol/mol$*)*

Main reaction

$$isobutene + Ethanol \leftrightharpoons ETBE \tag{3.11.C}$$

Product specification constraint

$$x_{NT,ETBE} \geq x_{NT,ETBE}^{min} \tag{3.11.D}$$

Condenser's operating pressure

$$P_1 = 9.5 \times 10^5\ Pa \tag{3.11.E}$$

Available catalyst volume

$$\frac{m_{cat,n}}{\rho_{cat}} \leq (YC_n)(Available\ volume) \tag{3.11.F}$$

Length to diameter ratio

$$h_{total} \leq 20 D_C \tag{3.11.G}$$

The case study in Eqs. (3.11.A)-(3.11.G) has been previously used to perform the optimal design of a CD column using a MESH equilibrium model[12,236,241]; however, its deterministic optimization with rate-based models has not been previously considered in the literature; hence the novelty of this work. The discrete configuration of the column optimized in those previous studies has a total condenser, a partial reboiler, and eight internal stages, with three consecutive reaction stages. Both Miranda et al.[241] and Bernal et al. [236] focused on the sequential and simultaneous design and control, respectively. In these studies, the discrete decisions were fixed to reduce the model's complexity. Liñán et al.[12] performed the optimal design and operation of the CD unit while considering rearrangements of the equipment's discrete configuration, i.e., feed stages, catalyst stages, and total number of stages. Based on the above, the preceding works can be used as a baseline to compare the optimal synthesis and design obtained with the rate-based CD model presented in section 3.3. Section 3.5.1 verifies the

viability of those previous works from a rate-based model perspective, whereas Section 3.5.2 presents the CD column's optimal synthesis and design. All simulations and optimization runs were implemented in GAMS 32.1[242] and solved using a PC with an Intel® Core™ i7-7700 CMU @ 3.6 GHz and 16GB of RAM.

3.5.1 Performance: Rate-based CD model

As discussed in the introduction section, the available literature provides background to support that rate-based models are more realistic than equilibrium models; consequently, the rate-based approach is used herein to verify the steady-state optimal design specified by Bernal et al.[236]. The first validation is a rate-based simulation of that column, with the operation and design variables fixed at the reported values, i.e., reflux ratio (RR), reboiler duty (Q_{Reb}), column diameter (D_c), stage height (h_s), weir height (h_w) and discrete configuration as specified in Figure3.7A. This simulation results in a column that does not satisfy the product specification constraint, with an ETBE bottoms composition of $x_{REB,ETBE} = 0.55$, i.e. 33% below the minimum threshold ($x^{min}_{REB,ETBE} = 0.83$). If this behavior is detected in practice, it is still possible to modify the operating variables to comply with the ETBE's composition target while keeping the design (integer) variables fixed. This is expressed as the optimization problem shown in Figure3.7B, where the squared difference between the bottoms ETBE composition and its target is minimized. The results of this optimization specify a column with a reflux ratio of $RR = 2.383$, a reboiler duty of $Q_{Reb} = 5.057\ kJ/s$ and a product composition of $x_{REB,ETBE} = 0.68$. This result shows that larger reflux and reboiler duties are required to overcome microscale mass and energy transfer limitations. However, these variables are limited by the large-scale hydrodynamic inequality constraints, which results in a limitation in achieving the desired ETBE product composition. To further investigate this behavior, hydrodynamic constraints in Eqs. (3.10.A)-(3.10.E) can be relaxed for this optimization problem, resulting in a composition of $x_{10,ETBE} = 0.82$; however, this composition is still infeasible and hydrodynamic limitations cannot be neglected in a real setting. Thus, assuming the rate-based model accurately describes the multiscale phenomena inside a real column, the design proposed by Bernal et al.[236] would become infeasible in practice.

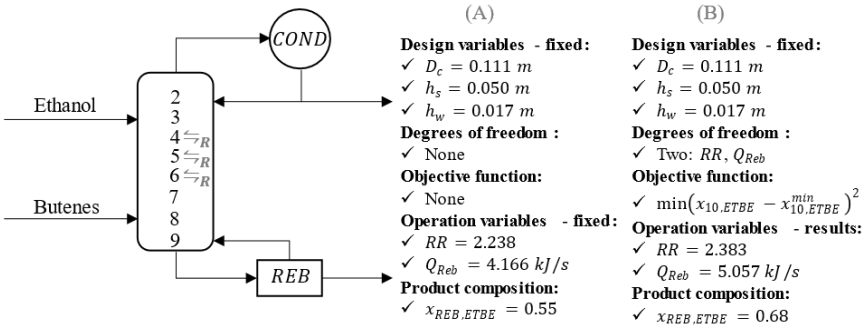


Figure 3.7. Rate-based simulation (A) and optimization (B) with of design presented in [236]. Symbol $\leftrightharpoons_R$ used for reactive stages.

Another CD column design available is proposed in our previous work [12]. This design results from the column's economic optimization with a MESH model while allowing discrete decisions to change, e.g., the total number of stages, location of feeds, and location of catalytic stages (Figure3.8A). The simulation of this column using the proposed rate-based model shown in Section 3.3 produces an ETBE with a composition of $x_{REB,ETBE} = 0.73$ thus making this design infeasible according to the rate-based CD model. Moreover, this column was designed under hydrodynamic constraints; however, the rate-based simulation suggests that this column does not satisfy the weeping constraint at every stage, i.e., it is hydrodynamically infeasible. This behavior has been reported before by Baur et al.[243], where it was concluded that results from equilibrium models might be infeasible in terms of flooding or weeping limitations when tested with the more realistic rate-based model. Accordingly, if the design (integer) variables remain fixed, it is still possible to search for operating conditions that guarantee hydrodynamic feasibility and a product with the required specification. In contrast to the column in Figure 3.7, the design found in[12] attains feasibility when operating variables are modified. Therefore, the optimization problem in Figure 3.8B aims to change the reflux ratio to minimize the column's reboiler duty. This not only allows one to achieve a feasible design but also to search for a solution with minimum energy requirements. The resulting column in Figure 3.8B satisfies the product specification constraint, hydrodynamic operation constraints, and rate-based equations. The results show that the heat required by this rate-based design is 12% larger than that required with the MESH model. This result agrees with outcomes reported by Baur and Krishna[244] for TAME's catalytic production, i.e. an equilibrium model may underpredict the reboiler load compared to the rate-based model. If the nonequilibrium premise is true, it is expected that those energy requirements will be closer to the actual

energy consumption for this column; however, there is no guarantee that this design is indeed optimal since discrete design variables remained fixed in the calculations. Hence, this feasible rate-based design can be used as a starting point for this column's optimal economic design where discrete and continuous variables are simultaneously considered (this optimization is performed in Section 3.5.2).

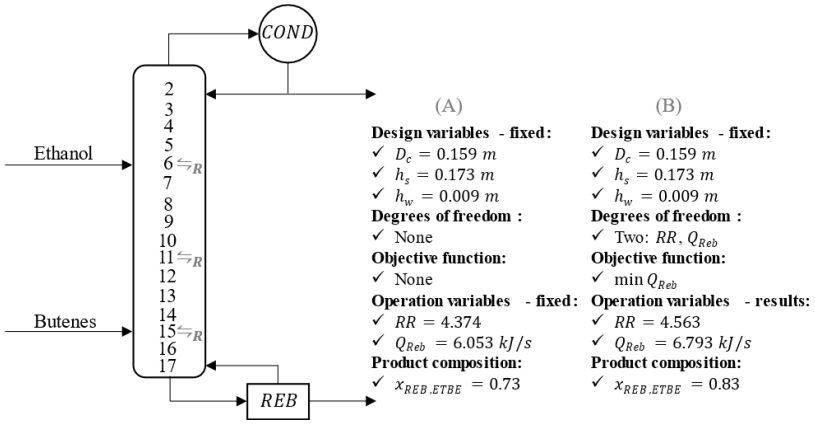


Figure 3.8. Rate-based simulation (A) and optimization (B) with of design presented in [12]. Symbol $\leftrightharpoons_R$ used for reactive stages.

Note that the results presented in this section do not consider discrete decisions as variables, e.g., number of stages; thus, the NLP solver CONOPT was selected to perform this analysis. Similarly, *Multi-Start* NLP (MSNLP) [245] was used to verify the results. The simulations and optimizations presented in this section were initialized using previous equilibrium results. First, consider that the mathematical model in Section 3.3 is equivalent to an equilibrium model when Eqs. (3.9.H) and (3.9.I) are replaced with the *simplifying constraints* $0 = ye_n\left(y_{n,i} - y^I_{n,i}\right)$ and $0 = ye_n\left(x_{n,i} - x^I_{n,i}\right), \forall n \in INT, \forall i \in COMP_R$. Thus, the first initialization stage consists of initializing the rate-based model with *simplifying constraints* by: (1) merging those variables that are common in both MESH and MERSHQ models, e.g. compositions, temperatures, equilibrium constants, etc., and (2) providing guess values of 1 for rate-based variables, e.g. transfer coefficients, thermodynamic factors, etc. Then, the resulting set of variables were used to initialize the rate-based model presented in Section 3.3 and to obtain the results presented in this section.

3.5.2 Rate-based optimization

Based on the results presented above, this section aims to propose a more realistic optimal design for the ETBE CD column using the D-SDA framework as a suitable optimization tool to allow the

optimization of this intensified multi-scale process. Additional logical constraints for this specific case study are presented first, followed by the statement of the economic objective function and the resulting optimization problem.

Additional logical constraints must be considered to comply with the specific requirements of the case study. The first consideration has to do with the concept of *structural redundancy*. It occurs when multiple combinations of binary variables represent the same physical structure, and it is a recurring phenomenon when dealing with superstructures in chemical engineering [246]. This issue has been resolved in the present case study by fixing the reflux location and allowing the boil-up location to define the total number of stages ($NT = 45$), as shown in Figure 3.9.

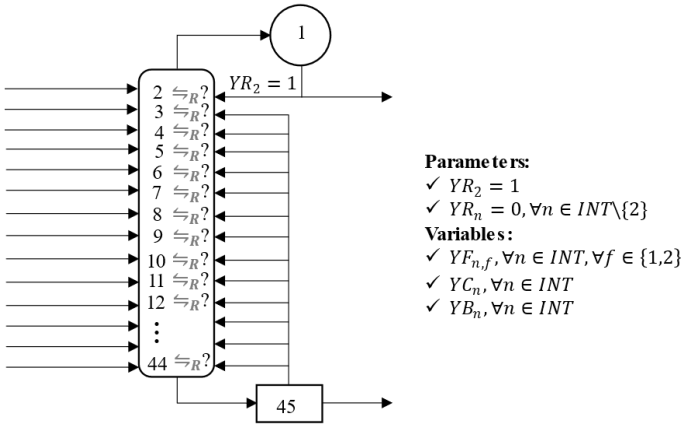


Figure 3.9. Representation of a strategy that avoids structural redundancy. Symbol $\leftrightharpoons_R$ used for reactive stages.

The design aspects for this specific case study are mathematically stated through Eqs. (3.12.A)-(3.12.D). These constraints guarantee the existence of the three zones defined within the CD column, i.e., rectification, reaction, and stripping; these constraints also allow these sections to vary in number. As shown by Eq. (3.12.A), the maximum number of reaction stages is fixed at three, following the recommendation by Sneesby et al.[239], which performed a detailed analysis of the advantages and disadvantages of varying the number of consecutive catalytic stages. In this work, the author analyzes the effect of keeping this number constant while allowing catalytic stages to be optimally placed along the column, i.e., catalytic stages are not forced to remain consecutive and are not necessarily bounded by the feeds. This allows us to compare results from this work with the column designed using an equilibrium model in our previous work, where the same logical constraints were considered. Note that

the present optimization problem is more challenging from a combinatorial point of view since it results in six discrete decisions (i.e., total number of stages, location of two feeds and three catalytic stages) compared to the conventional optimal design of CD columns that only consider the stages needed for the rectification, reaction and stripping sections, i.e., three discrete decisions.

There is a fixed number of catalytic stages

$$\sum_{n \in INT} YC_n = 3 \tag{3.12.A}$$

The ethanol feed is above the butenes feed

$$\sum_{t=2}^{n} YF_{t,1} \geq YF_{n,2}\ , \forall n\ \in INT \tag{3.12.B}$$

The catalytic stages are below the ethanol feed stage

$$\sum_{t=2}^{n} YF_{t,1} \geq YC_n\ , \forall n\ \in INT \tag{3.12.C}$$

The catalytic stages are above the butenes feed stage

$$\sum_{t=2}^{n} YF_{t,2} - YF_{n,2} \leq 1 - YC_n\ , \forall n\ \in INT \tag{3.12.D}$$

The economic objective function (f_{OBJ}) considered in this study represents the minimization of the total annualized cost, as shown in Eq. (3.13.A). This economic function was selected to keep consistency with previous works [12,236] for comparison purposes. The objective function represents the addition of the annualized investment cost (C_{inv}) and the annual operating costs (C_{op}) minus the annualized ETBE profit (C_{ETBE}), as shown in Eq. (3.13.B) to Eq. (3.13.D), respectively. The investment cost requires parameters such as the fixed annualized investment cost (C_0), which considers the fixed cost of installation of both the condenser and the reboiler; the annualizing factor (AF), calculated with a payback period of 5 years and an annual interest rate of 5%; the catalyst cost; a safety factor (SF) for the column height of 15% [214]; and the Marshall and Swift index ($M\&S$) of 1050 consistently used in previous works [12,236,241]. The remaining parameters are the annualized cost of feeds (C_1 and C_2); the annualized reboiler and condenser investment and operating costs affected by heat duties and lumped into single terms (C_{Reb} and C_{Cond}), and the annualized selling price of ETBE (C'_{ETBE}). Economic parameters are calculated assuming an operation of 8,000 hours per year.

Cost function

$$f_{OBJ} = Total\ cost = C_{inv} + C_{Op} - C_{ETBE} \quad (3.13.A)$$

Annualized investment cost

$$C_{inv} = C_0 + AF\left[C_{cat}\sum_{n\in INT} m_{cat,n} YC_n + \left(\frac{M\&S}{280}\right)\left(117.85 D_c^{1.066}\left((1+SF)\left(\sum_{n\in INT} h_s YP_n\right)\right)^{0.802} + 4.7 D_c^{1.55}\sum_{n\in INT} h_s YP_n\right)\right] \quad (3.13.B)$$

Annual operating costs

$$C_{op} = IN_1 C_1 + IN_2 C_2 + Q_{Reb} C_{Reb} + Q_{Cond} C_{Cond} \quad (3.13.C)$$

Annualized ETBE profit

$$C_{ETBE} = L_{NT} C'_{ETBE} \quad (3.13.D)$$

The resulting optimization problem is stated in Eq. (3.14). This problem comprises the minimization of the economic objective function in Eq. (3.13.A), subject to microscale and macroscale constraints. Microscale phenomena are directly affected by discrete decisions and they account for multicomponent convective and diffusive mass transfer between phases with interphase equilibrium (Eqs. (3.1.C)-(3.1.F),(3.9.G)-(3.9.L)); reaction kinetics limited by chemical equilibrium (Eq. (3.1.G)); and rates of energy transfer governed by saturation constraints (Eqs. (3.9.M)-(3.9.O)). Similarly, macroscale equations consider general structural discrete decisions and constraints (Eqs. (3.3.A)-(3.5.H)); condenser and reboiler mole and energy balances (Eqs. (3.6.A)-(3.8.D)); mole and energy balances for internal stages (Eqs. (3.9.A)-(3.9.F)); pressure drop calculations (Eq. (3.9.P)); hydrodynamic feasibility inequality constraints (Eqs. (3.10.A)-(3.10.E)); and additional discrete and continuous constraints for the specific case study, e.g. desired product specification (Eqs. (3.11.A)-(3.12.D)). In this problem, the type of column, i.e. stage column with sieve trays, the input flows and top operating pressure are specified *a priori*.

Conceptual Optimization problem:

$$\min \{ \text{Eq. (3.13.A)} \}$$

$$s.t. \quad (3.14)$$

$Equations$: (3.1.C)-(3.1.G), (3.2.A)-(3.2.G), (3.3.A)-(3.3.D), (3.4.A)-(3.4.B), (3.5.A)-(3.5.H), (3.6.A)-(3.6.J), (3.7.A)-(3.7.D), (3.8.A)-(3.8.D), (3.9.A)-(3.9.P), (3.10.A)-(3.10.E), (3.11.A)-(3.11.G), (3.12.A)-(3.12.D), (3.13.B)-(3.13.D).

$Major\ decisions\ (degrees\ of\ freedom)$: reflux ratio ($RR$), reboiler duty ($Q_{Reb}$), column diameter ($D_c$), stage height ($h_s$), weir height ($h_w$), number of stages, feeds location, reactive stages distribution.

3.5.2.1 D-SDA optimization results

The optimization problem summarized in Eq. (3.14) was solved using the D-SDA framework described in section 3.4. From a degree of freedom analysis performed by Gómez[201], this problem results in 1,147 continuous state variables, i.e., compositions, temperatures, flows, and condenser duty; 2 continuous input/operation variables, i.e., reflux ratio and reboiler duty; 3 independent continuous design variables, i.e., diameter, stage height, and weir height; and 172 independent binary variables. When implementing the D-SDA, an *external variable reformulation* (EVR) must be specified to express the 172 independent binary variables of the problem in terms of 6 *external variables* related to the 6 discrete decisions of the problem explained above. Thus, the D-SDA allows decreasing the degrees of freedom of this problem from 173 to 11 decisions with physical meaning, i.e., 6 discrete decisions, 3 design variables, and 2 operation variables. The implementation of each NLP subproblem in GAMS results in a problem with 14,380 constraints and 14,916 continuous variables. Since the NLP subproblems have few degrees of freedom and feasible or near-feasible initializations, active set methods such as those implemented in CONOPT and CONOP4 are preferable [247]. Moreover, the resulting NLP subproblems are nonconvex; accordingly, combining CONOPT with *Multi-Start* NLP (MSNLP) yields solutions more likely to be the global optimal without resorting to rigorous NLP global solvers, which become prohibitively expensive for problems of this size. Hence, the author decided to solve the NLP subproblems using CONOPT, CONOPT4, and MSNLP for comparison purposes. The D-SDA was executed with 2-*neighborhoods* to save computational time; the solution obtained was then verified using the ∞-*neighborhood* to guarantee *i-local* optimality. More details on the D-SDA framework can be found in appendix C.

The D-SDA was initialized using the design found from the results presented in Figure 3.8B. The optimization only required one execution of the D-SDA with 2-*neighborhoods* that required 9, 10.4, and 77 minutes for CONOPT, CONOPT4, and MSNLP, respectively. The solution obtained was verified with the ∞-*neighborhood* and resulted in an *i-local* solution after 18, 53, and 240 minutes of computation using CONOPT, CONOPT4, and MSNLP, respectively. In this case, the optimal solution obtained was independent of the NLP solver. The results of the economic optimization using the D-SDA are provided in the last column of Table 3.1 (Current work), and the resulting *i-locally* optimal

discrete configuration is depicted in Figure 3.10B. Table 3.1 also presents the equilibrium design reported by Liñán et al.[12] simulated with the rate-based approach (first column), and a design were only operation variables where economically optimized (middle column called "Operating feasibility"). It is acknowledged that, although results are reported with 3 digits after the decimal point, such degree of accuracy in operating variables during plant operation may not be attainable in practice. Thus, the reported manipulated variable values are just guidelines for the system operation, e.g., set-points for closed-loop operation. It is also acknowledged that, in practice, sizing variables such as column diameters or tray dimensions should be selected from a catalog with a finite number of available sizes to select from, e.g., see [248]. Thus, tray and column sizing considerations should be considered as external variables in the future.

Table 3.1. Results: solution status, feasibility, and continuous variables using a rate-based model for simulation and economic optimization.

Solution source	Liñán et al.[12] (simulation)	Operating feasibility (optimization)	Current work (optimization)
Solution status	Infeasible	Feasible	*i-locally* optimal
Discrete configuration	Figure 3.10A	Figure 3.10A	Figure 3.10B
Entrainment flooding	No	No	No
Downflow flooding	No	No	No
Weeping	Yes (infeasible)	No	No
Catalyst flooding	No	No	No
Molar reflux ratio [-]	4.374	4.563	5.296
Reboiler duty [kJ/s]	6.053	6.793	7.378
Column diameter [m]	0.159	0.159	0.172
Stage height [m]	0.173	0.173	0.149
Weir height [m]	0.009	0.009	0.008
ETBE purity [mol/mol]	0.73 (infeasible)	0.83	0.83
Annualized ETBE profit [$/year]	(Infeasible)	22,377	23,269
Annualized investment cost [$/year]	(Infeasible)	10,230	10,264
Annual operating cost [$/year]	(Infeasible)	36,350	36,450
Objective function [$/year]	(Infeasible)	24,202	23,445

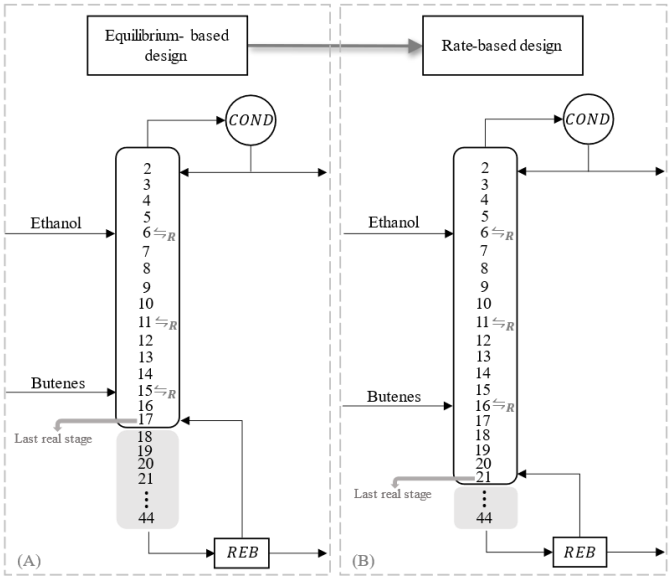


Figure 3.10. Optimization results: discrete decisions. A) Optimal equilibrium-based design [12], B) Optimal rate-based design. Symbol $\leftrightharpoons_R$ used for reactive stages.

As discussed in Section 3.5.1, the equilibrium design with a total cost of $22{,}410\ \$/year$ obtained by [12] is infeasible because it does not satisfy weeping and purity constraints; therefore, it would be infeasible to operate this unit if the rate-based model is a realistic representation of the actual CD unit. Under the assumption that the rate-based model is accurate enough to represent a real-life CD column and that the equilibrium design is used to build the column, then it would be necessary to redefine operation variables to optimize the economic objective function and satisfy design requirements, e.g., reach the desired ETBE purity. This economic optimization results in the same column presented in Figure 3.8B because costs were minimized by decreasing the reboiler duty, i.e., for this specific case, total cost minimization is equivalent to energy minimization assuming that the design variables remain fixed. As shown in Table 3.1, the resulting total cost is $24{,}202\ \$/year$, i.e. the cost of designing this column with an equilibrium model is $1792\ \$/year$. However, these additional costs will only become apparent once the unit has been built and is already in operation. In brief, this additional cost represents the economic penalty associated with the inadequate selection of the CD model used to perform the optimal design. This indicates an economic benefit when CD columns are designed with rate-based models at the conceptual design level. This result also highlights the need to keep investigating and validating rate-based models with experimental and industrial data to make them more attractive for CD synthesis, design, and intensification.

When discrete variables are considered in the formulation, the optimal solution found in this work is a larger column with 22 real stages, including the reboiler and the condenser (see Figure 3.10B and Table 3.1, "Current work (optimization)"). The infeasible equilibrium design reported by Liñán et al.[12] was the starting point of the optimization procedure herein described; thus, the obtained solution is not only feasible but *i-locally* optimal in terms of the rate-based model. In terms of the total cost, it would be unviable to build and operate the initial column reported in [12] since it does not fulfill neither weeping nor ETBE purity requirements. Nevertheless, it is still possible to calculate hypothetical costs for that unit; for instance, the minimum ETBE purity requirement is not satisfied, as shown in Table 3.1; hence, no profits will be reported for that column. Moreover, the investment cost would be $10{,}230\ \$/year$; the operating costs would slightly increase from $36{,}224\ \$/year$ (equilibrium) to $36{,}225\ \$/year$ (rate-based) due to changes in the condenser duty. As a result, the objective function would be $46{,}455\ \$/year$. Therefore, the optimization of the CD column using a rate-based model allowed a percentual change in the objective function of 50%. Furthermore, the optimal column found in this work ("Current work (optimization)" in Table 3.1) allows annualized cost savings of 3% when compared to the initial design with feasible operating variables ("Operating feasibility" in Table 3.1). These results show that the economic objective function considered in this study is mainly influenced by the continuous operating variables, while discrete and continuous design variables only play a minor role. For instance, as more stages are added to the unit, the reboiler duty tends to increase such that more ETBE with the desired product quality is produced, which results in an overall increment in profits [12].

In terms of process design, the most apparent change between Figure 3.10A and Figure 3.10B is an increment from 2 to 5 stripping stages for the rate-based column. These additional stages allow further enrichment of the ETBE product to attain the desired specification while making a marginal deterioration in the process's economics. A larger requirement of stripping stages by the rate-based model agrees with the results found by Baur and Krishna[244]. That study concluded that an equilibrium model might overestimate the separation efficiency in the stripping section for TAME's CD synthesis. Regarding the distribution of reaction stages and feeds location, the optimal solution follows the same trend as the optimal equilibrium-based model design, i.e., the reaction must be allowed at feed stages, and the remaining catalytic stage should be allocated uniformly between feeds. The optimal arrangement of catalytic stages will enable one to use the heat of reaction for separation optimally; to improve reaction efficiencies by allocating reaction stages in connection with reagents' compositions

and temperatures; and to change the species' composition to overcome the limitations imposed by azeotropes [201]. Additionally, the ETBE output flow has the most considerable positive impact on the objective function. In this case, a larger reboiler duty of $7.378\ kJ/s$ is needed not only to overcome heat and mass transfer limitations but also to allow more ETBE to be produced since the objective function is very sensitive to small changes in the outlet flow of ETBE, e.g. when comparing the initial feasible design and the *i-locally* optimal solution, an increment of 0.3% in operating costs caused by a larger heat requirement allows the ETBE profit (or output flow) to increase by 4%. The remaining variables do not have a significant effect on the objective function but they allow to have an optimal design that meets both macroscale constraints, e.g. hydrodynamic, discrete, structural, and geometric considerations, and microscale events, i.e. catalytic reactions, mass transfer, energy transfer, and interphase equilibrium.

Note that the validity of the results presented here is based on the broad range of previous studies that have validated rate-based models for CD distillation with experimental data taken from existing CD/RD units, e.g. [189–199]. Chemical systems considered in those works include ETBE, ethanol, tert-butanol; TAME, 2-methyl-1-butene, 2-methyl-2-butene, methanol; MTBE, methanol, isobutene, etc. Nonetheless, there are no reported experiments addressing the rate-based design, construction, and experimental validation of a RD/CD unit.

3.5.2.2 Comparison with state-of-the-art MINLP algorithms

To compare our D-SDA framework's computational performance in the solution of the proposed rate-based CD model, the MINLP problem presented in this work was solved using different MINLP solvers. A description of those solvers can be found in [249]. The following local solvers were considered: Alpha-ECP [250,251], DICOPT [252], and SBB [253]. Similarly, I also compared the performance of our D-SDA framework to MINLP global solvers, i.e., BARON [254–256], LINDOGLOBAL [257] and SCIP [258–260]. The MINLP problem involves 16,106 constraints and 15,389 variables, of which 172 are independent binary variables. Note that the resulting MINLP problem includes $10'737{,}573$ possible combinations for the discrete variables, thus making a *complete enumeration*[e] intractable to validate the results. The feasible initialization point in Figure 3.8B used for the D-SDA was also used to initialize these solvers. Also, a

[e] A *complete enumeration* is the exhaustive optimization of every possible discrete configuration.

time limit of 1 hour was set for those solvers. Table 3.2 compares the performance of those solvers to the D-SDA framework considered in this work.

Table 3.2. Solution times and objective function values for different solvers and initialization points.

Initialization	Liñán et al.[12]		Optimal solution	
Solver	Time [s]	Solution [$/year]	Time [s]	Solution [$/year]
D-SDA (CONOPT)[a]	**540.8**	**23,445**	**18.9**	**23,445**
D-SDA (CONOPT4)[a]	626.1	**23,445**	78.2	**23,445**
D-SDA (MSNLP)[a]	4,630.3	**23,445**	588.3	**23,445**
D-SDA (CONOPT)[b]	1,680.4	**23,445**	1,158.5	**23,445**
D-SDA (CONOPT4)[b]	3,848.7	**23,445**	3,300.8	**23,445**
D-SDA (MSNLP)[b]	19,059.2	**23,445**	15,017.2	**23,445**
Alpha-ECP	1,726.2	No solution found	1,789.6	**23,445**
DICOPT	10.6*	24,176	58.3*	**23,445**
SBB	721.9	23,592	976.2	**23,445**
BARON	3,600.0	24,202	3,600.0	**23,445**
LINDOGLOBAL	3,600.0	24,202	3,600.0	**23,445**
SCIP	45.9**	Infeasible	39.4	**23,445**

[a]:D-SDA executed with 2-*neighborhoods*, i.e., *i-local* optimality is not guaranteed.

[b]:D-SDA executed with 2-*neighborhoods*, plus ∞-*neighborhood* verification, i.e., *i-local* optimality is guaranteed.

Bold numbers: represent 1) the shorter computational times without failing and 2) the best objective function found.

*: DICOPT failed after the first iteration because of diverging gradients.

**: SCIP failed by erroneously claiming infeasibility.

As shown in Table 3.2, the D-SDA was the only algorithm that converged to the best solution found of $23{,}445\ \$/year$ using as the initialization the solution reported by [12]. Note that the NLP sub-solver was not found to be sensitive to the solution. In terms of solution time, the D-SDA's best performance was recorded when CONOPT was used as the NLP solver, which is known for identifying and finding feasible solutions fast [247]. CONOPT was also used for MINLP algorithms that required an NLP sub-solver. SBB was the second-best solver at improving the objective function; however, the same issue reported in [12] was found, i.e., SBB only modifies binary variables such as the position of catalyst and feeds, but it finds challenging to alter the total number of stages. DICOPT also reported a better solution

than the first initialization point; however, it failed after the first major iteration because certain gradients diverged, which indicates numerical instability. Similarly, global solvers were not able to find a solution better than the initialization of this problem. SCIP claims that the problem is infeasible, although this is false given that 4 feasible solutions are found for this problem, including the initialization. In contrast, BARON and LINDOGLOBAL found the solution corresponding to the initial integer decisions, i.e., they did not modify any of the discrete variables. Furthermore, these solvers could not close the optimality gap after 1 hour of CPU time. Thus, I increased the time limit to 12 hours for both solvers; this, however, did not improve the bounds reported after an hour of computation.

I also initialized each of the solvers with the best solution found by the D-SDA to test their ability to verify optimality or achieve a better solution (see optimal solution's column in Table 3.2). Once initialized with the best solution found by the D-SDA, no other solver except SCIP was able to confirm global optimality; nonetheless, I avoid relying on this result since the solver failed by determining feasibility using the other initialization. In contrast, the D-SDA cannot provide global optimality guarantees, but it efficiently verifies potential improvement using the neighborhood method considered in this framework. In particular, the best solution found is indeed an *i-local* optimum, based on the theory of discrete convex analysis applied to MINLP problems.

As shown in section 3.3, Eqs. (3.9.G)-(3.9.O) decide if a stage is *real* or *leftover*; thus, infeasible solutions are obtained when binary variables are allowed to take non-integer values during the solution procedure of general-purpose MINLP solvers. Instead of using the product of the binary variables and the constraints, a Big-M reformulation was applied to Eqs. (3.9.G)-(3.9.O), with M values adequately calculated for each constraint [233]. In this work, preliminary optimization runs were performed using the Big-M formulation for all MINLP solvers, but this led to a worsening in the performance in terms of solution times.

Overall, it is advantageous to have a tool such as the D-SDA to solve the optimal design of complex rate-based CD systems efficiently. Still, the limitations of the D-SDA approach are acknowledged. The D-SDA requires a reformulation into *external variables* which, although well fitted for distillation superstructures, may not be trivial for a general class of MINLP problems. Moreover, the algorithm relies on a neighborhood search; therefore, it provides a practical local stopping criterion, but it cannot guarantee global optimality. The algorithm's performance also heavily relies on the solver used to solve the NLP subproblems, where a tradeoff between solution quality and computational effort may exist

for nonconvex problems. The D-SDA also depends on the user's initial point to perform the discrete neighborhood search, which may limit its applicability, e.g., an initial infeasible solution surrounded by other infeasible solutions would make the algorithm stop. On the other hand, some of these limitations are avoided when solving an optimal design and synthesis problem, given that usually a good initial design is provided beforehand and the determination of optimality with respect to neighboring solutions is still informative, e.g., the global optimal solution can be found if convexity conditions are fulfilled. For distillation superstructures, the results show how this algorithm is well fitted given that: 1) it avoids diverging gradients that caused problems for other solvers such as DICOPT; 2) it prevents issues related to zero-flows; thus, special bypassing strategies such as that discussed by Ghouse et al.[261] are not required; 3) its solution strategy avoids nonlinearities arising from binary variables multiplying constraints and does not require special reformulations such as Big-M; 4) it prioritizes certain discrete choices, leading to an improved performance against a naive branch-and-bound search such as SBB; and 5) its alternative optimality test based on neighborhood information is suitable for distillation column design.

Chapter 4
A Benders Decomposition Framework for the Optimization of Disjunctive Superstructures with Ordered Discrete Decisions

This chapter introduces a new algorithm based on discrete convex analysis and Logic-Based Benders Decomposition (LBBD) intended to solve Generalized Disjunctive Programming (GDP) problems involving ordered decisions. This algorithm implements Benders cuts, which allow updating external variables in a master problem at each iteration, while the remaining variables are optimized in a subproblem. Benders cuts can be designed based on the specific characteristics of each optimization problem, which represents an opportunity to exploit special structures through the development of custom Benders cuts[10]. The algorithm proposed in this chapter is the Logic-based Discrete-Benders Decomposition (LD-BD) method and is tested with superstructure problems involving ordered Boolean variables.

Different Benders methods have been proposed to solve nonlinear programming problems involving both continuous and integer decisions[56,262–265]. All benders-based strategies follow the same principle, i.e., they implement a master-subproblem decomposition strategy that decouples a group of complicating variables from the formulation to facilitate the iterative optimization of the problem. Early works in the field proposed a Generalized Benders Decomposition (GBD) methodology applicable to convex MINLP problems[56,266,267]. The key differences between GBD and the proposed LD-BD are that the former was designed to solve MINLPs while the latter attempts to solve GDPs. In addition, GBD generates cuts based on Lagrangean duality while LD-BD generates cuts based on neighborhood information. Similar to the proposed LD-BD method, other works in the field have focused on developing Benders-based methods that exploit special structures within specific problems. Some applications of Benders-based algorithms for MINLP problems include the optimization of distillation sequences[265]; the solution of two-stage and multi-stage stochastic MINLP problems[262–264], and the optimal integration of decision layers, e.g., simultaneous scheduling and control, process design and operation under stochastic uncertainty, and integrated planning, scheduling and dynamics [48,60,268]. While these methods build upon GBD, the proposed LD-BD strategy incorporates concepts of discrete convex analysis within the LBBD theory to introduce a new type of Benders-based method. Note that the proposed LD-BD method does not incorporate features considered by other Benders methods for MINLPs, e.g., including convex relaxations of nonconvex functions within a nonconvex-GBD strategy

that guarantees global optimality[262,268]; adding multiple cuts per iteration to increase convergence speed[48]; or hybridizing GBD with other methods (e.g., branch and bound) to make the algorithm more efficient[60,263].

The LD-BD algorithm overcomes some of the practical limitations of the current logic-based D-SDA (LD-SDA) variant of D-SDA to solve GDP problems. For instance, the LD-SDA requires the user to provide an initial feasible guess for external variables; otherwise, the algorithm will not converge to a solution[7]. Also, LD-SDA performs an optimization near the initialization of external variables without evaluating other solution candidates that may be more distant from the initialization. The case studies discussed in this work suggest that the proposed LD-BD algorithm performs a more exhaustive exploration of the search space than LD-SDA. For instance, the proposed algorithm can find feasible solutions when initialized from infeasible points. In addition, LD-BD can be "warm started" by evaluating some solution candidates before running the algorithm to increase the quality of the cuts generated during the first iterations of the algorithm. These features are illustrated through computational experiments involving three case studies.

This chapter is organized as follows: section 4.1 states the optimization problems that are within the scope of the LD-BD methodology. Section 4.2 introduces the LD-BD framework, and explains the proposed procedure to generate and update Benders cuts. Section 4.3 illustrates the features of LD-BD through case studies, including a numerical case study, the optimal design of a distillation column, and the optimal design of a rector network.

4.1 Problem statement

The key feature of the GDP problems considered in this work is the presence of ordered Boolean variables $\boldsymbol{Y_R}$ that can be reformulated as external variables $\boldsymbol{e}$. Boolean variables $\boldsymbol{Y_R}$ are used to formulate a disjunctive problem that requires the activation/deactivation of constraints, while integer external variables $\boldsymbol{e}$ recapture the ordered information behind Boolean variables. For example, Boolean variables $\boldsymbol{Y_R} = [Y_1, Y_2, Y_3]$ may indicate the presence/absence of a single feed stream within trays 1, 2 and 3 of a distillation column, whereas the external variable $e \in \{1,2,3\}$ associated to these Boolean variables may represent the actual feed location. The reformulation in this case would be of the form $Y_i = True$ if $e = i$ and $Y_i = False$ otherwise, for all $i \in \{1,2,3\}$. More details about this reformulation are presented in section D.1 of appendix D.

Consider the general optimization problem (4.1), where the objective is to minimize an objective function f. Both continuous ($\boldsymbol{x}$) and Boolean ($\boldsymbol{Y_R}$) variables are present in the formulation; Boolean variables are expressed as a function of external variables ($\boldsymbol{e}$), i.e., notation $\boldsymbol{Y_R}(\boldsymbol{e})$ is used to indicate that Boolean variables $\boldsymbol{Y_R}$ are completely specified when external variables $\boldsymbol{e}$ are fixed (see section D.1 of appendix D). For instance, fixing $e = 2$ implies that Y_2 is $True$, while Y_1 and Y_3 are $False$ in the distillation example described above. The vector of continuous variables ($\boldsymbol{x}$) belongs to a bounded and closed set X, while external variables ($\boldsymbol{e}$) belong to bounded set E, i.e., $\boldsymbol{x} \in X \subset \mathbb{R}^{n_x}$ and $\boldsymbol{e} \in E \subset \mathbb{Z}^{n_e}$, where n_x and n_e represent the number of continuous and external variables, respectively. Boolean variables can take the value of $True$ or $False$, i.e., $\boldsymbol{Y_R} \in \{True, False\}^{n_{Y_R}}$, where n_{Y_R} is the number of Boolean variables. Although Boolean variables $\boldsymbol{Y_R}$ must appear in the formulation to model disjunctions, their values are determined by the values of the external variables $\boldsymbol{e}$ through $\boldsymbol{Y_R}(\boldsymbol{e})$.

$$\min_{\boldsymbol{x},\boldsymbol{e}} f\big(\boldsymbol{x}, \boldsymbol{Y_R}(\boldsymbol{e})\big) \tag{4.1}$$
$$s.t.\ [\boldsymbol{x}, \boldsymbol{Y_R}(\boldsymbol{e})] \in S_{Feasible},\ \boldsymbol{x} \in X \subset \mathbb{R}^{n_x},\ \boldsymbol{e} \in E \subset \mathbb{Z}^{n_e}$$

The problem is constrained such that variables $\boldsymbol{x}$ and $\boldsymbol{Y_R}$ belong to feasible region $S_{Feasible}$. This feasible region is defined in Eq. (4.2) and consists of vectors of nonlinear inequality constraints $\boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}$ and equality constraints $\widehat{\boldsymbol{g}}(\boldsymbol{x}) = \boldsymbol{0}$ that are independent of $\boldsymbol{Y_R}$; a set of disjunctions Q, where each disjunction q contains multiple disjuncts d within set D_q related through an "OR" operator. Each of these disjuncts is associated with a Boolean variable $Y_{d,q}$ that belongs to $\boldsymbol{Y_R}$ and activates/deactivates constraints $\boldsymbol{h}_{d,q}(\boldsymbol{x}) \leq \boldsymbol{0}$ and $\widehat{\boldsymbol{h}}_{d,q}(\boldsymbol{x}) = \boldsymbol{0}$. Additional logical relationships between the Boolean variables are denoted by $\Omega(\boldsymbol{Y_R}) = True$, e.g., exclusive or, implications, etc.

$$S_{Feasible} = \begin{cases} \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0};\ \widehat{\boldsymbol{g}}(\boldsymbol{x}) = \boldsymbol{0} \\ \bigvee_{d \in D_q} \begin{bmatrix} Y_{d,q} \\ \boldsymbol{h}_{d,q}(\boldsymbol{x}) \leq \boldsymbol{0};\ \widehat{\boldsymbol{h}}_{d,q}(\boldsymbol{x}) = \boldsymbol{0} \end{bmatrix}, \forall q \in Q \\ \Omega(\boldsymbol{Y_R}) = True \end{cases} \tag{4.2}$$

The objective function f may represent a combination of process economics, environmental requirements, sustainability incentives, etc. Continuous variables $\boldsymbol{x}$ may include equipment design variables (e.g., distillation column diameter, reactor size), operating variables (e.g., heat duties, manipulated flows), mass and energy balance variables (e.g., compositions, temperatures and flows).

Chemical engineering examples of external variables $\boldsymbol{e}$ include: the number of stages in a multi-effect evaporator sequence[269]; the number of dosing segments in a load-flexible fixed bed reactor[270]; the number of separators in a multi-stage flash desalinization process[271]; feed locations, side stream locations, catalyst locations and number of stages in conventional and intensified distillation/reactor networks[272,273]. The feasible region $S_{Feasible}$ comprises multiple constraints including: the mass and energy balances of the systems, product specification constraints, hydrodynamic constraints, geometric relationships between design variables, etc.

4.2 Mathematical framework

The approach proposed in this work to solve problem (4.1) is based on a master-subproblem decomposition, with Benders cuts specifically designed for problems involving external variables. Section 4.2.1 provides an overview of the standard LBBD method and the corresponding master-subproblem decomposition, which serve as background for the developments presented in this work. Sections 4.2.2 and 4.2.3 describe the strategy to obtain the Benders cuts developed in this work whereas section 4.2.4 presents the implementation of this cutting strategy within the proposed LD-BD method.

4.2.1 Background and problem decomposition

The key idea of LBBD strategies is that fixing some complicating variables in an optimization problem results in a smaller and easier to solve problem called the *subproblem*. The decomposition of problem (4.1) results in a subproblem (Eq. (4.3)) and a master problem (Eq. (4.4)) that are solved in an iterative fashion. The variables $\boldsymbol{e}$ and $\boldsymbol{x}$ represent the complicating and non-complicating variables, respectively. Note that non-complicating variables $\boldsymbol{x}$ do not appear in the master problem formulation, while the subproblem keeps $\boldsymbol{e}$ fixed. When complicating variables $\boldsymbol{e}$ are fixed by the master problem, the resulting NLP subproblem is easier to solve than the original MINLP. Thus, the subproblem optimizes the non-complicating variables with variables $\boldsymbol{e}$ and $\boldsymbol{Y_R}(\boldsymbol{e})$ fixed, and generates a new Benders cut at each iteration, while the master problem optimizes external variables $\boldsymbol{e}$ subject to the cuts generated up until that iteration step. Variables in $\boldsymbol{Y_R}$ remain fixed in the subproblems, however they must appear in Eq. (4.3). This is because Boolean variables in $\boldsymbol{Y_R}$ are required to decide which disjuncts will remain active/inactive in each subproblem, and to verify the feasibility of the logical relationships within $S_{Feasible}$. The master problem and the subproblem are solved iteratively, with iterations denoted by superscript k. In Eq. (4.3), $f^*(\boldsymbol{e}^k)$ is the optimal objective function value of the subproblem, with $\boldsymbol{e}$

fixed at $\boldsymbol{e}^k$. In the master problem (Eq. (4.4)), the key idea of LBBD is to represent the original objective function with an optimization variable $z \in \mathbb{R}$, which is bounded below by the Benders cuts obtained until iteration k, i.e., $\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}), \forall\, \hat{k} \in \{1,2,\dots,k\}$. For example, $\beta_{\boldsymbol{e}^k}(\boldsymbol{e})$ is the cut obtained after solving the subproblem for trial value $\boldsymbol{e}^k$. The algorithm converges when the objectives from the master problem and the subproblem converge to the same value[9].

$$f^*\left(\boldsymbol{e}^k\right) = \min_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{Y_R}(\boldsymbol{e}^k)) \tag{4.3}$$
$$s.t.\ [\boldsymbol{x}, \boldsymbol{Y_R}(\boldsymbol{e}^k)] \in S_{Feasible}, \boldsymbol{x} \in X \subset \mathbb{R}^{n_x}$$

$$[z^{k+1}, \boldsymbol{e}^{k+1}] = \arg\min_{z,\boldsymbol{e}} z \tag{4.4}$$
$$s.t.\ z \geq \beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e})\ \forall\, \hat{k} \in \{1,2,\dots,k\}, \boldsymbol{e} \in E \subset \mathbb{Z}^{n_e}, z \in \mathbb{R}$$

The standard LBBD algorithm proceeds as follows: 1) At $k = 1$ choose a feasible $\boldsymbol{e}^k \in E$ as initialization and set $z^k = -\infty$, 2) Solve subproblem (4.3) with optimal solution $f^*(\boldsymbol{e}^k)$, if subproblem (4.3) is infeasible, then set $f^*\left(\boldsymbol{e}^k\right) = +\infty$, 3) If $\min\left\{f^*\left(\boldsymbol{e}^{\hat{k}}\right): \hat{k} \in \{1,2,\dots,k\}\right\} > z^k$ formulate a cut $\beta_{\boldsymbol{e}^k}(\boldsymbol{e})$, add it to the master problem and proceed to Step 4; otherwise, stop: the optimal solution of problem (4.1) has been found, i.e., z^k, 4) Solve the master problem (4.4). 5) If an optimal solution $[z^{k+1}, \boldsymbol{e}^{k+1}]$ was found from problem (4), update $k = k + 1$ and go back to Step 2, otherwise, stop: problem (1) is infeasible[9]. This standard LBBD method requires the user to provide feasible values for the complicating variables ($\boldsymbol{e}^1$). If this is not the case, i.e., $\boldsymbol{e}^1$ is infeasible, a feasibility version of the LBBD algorithm can be executed. The main distinction between the feasibility version of LBBD and the standard LBBD is that the Benders cuts are replaced by feasibility cuts, i.e., constraints that are violated when $\boldsymbol{e}^k$ takes infeasible values[10].

The standard LBBD algorithm will be used as a basis to develop the proposed LD-BD strategy, which introduces enhancements required to address the potentially non-convex problems considered in this study (see section 4.2.3). Cuts $\beta_{\boldsymbol{e}^k}(\boldsymbol{e})$ must be developed specifically for each application and no general equation is available to obtain those cuts[9]. Hence, LBBD can be seen as a framework to create ad-hoc optimization strategies that exploit the problem's structure. The following section explains the strategy developed to obtain cuts in this study.

4.2.2 Generation of Benders cuts

As described above, the general LBBD framework imposes a requirement with respect to traditional Benders based methods, i.e., specialized cuts must be developed for each application. There are different strategies to obtain Benders cuts where the most common approach consists of using the subproblem's dual information[9], e.g., BD uses classical duality for LP[274], while GBD uses Lagrangean duality for convex NLP problems[56]. A major limitation of the available cut generation strategies is that they do not consider the ordered nature of external variables and the optimality guarantees that can be exploited from ordered discrete decisions. To address this issue, a new cut generation strategy is presented in this section. By convention, assume that infeasible solutions of subproblems return an objective value of $+\infty$. Thus, function $f^*: \mathbb{Z}^{n_e} \mapsto \mathbb{R} \cup \{+\infty\}$ in Eq. (4.3) takes the optimal value of the subproblem for feasible values of $\boldsymbol{e}$ and $+\infty$ otherwise. Objective function values of $-\infty$ are not allowed, meaning that subproblems are not unbounded. Also, assume that the effective domain of f^* ($\mathrm{dom}_{\mathbb{Z}}(f^*)$) is nonempty, i.e., there is at least one feasible subproblem with a finite objective function value.

Two important guidelines that should be considered when generating Benders cuts are shown in Eqs. (4.5) and (4.6). Eq. (4.5) indicates that the Benders cuts obtained until iteration $\hat{k}$, i.e., $\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}), \forall\, \hat{k} \in \{1,2,\dots,k\}$, must be valid. As it is shown in Eq. (4.5), a valid cut $\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e})$ must underestimate the epigraph of the objective function of the problem. On the other hand, Eq. (4.6) indicates that Benders cuts must be tight lower bounding estimators of the subproblems' optimal objectives (i.e., f^*) for $\boldsymbol{e} = \boldsymbol{e}^{\hat{k}}, \forall \hat{k} \in \{1,2,\dots,k\}$. Generating cuts that satisfy Eqs. (4.5) and (4.6) allows to guarantee termination and convergence of the generic LBBD algorithm to a globally optimal solution of problem (4.1)[9].

$$\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}) \leq f\big(\boldsymbol{x}, \boldsymbol{Y_R}(\boldsymbol{e})\big), \forall \boldsymbol{x} \in X \subset \mathbb{R}^{n_x}, \forall \boldsymbol{e} \in E \subset \mathbb{N}^{n_e}, \forall\, \hat{k} \in \{1,2,\dots,k\} \tag{4.5}$$

$$\beta_{\boldsymbol{e}^{\hat{k}}}\big(\boldsymbol{e}^{\hat{k}}\big) = f^*\big(\boldsymbol{e}^{\hat{k}}\big), \forall\, \hat{k} \in \{1,2,\dots,k\} \tag{4.6}$$

Considering the above criteria, an ideal strategy to obtain Benders cuts for problems with external variables would be using the convex envelope of f^* to derive cuts that satisfy Eqs. (4.5) and (4.6). The convex envelope of f^* is denoted as $\bar{f}^*$ and it is a convex function that tightly under-estimates f^*. It is possible to calculate $\bar{f}^*$ at $\boldsymbol{e}^{\hat{k}}$ through the Linear Programming (LP) optimization problem (4.7), where

notation $\langle \boldsymbol{p}, \boldsymbol{e} \rangle$ is used to represent the dot product between $\boldsymbol{p}$ and $\boldsymbol{e}$[6]. This constrained LP problem optimizes the normal vector $\boldsymbol{p}$ and the offset α of a cutting plane evaluated at $\boldsymbol{e}^{\hat{k}}$. Note that a complete enumeration of $f^*(\boldsymbol{e})$ is required to define the constraints in problem (4.7). The solution of the LP in problem (4.7) allows to obtain a cut defined as $\langle \boldsymbol{p}^{\hat{k}}, \boldsymbol{e} \rangle + \alpha^{\hat{k}}$ through Eq. (4.8). Hence, $\boldsymbol{p}^{\hat{k}}$ and $\alpha^{\hat{k}}$ in Eq. (4.8) represent the optimal solution of problem (4.7) and define a cutting plane $\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}) = \langle \boldsymbol{p}^{\hat{k}}, \boldsymbol{e} \rangle + \alpha^{\hat{k}}$ that underestimates f^* for any value of $\boldsymbol{e}$. In particular, cut $\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e})$ is equal to the convex envelope $\bar{f}^*(\boldsymbol{e})$ when $\boldsymbol{e} = \boldsymbol{e}^{\hat{k}}$.

Benders cuts obtained through Eqs. (4.7) and (4.8) can be assessed by analyzing the convexity properties of f^*. Function $f^*: \mathbb{Z}^{n_e} \mapsto \mathbb{R} \cup \{+\infty\}$ is *integrally convex* if $\tilde{f}^*: \mathbb{R}^{n_e} \mapsto \mathbb{R} \cup \{+\infty\}$ is convex, where $\tilde{f}^*$ is the local convex extension of f^* (the mathematical definition of $\tilde{f}^*$ is available in section D.2 of appendix D)[6]. Under the assumption that f^* is integrally convex, cuts obtained through Eqs (4.7) and (4.8) would satisfy Eqs. (4.5) and (4.6), i.e., these cuts would be valid and guarantee convergence to a globally optimal solution. This notion is formally stated in the following theorem.

$$\bar{f}^*\left(\boldsymbol{e}^{\hat{k}}\right) = \max_{\boldsymbol{p} \in \mathbb{R}^{n_e}, \alpha \in \mathbb{R}} \left\{ \langle \boldsymbol{p}, \boldsymbol{e}^{\hat{k}} \rangle + \alpha : \langle \boldsymbol{p}, \boldsymbol{e} \rangle + \alpha \leq f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in \mathbb{Z}^{n_e} \right\} \tag{4.7}$$

$$\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}) = \langle \boldsymbol{p}^{\hat{k}}, \boldsymbol{e} \rangle + \alpha^{\hat{k}}, \left(with \left[\boldsymbol{p}^{\hat{k}}, \alpha^{\hat{k}} \right] \in \arg\max \bar{f}^*\left(\boldsymbol{e}^{\hat{k}}\right) \right) \tag{4.8}$$

Theorem 4.1. *Assume that* $f^*: \mathbb{Z}^{n_e} \mapsto \mathbb{R} \cup \{+\infty\}$ *is an integrally convex function. Then, Benders cut* $\langle \boldsymbol{p}^{\hat{k}}, \boldsymbol{e} \rangle + \alpha^{\hat{k}}$ *obtained from* Eq. (4.8) *satisfy* Eqs. (4.5) *and* (4.6).

The proof of this theorem is presented in section D.2 of appendix D. Despite their theoretical properties, Eqs. (4.7) and (4.8) only serve as a starting point to obtain Benders cuts for the general decomposed problems shown in Eqs. (4.3) and (4.4). This is because using Eqs. (4.7) and (4.8) requires knowledge of f^* over the feasible region of the external variables, i.e., it would be equivalent to a complete enumeration. Thus, Theorem 4.1 does not offer a practical method to compute Benders cuts at each iteration of the Benders algorithm. To partially address this issue, Benders cuts are herein approximated using local neighborhood information of the external variables, i.e., in a similar way gradients provide neighborhood information for continuous variables. There are many alternatives to define a neighborhood for integer decisions such as external variables. The integrally convex

neighborhood and its subsets, e.g., the separable convex, the $L^\natural$-convex and the $M^\natural$-convex neighborhoods, are commonly used in the literature[6,275]. Without loss of generality, the integrally convex neighborhood defined as $N_\infty(\boldsymbol{e}^k) = \{\boldsymbol{e} \in \mathbb{Z}^{n_e} : \|\boldsymbol{e} - \boldsymbol{e}^k\|_\infty \leq 1\}$ is considered in this study. When considering neighborhood $N_\infty(\boldsymbol{e}^k)$, having $f^*(\boldsymbol{e}^k) \leq f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in N_\infty(\boldsymbol{e}^k)$ implies that $\boldsymbol{e}^k$ is a local optimum with respect to $N_\infty(\boldsymbol{e}^k)$[3,4]. Note that the neighborhood $N_\infty(\boldsymbol{e}^k)$ grows exponentially as the number of external variables in $\boldsymbol{e}^k$ increases; thus, evaluating f^* over $N_\infty(\boldsymbol{e}^k)$ may be unsuitable if computational resources are limited. In those cases, smaller neighborhoods such as the separable convex neighborhood can be considered, i.e., $N_2(\boldsymbol{e}^k) = \{\boldsymbol{e} \in \mathbb{Z}^{n_e} : \|\boldsymbol{e} - \boldsymbol{e}^k\|_2 \leq 1\}$. There may be a trade-off between computational costs and the quality of the solution obtained depending on the neighborhood selected, e.g., a solution strategy that implements the $N_2(\boldsymbol{e}^k)$ neighborhood may converge faster than its equivalent version with the $N_\infty(\boldsymbol{e}^k)$ neighborhood, at the expense of potentially ignoring solution candidates within $N_\infty(\boldsymbol{e}^k)$ with better objective function values. Alternative reformulation strategies to improve the solution quality have been proposed in the literature[4]. In the present work, Benders cuts are calculated using information from the current subproblem at $\boldsymbol{e}^k$ and its integrally convex neighborhood $N_\infty(\boldsymbol{e}^k)$. Adapting the developments presented through this work to other neighborhoods, e.g., subsets of N_∞, is straightforward and not shown here for brevity. Note that the selection of the neighborhood search method does not affect the convergence properties of the algorithm discussed in section 4.2.4.

Using local neighborhood information implies that the algorithm presented in section 4.2.4 only guarantees local optimality with respect to the specified neighborhood of external variables. A naïve approach to obtain Benders cuts based on neighborhood information would be to replace $\mathbb{Z}^{n_e}$ by $N_\infty(\boldsymbol{e}^{\hat{k}})$ in Eq. (4.7); however, this strategy does not guarantee the identification of valid Benders cuts. A strategy to iteratively refine Benders cuts is presented next.

4.2.3 Iterative refinement of Benders cuts

Valid Benders cuts should not exclude feasible solutions of the search space. If cuts are not valid, the algorithm may stop in a sub-optimal solution that may not even be locally optimal, since optimal solutions may lie in a wrongly excluded region. Thus, generating valid cuts through the solution procedure is the keystone of traditional Benders decomposition algorithms. Note that the solution of large-scale case studies with non-valid Benders cuts is an approach that has been used in the literature

to accelerate the algorithm, at the expense of sacrificing optimality guarantees [10,276]. To partially overcome the optimality issue, a promising approach that has been previously used in the literature consists of correcting previously generated cuts as the iteration proceeds, such that all previously generated cuts remain valid. This dynamic refinement in the Benders cuts has allowed to obtain proven optimal solutions for MIPs and MINLPs [264,277]. To take advantage of this feature, a refinement strategy is proposed in this work. The refinement procedure guarantees that, when the Benders algorithm stops, the solution obtained is a locally optimal solution.

The proposed refinement procedure aims to guarantee that, at each iteration, every cut is valid with respect to every external variable value previously evaluated. To do that, I introduce set D^k, which is defined as every fixed value of external variables that has been evaluated prior and up to iteration k. D^0 must be defined before starting the algorithm and can be initialized as an empty set $D^{init} = \emptyset$ by default (see Eq. (4.9)). Alternatively, external variables for which f^* is known prior starting the algorithm can be added to D^{init}; this can be interpreted as a *warm start* of the algorithm[10]. After its initialization, D^k will be recursively defined as indicated in Eq. (4.10). If the subproblem (4.3) at iteration k is feasible (i.e., $f^*(\boldsymbol{e}^k) < +\infty$), the subproblems within a neighborhood $N_\infty(\boldsymbol{e}^k)$ of $\boldsymbol{e}^k$ are solved, and the variable values within $N_\infty(\boldsymbol{e}^k)$ (with $\boldsymbol{e}^k$ included) are added to D^k. Alternatively, when the subproblem is infeasible (i.e., $f^*(\boldsymbol{e}^k) = +\infty$), it is not necessary to evaluate f^* over the neighborhood, because an infeasible subproblem at $\boldsymbol{e}^k$ already indicates that the master problem should move away from $\boldsymbol{e}^k$. Note that the objective function value f^* of external variables in D^k should also be stored to avoid the solution of the same subproblem multiple times. Based on this definition of D^k, expressions that allow to compute Benders cuts for problems with external variables are given in Eqs. (4.11) and (4.12).

$$D^0 = D^{init} \tag{4.9}$$

$$D^k = \begin{cases} D^{k-1} \cup N_\infty(\boldsymbol{e}^k), & f^*(\boldsymbol{e}^k) < +\infty \\ D^{k-1} \cup \{\boldsymbol{e}^k\}, & f^*(\boldsymbol{e}^k) = +\infty \end{cases} \tag{4.10}$$

$$\left[\boldsymbol{p}^{\hat{k}}, \alpha^{\hat{k}}\right] = \arg \max_{\boldsymbol{p} \in \mathbb{R}^{n_e}, \alpha \in \mathbb{R}} \left\{ \langle \boldsymbol{p}, \boldsymbol{e}^{\hat{k}} \rangle + \alpha : \langle \boldsymbol{p}, \boldsymbol{e} \rangle + \alpha \leq f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in D^k \right\}, \forall\, \hat{k} \in \{1,2,\dots,k\} \tag{4.11}$$

$$\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}) = \langle \boldsymbol{p}^{\hat{k}}, \boldsymbol{e} \rangle + \alpha^{\hat{k}}, \forall\, \hat{k} \in \{1,2,\dots,k\} \tag{4.12}$$

Based on the above, the generation of cuts at each iteration of the LD-BD algorithm that will be presented in section 4.2.4 involves the steps detailed in Table 4.1.

Table 4.1. Steps required to obtain benders cuts at iteration k

Step 1. Feasibility verification If $f^*(\boldsymbol{e}^k) < +\infty$, solve the subproblem in Eq. (4.3) for every $\boldsymbol{e}$ within $N_\infty(\boldsymbol{e}^k)$.
Step 2. Updating D^k If $k = 1$, let $D^0 = D^{init}$ (see Eq. (4.9)). If $f^*(\boldsymbol{e}^k) < +\infty$, add every element of $N_\infty(\boldsymbol{e}^k)$ to D^k and store $f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in N_\infty(\boldsymbol{e}^k)$. Else, add $\boldsymbol{e}^k$ to D^k and store $f^*(\boldsymbol{e}^k)$ (see Eq. (4.10)).
Step 3. Refinement of Benders cuts For every $\hat{k}$ in $\{1,2,\dots,k\}$, solve the optimization problem in Eq. (4.11). Then use the optimal solution found to obtain $\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e})$ using Eq. (4.12).

4.2.4 The Logic-Based Discrete-Benders Decomposition (LD-BD) algorithm

Based on the cutting strategy and the refinement procedure shown in the previous section, a Logic-Based Discrete-Benders Decomposition algorithm (LD-BD) to solve problems with external variables as complicating variables is presented in Figure 4.1. This algorithm offers an alternative to solve the problem stated in Eq. (4.1) by decomposing it into an NLP subproblem that contains nonlinear constraints and optimizes continuous variables (see Eq. (4.3)), and a MIP master problem that optimizes external variables (see Eq. (4.4)).

The LD-BD algorithm requires external variables and set D^0 to be initialized at $\boldsymbol{e}^{init}$ and D^{init}, respectively. For the initialization of external variables, $\boldsymbol{e}^{init}$ can be feasible or infeasible. When $\boldsymbol{e}^{init}$ is feasible, the algorithm proceeds as depicted in Figure 4.1. Conversely, a preliminary task needs to be performed when $\boldsymbol{e}^{init}$ is infeasible. A preliminary procedure that allows to find a feasible combination of external variables $\boldsymbol{e}^{init}$ before proceeding to step 1 in Figure 4.1 is discussed in section D.3 of appendix D. In that procedure, the LD-BD method is used to find a feasible combination of external variables through the minimization of an objective function that quantifies the infeasibility of the model. The specific steps of the proposed LD-BD method are discussed below.

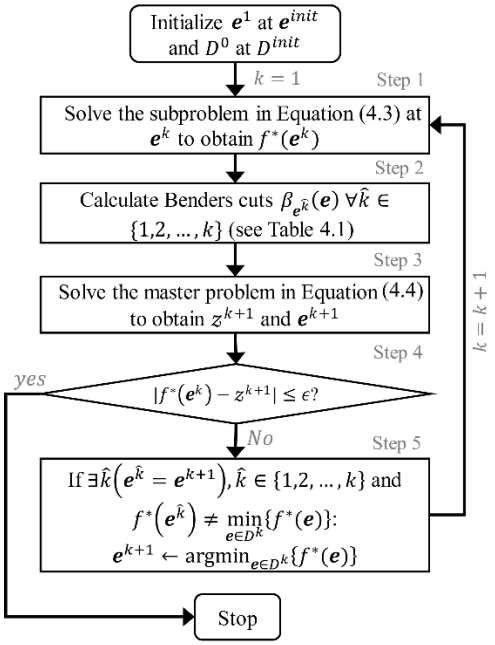


Figure 4.1. The Logic-Based Discrete-Benders Decomposition algorithm (LD-BD).

As shown in Figure 4.1, the external variables are fixed at $\boldsymbol{e}^k$ in step 1, e.g., in the first iteration (*k=1*) $\boldsymbol{e}^k = \boldsymbol{e}^{init}$. Then, the resulting subproblem in Eq. (4.3) is solved, while keeping $\boldsymbol{e}^k$ fixed. The objective function from this subproblem (i.e., $f^*(\boldsymbol{e}^k)$) is then used as an input to step 2, where Benders cuts from previous iterations are recalculated and a new Benders cut is obtained, i.e., cuts $\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e})\ \forall \hat{k} \in \{1,2,\dots,k-1\}$ are updated, and a new cut $\beta_{\boldsymbol{e}^k}(\boldsymbol{e})$ is obtained. The steps required to obtain these cuts are described in Table 4.1. Step 3 uses the Benders cuts $\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e})\ \forall \hat{k} \in \{1,2,\dots,k\}$ obtained in step 2 to solve the master problem depicted in Eq. (4.4). The solution of this master problem specifies trial values of the master problem objective (z^{k+1}) and external variables values ($\boldsymbol{e}^{k+1}$) for the next iteration. Step 4 compares the objective function values from the master problem (z^{k+1} from Eq. (4.4)) and the subproblem ($f^*(\boldsymbol{e}^k)$ from Eq. (4.3)). z^{k+1} and $f^*(\boldsymbol{e}^k)$ provide lower and upper bounds for the objective function of the original problem in Eq. (1), respectively. If the gap between these bounds has been closed within an ϵ-tolerance, the algorithm stops and returns a locally optimal solution; otherwise, the algorithm proceeds to step 5. The purpose of step 5 is to guarantee the convergence of the LD-BD method in a finite number of iterations, by preventing the algorithm from revisiting the same combination of external variables indefinitely. This situation may occur when $\boldsymbol{e}^{k+1}$ is equal to a trial value $\boldsymbol{e}^{\hat{k}}$ explored in previous iterations (i.e., $\exists \hat{k}\left(\boldsymbol{e}^{\hat{k}} = \boldsymbol{e}^{k+1}\right)$, with $\hat{k} \in \{1,2,\dots,k\}$) and the

subproblem objective of $\boldsymbol{e}^{\hat{k}}$ is not optimal within D^k (i.e., $f^*(\boldsymbol{e}^{\hat{k}}) \neq \min_{\boldsymbol{e}\in D^k}\{f^*(\boldsymbol{e})\}$). If this situation occurs, $\boldsymbol{e}^{k+1}$ is updated with the best solution of the problem obtained thus far, i.e., $\boldsymbol{e}^{k+1} \leftarrow \arg\min_{\boldsymbol{e}\in D^k}\{f^*(\boldsymbol{e})\}$. Then, a new iteration begins, and the algorithm returns to step 1.

The proposed LD-BD algorithm refines Benders cuts (step 2) before evaluating the stopping criterion of candidate solution $\boldsymbol{e}^k$ (step 4), which is a key feature introduced by the proposed method. Another novelty is the addition of step 5, which guarantees that the algorithm eventually terminates in a finite number of iterations. Overall, these improvements allow the application of LD-BD to optimization problems involving chemical processes, which are typically described by non-linear equations. Notably, if these enhancements are not considered, local optimality guarantees may not be ensured for the solution obtained when dealing with non-convex optimization problems often encountered in superstructure optimization, i.e., when LD-BD converges, local optimality is ensured for the solution obtained. These properties are formally defined in Theorem 4.2.

Theorem 4.2. *Assume that an absolute optimality gap $\epsilon = 0$ is stablished, $\boldsymbol{e}^{init}$ returns a feasible solution for the problem, the master problem is always solved to optimality, none of the subproblems has an unbounded objective function, and the feasible region of external variables E is bounded, then the LD-BD algorithm stops in a finite number of iterations, and the solution obtained is feasible and locally optimal.*

Proof of Theorem 4.2 is shown in section D.2 of appendix D. Under the assumptions stated in the theorem, LD-BD guarantees local optimality, i.e., the optimality of non-complicating variables in $\boldsymbol{x}$ is dictated by the subproblem solver, while complicating variables in $\boldsymbol{e}$ are guaranteed to be optimal within neighborhood $N_\infty(\boldsymbol{e})$. Despite being a local strategy, the LD-BD algorithm may converge to the global optimum if other requirements are met. In general, global optimality can be achieved if all the locally optimal solutions of the problem share the same objective function value. This situation may be encountered when the function that returns the optimal objective values of the subproblems, i.e., $f^*(\boldsymbol{e})$, is integrally convex. The integral convexity of $f^*(\boldsymbol{e})$ is related to the convexity properties of the effective domain of $f^*(\boldsymbol{e})$ ($\mathrm{dom}_{\mathbb{Z}}(f^*)$). The effective domain is the subset of the search space of external variables E where $f^*(\boldsymbol{e})$ takes finite values, i.e., the optimization problem is neither infeasible nor unbounded within $\mathrm{dom}_{\mathbb{Z}}(f^*) \subseteq E$. To understand this relationship, first consider the definition of an *integrally convex set*: take a finite set $Z \subset \mathbb{Z}^{n_z}$, define $\mathrm{co}(Z)$ as the convex hull of Z and define the

discrete neighborhood of a real-valued point $\boldsymbol{y} \in \mathbb{R}^{n_e}$ as $N(\boldsymbol{y}) = \{\boldsymbol{z} \in \mathbb{Z}^{n_e} : \|\boldsymbol{z} - \boldsymbol{y}\|_\infty < 1\}$. If for every $\boldsymbol{y} \in \mathbb{R}^{n_e}$ having $\boldsymbol{y} \in \text{co}(Z)$ implies that $\boldsymbol{y} \in \text{co}\big(Z \cap N(\boldsymbol{y})\big)$, then Z is integrally convex[278], e.g., a discrete set given by upper and lower bounds over its elements (e.g., $Z = \{\boldsymbol{e} = [e_1, e_2] \in \mathbb{Z}^2 : 1 \leq e_1 \leq 4{,}5 \leq e_2 \leq 8\}$) is integrally convex, while set $Z = \{[1{,}1], [5{,}5]\}$ contains two disjoint points that are distant from each other and do not define an integrally convex region. The key relationship between $f^*(\boldsymbol{e})$ and its effective domain $\text{dom}_{\mathbb{Z}}(f^*) \subseteq E$ is that if $\text{dom}_{\mathbb{Z}}(f^*)$ is not integrally convex, then $f^*(\boldsymbol{e})$ is not integrally convex and global optimality cannot be ensured by the proposed LD-BD method[6], i.e., $\text{dom}_{\mathbb{Z}}(f^*)$ must necessarily be integrally convex for $f^*(\boldsymbol{e})$ to be integrally convex.

The attributes of LD-BD with respect to LD-SDA are discussed below. On the one hand, LD-BD and LD-SDA share some similarities, e.g., they both explore external variables using neighborhood information. Similar to the LD-SDA, the worst-case performance of LD-BD occurs when a complete enumeration of all the combinations of external variables is performed. Although a complete enumeration was not observed in the case studies considered in this work, this condition may occur when the external variables are constrained to decide between two possible values, e.g., when the external variables are constrained to take binary values. On the other hand, the LD-BD strategy differs from LD-SDA in different ways. While the LD-SDA performs an iterative neighborhood search nearby $\boldsymbol{e}^{init}$, the master problem in LD-BD may return solutions that are far away from $\boldsymbol{e}^{init}$ in the early iterations of the algorithm for which only a few cuts have been generated. In some cases, this may represent a disadvantage of the LD-BD algorithm with respect to the LD-SDA in terms of computational time, given that LD-BD may require solving more subproblems during the search. For instance, if both LD-SDA and LD-BD are initialized from a locally optimal solution, LD-SDA will stop in a single iteration whereas LD-BD may explore other candidate solutions before concluding that the user-provided initialization is optimal. Note that in some cases this may also represent an advantage for LD-BD since the initial exploration may detect another locally optimal solution with a better objective function, i.e., it may move away from local optima. Another advantage of the proposed algorithm with respect to the LD-SDA is that LD-BD implements a strategy to find feasible solutions when the problem is initialized from an infeasible point $\boldsymbol{e}^{init}$. Furthermore, the LD-BD algorithm allows D^0 to be initialized with external variables' values for which f^* is known. In the ideal case where D^0 is equal to the feasible region of external variables (i.e., $D^0 = E$) and f^* is known over E, the algorithm will converge to the global optimum of the problem because this would be equivalent to imposing the ideal Benders cuts defined by Eqs. (4.7) and (4.8). Thus, a pre-processing step that samples the feasible

region of external variables and computes some benders cuts before running the algorithm would increase the likelihood of finding the globally optimal solution of the problem. All these features of the LD-BD algorithm are illustrated in the following section.

4.3 Computational experiments

The main characteristics and features of the proposed LD-BD algorithm are illustrated through three case studies. The first case study is an introductory mathematical example that illustrates how the LD-BD algorithm optimizes the discrete decisions of the problem. This first case study also compares the performance of this method against the previously proposed LD-SDA. The second case study considers the optimal design of a distillation column using a GDP model previously proposed in the literature. The main challenge of this problem is that the solution obtained may depend on the initial guess of discrete decisions when using previously developed local logic-based strategies such as L-OA and LD-SDA whereas the proposed LD-BD strategy is shown to consistently return the best solution known for this case study. The third case study considers the optimal design of a reactor network, which is used to illustrate the benefits of using warm-starts combined with the proposed LD-BD algorithm to find solutions with a better objective function value.

The experiments were implemented in Pyomo in a PC Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz with 16 GB RAM. The following solvers were used for all the runs of solvers LD-BD, LD-SDA, and L-OA through this section. CPLEX was used to solve the master problems, except for LD-SDA which does not require a MIP solver; likewise, MSNLP was used for the NLP subproblems, except for the second case study for which solver KNITRO showed better convergence performance for all logic-based solvers. Also, an absolute optimality gap $\epsilon = 0$ was imposed for all the case studies considered in this work.

4.3.1 Case study 1: Mathematical illustrative example

The GDP optimization problem described through Eqs. (4.13)-(4.16), with sets $I_1 = \{1,2,3,4,5\}$ and $J_2 = \{1,2,3,4,5\}$ is considered in this section. This problem was adapted from the Gomez and Levy bivariate nonlinear optimization test problem[279]. This case study considers the minimization of objective function f in Eq. (4.13), which depends on continuous variables α_1 and β_2, i.e., $\boldsymbol{x} = [\alpha_1, \beta_2]$. The feasible region of these variables is given by $\boldsymbol{x} \in X = \{[\alpha_1, \beta_2] \in \mathbb{R}^2 : -0.1 \leq \alpha_1 \leq 0.3, -0.9 \leq \beta_2 \leq -0.5\}$. The value of these variables is determined based on the value of $\boldsymbol{Y_R} = [\boldsymbol{Y_1}, \boldsymbol{Y_2}]$, with

Boolean vectors $\boldsymbol{Y}_1 = [Y_{1,1}, Y_{1,2}, \dots, Y_{1,5}]$ and $\boldsymbol{Y}_2 = [Y_{2,1}, Y_{2,2}, \dots, Y_{2,5}]$. The feasible region $S_{Feasible}$ is defined by disjunctions in Eqs. (4.14) and (4.15) and the logical constraints in Eq. (4.16). The reformulation with external variables $\boldsymbol{e} = [e_1, e_2]$ (i.e., $\boldsymbol{Y_R}(\boldsymbol{e})$) is shown in Eq. (4.17) and (4.18), which define the search space of external variables as a bounded discrete grid, i.e., $\boldsymbol{e} \in E = \{[e_1, e_2] \in \mathbb{Z}^2 : 1 \leq e_1 \leq 5, 1 \leq e_2 \leq 5\}$.

$$\min_{\alpha_1, \beta_2, \boldsymbol{Y_1}, \boldsymbol{Y_2}} f = 4\alpha_1^2 - 2.1\alpha_1^4 + \frac{1}{3}\alpha_1^6 + \alpha_1\beta_2 - 4\beta_2^2 + 4\beta_2^4 \tag{4.13}$$

$$s.t. \underset{i_1 \in I_1}{\vee} \begin{bmatrix} Y_{1,i_1} \\ \alpha_1 = -0.1 + 0.1(i_1 - 1) \end{bmatrix} \tag{4.14}$$

$$\underset{j_2 \in J_2}{\vee} \begin{bmatrix} Y_{2,j_2} \\ \beta_2 = -0.9 + 0.1(j_2 - 1) \end{bmatrix} \tag{4.15}$$

$$\Omega = \left\{ \underset{i_1 \in I_1}{\veebar} Y_{1,i_1}, \underset{j_1 \in J_1}{\veebar} Y_{2,j_1}, \neg Y_{1,3} \right\} = True \tag{4.16}$$

$$Y_{1,i_1} = \begin{cases} True, e_1 = i_1 \\ False, e_1 \neq i_1 \end{cases}, \forall i_1 \in \{1,2,3,4,5\}, e_1 \in \{1,2,3,4,5\} \tag{4.17}$$

$$Y_{2,j_2} = \begin{cases} True, e_2 = j_2 \\ False, e_2 \neq j_2 \end{cases}, \forall j_2 \in \{1,2,3,4,5\}, e_2 \in \{1,2,3,4,5\} \tag{4.18}$$

The problem was first solved using $\boldsymbol{e}^1 = \boldsymbol{e}^{init} = [5,1]$ and $D^0 = \emptyset$, as an illustrative example to show the step-by-step implementation of the LD-BD method. Figure 4.2 shows iterations $k = 1$, $k = 2$ and the last iteration ($k = 7$) of the algorithm. The results of the first iteration ($k = 1$) of the problem are shown in Figures 4.2A, 4.2B and 4.2C. The solution of the first subproblem at $\boldsymbol{e}^1$ is feasible with objective value $f^*(e_1) = -0.542$ (step 1 in Figure 4.1). The subproblem is then solved within neighborhood $N_\infty(\boldsymbol{e}^1) = \{[5,1], [5,2], [4,1], [4,2]\}$, represented graphically through green circles in Figures 4.2A and 4.2B. The solutions of all subproblems within $N_\infty(\boldsymbol{e}^1)$ are then used as inputs to Eqs. (4.10)-(4.12) with $D^1 = N_\infty(\boldsymbol{e}^1)$, which return the first cut: $\beta_{\boldsymbol{e}^1}(\boldsymbol{e}) = 0.096e_1 - 0.286e_2 - 0.739$ (step 2 in Figure 4.1). This cut is represented through the red plane Figure 4.2B. Then, the master problem in Eq. (4.4) is solved, returning the magenta square in Figures 4.2A and 4.2B with external variables $\boldsymbol{e}^2 = [1,5]$ and objective $z^2 = -2.073$ (step 3 in Figure 4.1). The gap between the master

problem and the subproblem is shown in Figure 4.2C (step 4 in Figure 4.1). Since $\boldsymbol{e}^2$ is an unexplored trial vector of external variables, step 5 in Figure 4.1 is skipped in this iteration. Vector $\boldsymbol{e}^2$ is used as the new incumbent solution for the next iteration $k = 2$, represented graphically in Figures 4.2D, 4.2E and 4.2F. This new iteration $k = 2$ considers information from the previous iteration $k = 1$. For instance, $D^2 = N_\infty(\boldsymbol{e}^2) \cup N_\infty(\boldsymbol{e}^1)$ considers subproblems evaluated in the current and previous iterations. Similarly, the red surface in Figure 4.2E represents the most recent cut $\beta_{\boldsymbol{e}^2}(\boldsymbol{e})$, and an updated version of the previously generated cut $\beta_{\boldsymbol{e}^1}(\boldsymbol{e})$. As iterations proceed, new points are added to D^k to increase the quality of the Benders cuts. The LD-BD algorithm keeps iterating until iteration $k = 7$ depicted in Figures 4.2G, 4.2H and 4.2I, when the gap between the objective functions from the master and the subproblem closes, and the solution given by discrete variables $\boldsymbol{e}^7 = [2,3]$ and continuous variables $\alpha_1 = 0$ and $\beta_2 = -0.7$ has local optimality characteristics, i.e., $f^*(\boldsymbol{e}^7) = -0.999$ is minimum within $N_\infty(\boldsymbol{e}^7)$.

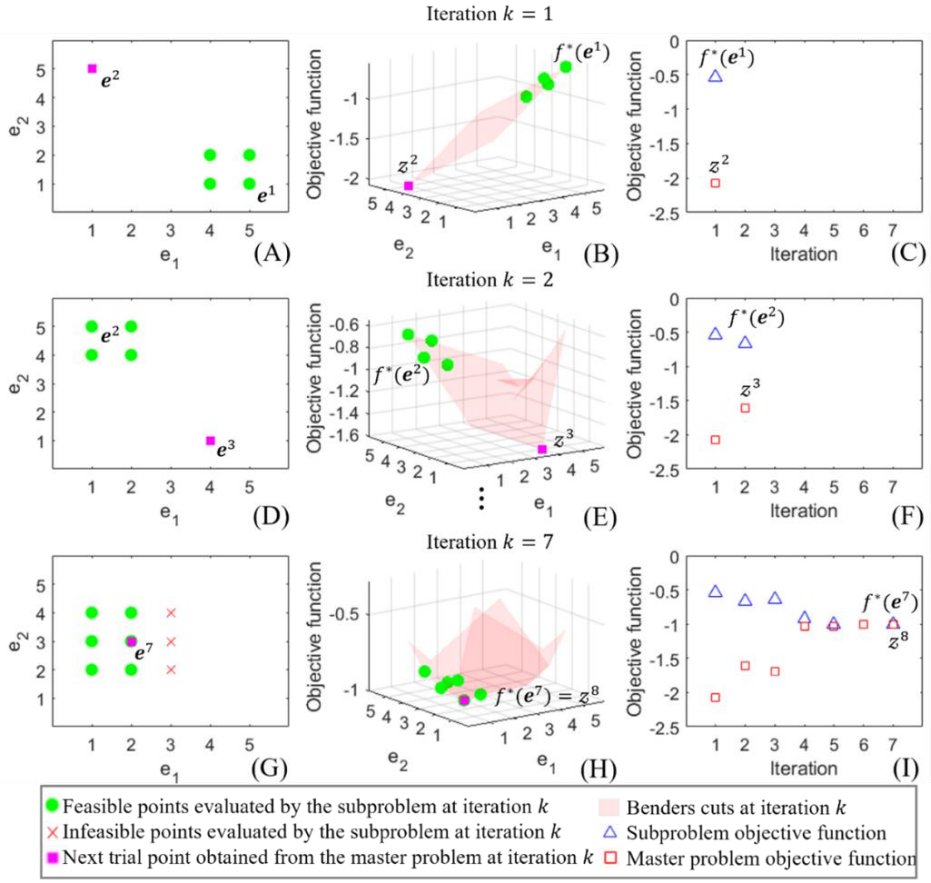


Figure 4.2. Iterations of the LD-BD algorithm for the mathematical example.

The performance of the LD-BD algorithm was compared with that obtained by the LD-SDA, as depicted in Figure 4.3. For this problem, the LD-SDA only performs a local search nearby the initialization provided by the user, which starts with a neighborhood verification around [5,1] followed by a line search over the steepest descent direction (i.e., red line from $k = 1$ to $k = 2$ in Figure 4.3). Due to the infeasible solutions defined by $e_1 = 3$, the LD-SDA is only able to perform a local search within the right-hand-side section of the feasible region in Figure 4.3. In contrast, the LD-BD strategy can move away (back-off) from the initialization in the first iteration of the algorithm, which allows to identify a feasible region of the problem in Eqs. (4.13)-(4.18) that was unexplored by the LD-SDA. In addition, when comparing the local optimum found by each algorithm, the solution obtained by LD-BD ([2,3]) turned out to be the global optimum.

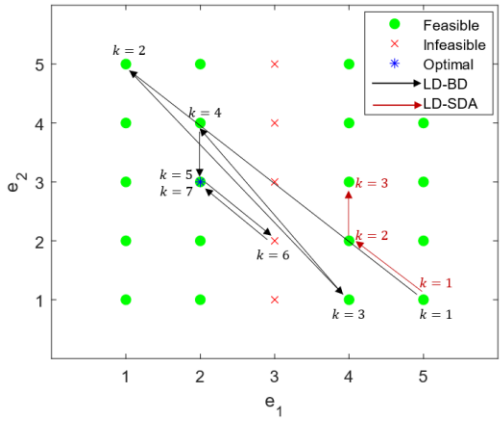


Figure 4.3. Iterations of the LD-BD and LD-SDA algorithms. The arrows indicate the sequence of iterations.

To further verify the features of the LD-BD strategy, a sensitivity analysis was performed over all the possible initial guesses of external variables, i.e., 25 initializations. Then, the optimization problem was solved using the LD-BD and the LD-SDA from all these initializations. Although both algorithms were sensitive to the initialization of $\boldsymbol{e}$, this test showed that LD-BD converged to the global optimum of the problem in 21 out of 25 of the optimization runs tested while the LD-SDA was only able to find the global solution in 11 out of 25 runs. Notably, LD-BD converged to the global optimum when initialized from [4,3] (i.e., a local minimum), which indicates that LD-BD has the ability to escape from the local minimum of this case study. Figure 4.3 also shows that the effective domain ($\mathrm{dom}_{\mathbb{Z}}(f^*)$) of this problem (green circles) is characterized by two disconnected regions with an infeasible region between them at $e_1 = 3$. Hence, $\mathrm{dom}_{\mathbb{Z}}(f^*)$ is not integrally convex despite E being integrally convex.

Therefore, LD-BD cannot guarantee global optimality for this problem independently of the initialization of $\boldsymbol{e}$. Note that the search performed by the LD-BD required a larger number of iterations than the LD-SDA for this problem, as illustrated in Figure 4.3. Accordingly, the LD-BD requires more time to solve this optimization problem than the LD-SDA, i.e., LD-BD requires 1.61 seconds, while LD-SDA requires 0.57 seconds on average.

4.3.2 Case study 2: Distillation superstructure example

This section considers the optimal design of a trayed distillation column that separates benzene from toluene. This optimization problem is summarized in Eqs. (4.19)-(4.25) and involves continuous and discrete variables. The variables $\boldsymbol{x}$ represent the continuous degrees of freedom of the problem, e.g., the reflux ratio (RR) and the boil-up ratio (BR); the reboiler (Q_R) and condenser duties (Q_C) in kW; product composition including the distillate benzene composition ($x_{D,Benzene}$) and the bottoms toluene composition ($x_{B,Toluene}$); temperatures, enthalpies, phase equilibrium variables, vapor and liquid flows and compositions at every stage of the column. A superstructure with 15 potential internal stages, one condenser, and one reboiler is defined to find the optimal number of stages and feed location. The optimization problem considered for this case study is presented in Eqs. (4.19)-(4.25). The objective function in Eq. (4.19) is the sum of the capital cost ($10^3 \sum_{n\in\{2,3,\dots,16\}} y_n$) and operating costs ($Q_R + Q_C$); the term $\sum_{n\in\{2,3,\dots,16\}} y_n$ in the capital cost represents the number of trays as described below. Eqs. (4.20) and (4.21) impose bounds on the reflux ratio and boil-up ratio, respectively, while Eqs. (4.22) and (4.23) define the required product specifications. Boolean variables $\boldsymbol{Y} = [Y_2, Y_3, \dots, Y_{16}]$ are used to find the optimal number of stages and feed location; each Y_n represents the existence/non-existence of candidate internal stage n. Accordingly, Y_1 and Y_{17} represent the reboiler and the condenser respectively; these are fixed at $True$ to represent the existence of a reboiler and a condenser in the final design. Material and energy balances representing the feed stream appear in the formulation at stage 9, i.e., Y_9 is fixed at $True$. Although the feed to the superstructure remains fixed at stage 9, the actual feed stage of the column is expected to be defined by the values of $\boldsymbol{Y}$ after solving the optimization problem. Selecting stage 9 as the feed to the superstructure allows to leave enough candidate internal stages above and below the feed to optimize the size of the rectifying and stripping sections within the column, e.g., if a feed stage at internal stage 16 was selected, then there would be no space for the rectification stage.

Consider for example a solution of the superstructure with fixed terms $[Y_1, Y_9, Y_{17}] = True$ and active variables $[Y_5, Y_6, Y_7, Y_8, Y_{10}, Y_{11}, Y_{12}, Y_{13}] = True$. This process design layout represents a column with nine internal stages, a condenser, a reboiler, and the feed stage located five internal stages above the reboiler. Note that the remaining variables $[Y_2, Y_3, Y_4, Y_{14}, Y_{15}, Y_{16}] = False$ represent those candidate stages that will not physically exist in the final design. A graphical representation of this example is presented in Figure 4.4A. Disjunctions in Eq. (4.24) represent the presence (Y_n)/absence ($\neg Y_n$) of internal stage n. Thus, constraints $\widehat{\boldsymbol{h}}_1(\boldsymbol{x}) = \mathbf{0}$ are the Mass, Equilibrium, Summation, EntHalpy (MESH) that describe existing stages with $Y_n = True$. On the contrary, *bypass* constraints $\widehat{\boldsymbol{h}}_2(\boldsymbol{x}) = \mathbf{0}$ ensure that flows, temperatures, and compositions remain unchanged for stages with $Y_n = False$, indicating that stage n will not appear in the design. In addition, each Boolean variable Y_n is associated to a binary variable y_n that allows to define other constraints in the formulation, e.g., the number of internal stages in the objective function is given by $\sum_{n\in\{2,3,\dots,16\}} y_n$. Eq. (4.25) contains logical constraints, i.e., the column must have at least 8 internal stages; reboiler, condenser and feed stages always remain active; and active stages with $Y_n = True$ should be adjacent to the feed stage (Y_9) to avoid structural redundancy.

$$\min_{\boldsymbol{x},\boldsymbol{Y}} f = \left(10^3 \sum_{n\in\{2,3,\dots,16\}} y_n\right) + (Q_R + Q_C) \tag{4.19}$$

$$s.t.\ 0.5 \leq RR \leq 4 \tag{4.20}$$

$$1.3 \leq BR \leq 4 \tag{4.21}$$

$$x_{D,Benzene} \geq 0.95 \tag{4.22}$$

$$x_{B,Toluene} \geq 0.95 \tag{4.23}$$

$$\begin{bmatrix} Y_n \\ \widehat{\boldsymbol{h}}_1(\boldsymbol{x}) = \mathbf{0} \\ y_n = 1 \end{bmatrix} \vee \begin{bmatrix} \neg Y_n \\ \widehat{\boldsymbol{h}}_2(\boldsymbol{x}) = \mathbf{0} \\ y_n = 0 \end{bmatrix}, \forall n \in \{2,3,\dots,16\} \tag{4.24}$$

$$\Omega(\boldsymbol{Y}) = True \tag{4.25}$$

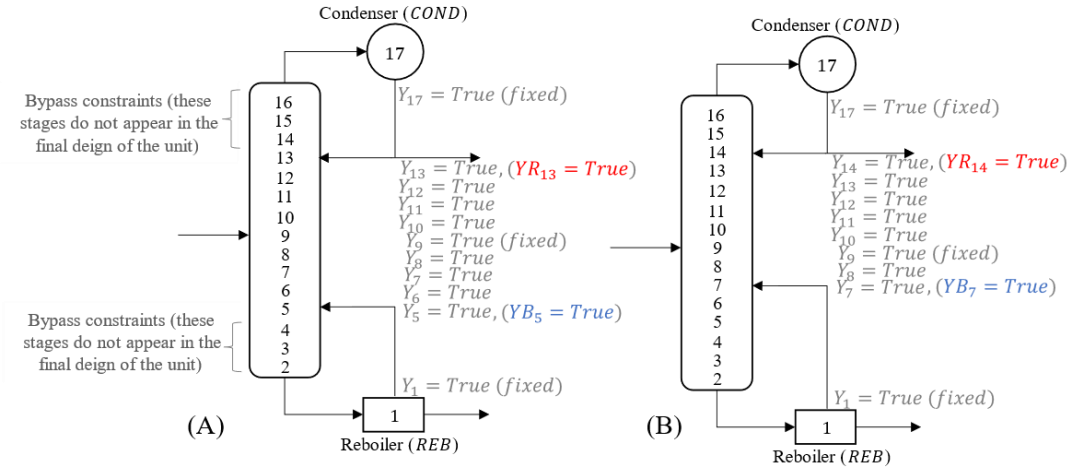


Figure 4.4. (A): Example of a column with 9 internal stages and the feed located at the fifth stage from bottom to top. (B): Example of a column with 8 internal stages and the feed located at the third stage from bottom to top. The remaining Boolean variables in (A) and (B) are $False$.

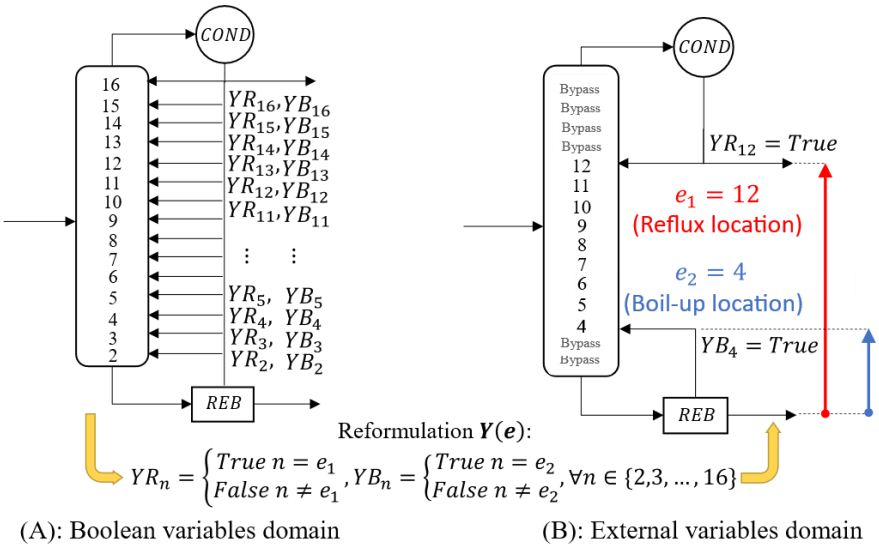
The Boolean variables $\boldsymbol{Y}$ in this case study can be represented with external variables. To perform this reformulation, auxiliary variables YR_n and $YB_n, \forall n \in \{2,3,\dots,16\}$ are introduced in the formulation to represent the reflux and boil-up locations in the superstructure, respectively, e.g., $YB_5 = True$ and $YR_{13} = True$ in Figure 4.4A indicate that $Y_n = True$ from $n = 5$ to $n = 13$. The equivalence expression that relates Y_n with YR_n and YB_n is shown in Eq. (4.26). Note that Eq. (4.27) guarantees that reflux and boil-up flows are recirculated only once within the column. Also, note that $Y_9 = True$ remains fixed, hence varying YR_n and YB_n to define the reflux and boil-up locations indirectly determines the feed location and the number of stages, e.g., different feed locations and number of stages are obtained by varying YR_n and YB_n in Figures 4.4A and 4.4B. A reformulation with external variables applied over variables YR_n and $YB_n, \forall n \in \{2,3,\dots,16\}$ results in integer variables e_1 and e_2, which represent the reflux and boil-up locations, respectively.

$$\left(\bigvee_{j \in \{n, n+1, \dots, 16\}} YR_j\right) \wedge \left(\bigwedge_{j \in \{n, n+1, \dots, 16\}} \neg YB_j \vee YB_n\right) \Leftrightarrow Y_n \tag{4.26}$$

$$\underline{\bigvee}_{n \in \{2,3,\dots,16\}} YR_n, \underline{\bigvee}_{n \in \{2,3,\dots,16\}} YB_n \tag{4.27}$$

The reformulation with external variables is depicted in Figures 4.5A and 4.5B. For instance, Figure 4.5B shows a feasible combination of $e_1 = 12$ and $e_2 = 4$. According to Figure 4.5, these feasible

values of e_1 and e_2 allow to fix $YR_{12} = True$, $YB_4 = True$, $YR_n = False\ \forall n \in \{2,3,\dots,16\}\backslash\{12\}$, and $YB_n = False\ \forall n \in \{2,3,\dots,16\}\backslash\{4\}$. Likewise, YR_n and YB_n define $[Y_4, Y_5, \dots, Y_{12}] = True$ through the equivalence in Eq. (4.26). Thus, the discrete configuration of the system (i.e., number of stages and feed location) is completely specified by fixing e_1 and e_2. When considering the reformulated version of this problem, it is possible to perform a complete enumeration of the combinations of e_1 and e_2, and solve the resulting NLP subproblems to identify the optimal solution of this optimization problem. The result of this test is shown in Figure 4.6, with a unique optimal solution at $\boldsymbol{e} = [14,5]$. Knowing the optimal solution of this problem *a priori* allows to assess the performance of the LD-BD method with respect to existing strategies.



(A): Boolean variables domain (B): External variables domain

Figure 4.5. Representation of the discrete decisions of the distillation problem in the Boolean variables domain (A) and the external variables domain (B).

The solution of distillation design problems with superstructure optimization strategies is typically sensitive to the initialization of the discrete decisions (i.e., initial guess of $\boldsymbol{e}$). Therefore, the performance of the LD-BD method can be tested by defining multiple optimizations runs, each with a different initialization of external variables. This allows to compare its ability to find the global optimum with respect to the existing local logic-based strategies, i.e., L-OA and LD-SDA. In this work, multiple initializations were sorted with respect to their distance to the global optimum of the problem, e.g., [14,5] and [10,10] represent initializations 1 and 63, respectively, as shown in Figure 4.6. Note that some of the initialization points are infeasible. The solutions obtained from these optimization tests

are shown in Figure 4.7. From Figure 4.7A, the solver L-OA was not able to consistently find the optimal objective function, converging to the globally optimal solution of the problem in only 2 out of 63 optimization trials, i.e., only 3.2% of the runs converged to $\boldsymbol{e} = [14,5]$. LD-SDA improved with respect to L-OA and converged to the optimal solution in 49.2% of the optimization runs; however, this strategy failed when initialized from infeasible points, e.g., initializations 55 to 63. In contrast to these solution strategies, the proposed LD-BD method converged to the global optimal solution in all (100%) of the optimization trials. This result shows the independence of this method from the initialization of discrete decisions. Notably, this optimization problem is not integrally convex, as demonstrated in section D.4 of appendix D. Nevertheless, LD-BD still converged to the global solution independently of the initial guess of $\boldsymbol{e}$ provided, as opposed to the case study presented in section 4.3.1. This is because there is a single discrete point (i.e., the optimal solution in Figure 4.6) that satisfies local optimality within the $N_\infty(\boldsymbol{e})$ neighborhood; similarly, all the conditions in Theorem 4.2 were met, i.e., the algorithm ran until the optimality gap closed, none of the subproblems was unbounded, feasible region E was bounded, the master problem was solved to optimality at each iteration, and the procedure discussed in section D.3 of appendix D to attain feasible solutions converged to a feasible point when needed. Since LD-BD ensures convergence to a locally optimal solution with respect to the $N_\infty(\boldsymbol{e})$ neighborhood (see Theorem 4.2), and this local optimum is unique in this case study, convergence to the global optimum was possible from every initial point in E.

As shown in Figure 4.7B, the solution time required by LD-BD is on average 23.4 seconds, which is within the average CPU times required by LD-SDA (9.9 seconds) and L-OA (30.1 seconds). Note that LD-BD also shows the best performance in terms of solution quality with respect to local MINLP solvers (see section D.5 of appendix D). For instance, DICOPT shows the best performance in terms of computational time (converged to a solution in 9.8 seconds on average); however, only 1 out of 63 optimization trials converged to the best-known solution. Moreover, SBB was the best MINLP solver for this problem converging to the optimal solution in 90.4% of the optimization trials though it required 90.9 seconds on average to converge, i.e., almost a quarter of the time need by LD-BD. Thus, LD-BD provides a benchmark for the solution of this optimization problem using local logic-based and MINLP optimization strategies.

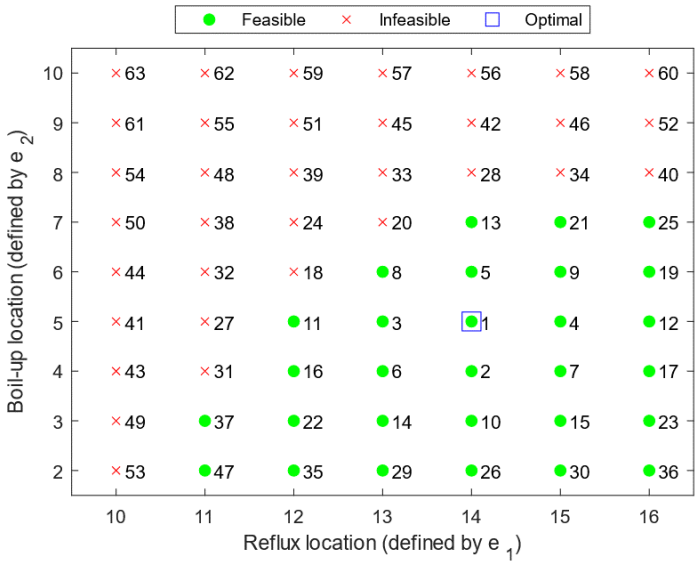


Figure 4.6. Feasible region of the distillation problem and enumeration of the initial guesses for external variables. The numbers in the grid are identifiers assigned to each point within the feasible region.

In this case, the good performance of LD-BD and LD-SDA with respect to L-OA is attributed to the requirements of each strategy to guarantee the global solution of the problem. While L-OA guarantees convergence to a global optimum when the continuous relaxation of all disjunctions and constraints is convex, LD-BD and LD-SDA guarantee global optimality in different circumstances, e.g., when the objective function of the problem is integrally convex, or when only one discrete point within the search region E is optimal within a local neighborhood (see section 4.2.4 for details)[4]. The fact LD-BD and LD-SDA provide alternative convergence properties based on the ordered nature of discrete decisions allows their convergence to improved locally optimal solutions. Although the continuous relaxation of this case study is non-convex, only one discrete solution ($\boldsymbol{e} = [14,5]$) within the search space satisfies the requirement that its objective function is minimum within $N_\infty(\boldsymbol{e})$; thus, LD-BD and LD-SDA always attained the best-known solution of the problem when they converged.

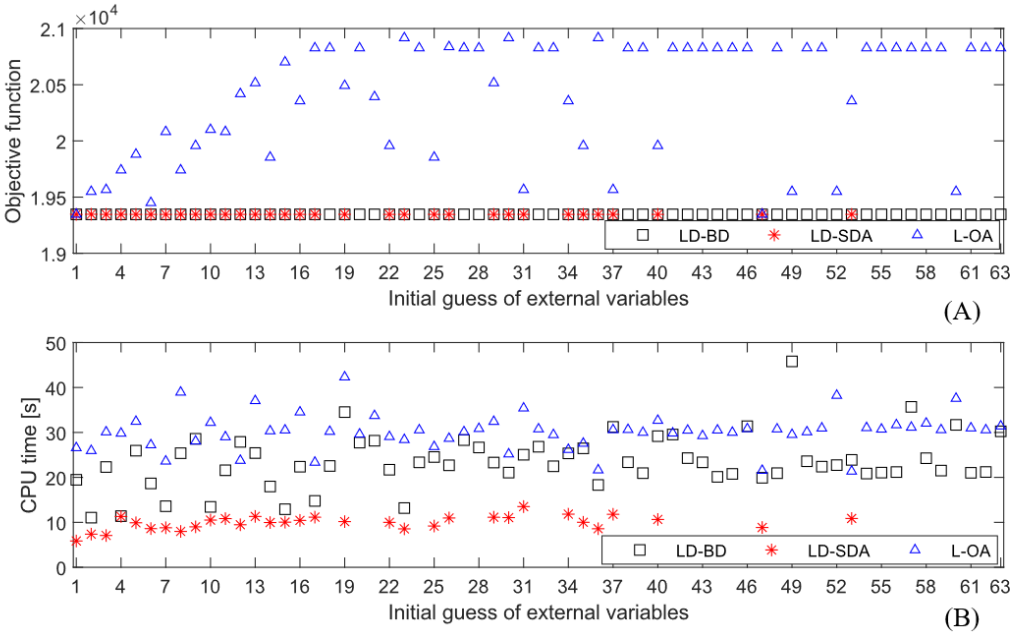


Figure 4.7. Results of running LD-BD, LD-SDA and L-OA from different initializations of external variables. (A): objective function, (B): computational time.

4.3.3 Case study 3: Reactor network example

This section considers the optimal design of a reactor network, which is a GDP problem previously proposed in the literature[3]. The autocatalytic liquid-phase reaction $A + B \rightarrow 2B$ with rate expression $r_A = -k_{rate} C_A C_B$ is considered, were k_{rate} is the kinetic constant in $l/mol\, s$ and C_i is the concentration of component i. This problem aims to minimize the total reactor volume of a series of CSTR reactors, while guaranteeing the desired product specification and reactors of equal volume. The problem considers the number of reactors in series (e_1) and the location of a recycle (e_2) as external variables. Since the problem's formulation allows a maximum of 5 reactors in series and a possible recycle located in any of those reactors, there is a total of 25 different discrete configurations for this problem. More details about this problem and its reformulation are available in section D.6 of appendix D.

In contrast to the distillation problem considered in section 4.3.2, this optimization problem is known to have 4 solutions that satisfy the optimality conditions within the $N_\infty(\boldsymbol{e})$ neighborhood[4]. Among these solutions, $\boldsymbol{e} = [5,5]$ is the unique global optimum of the problem. Since there are multiple local solutions with different objective function values, neither LD-SDA nor LD-BD showed independence

of the initialization of external variables for this problem, i.e., both algorithms found different local optima depending on the initialization. One advantage of LD-BD is that it can be warm-started if D^{init} is initialized with a set of discrete configurations whose NLP subproblems were solved before running the LD-BD algorithm. The larger the number of discrete configurations previously evaluated, the higher the likelihood to find the global optimum of the problem. To computationally demonstrate this feature, ω discrete configurations were sampled without replacement from feasible region $E = \{[e_1, e_2] \in \mathbb{Z}^2 : 1 \leq e_1 \leq 5, 1 \leq e_2 \leq 5\}$ and added to D^{init} before running the LD-BD method. This test was repeated 500 times for each ω, with ω ranging from $\omega = 10$ to $\omega = 20$. The percentage of runs that converged to the global optimum ($P(\omega)$) was computed for each ω and is depicted in Figure 4.8.

As shown in Figure 4.8, $P(\omega)$ becomes larger than 90% at $\omega = 16$, and approaches 95% when $\omega = 20$. Consequently, warm-starting LD-BD with either the naïve sampling strategy considered in this work, or any other pre-processing sampling strategy improves the performance of LD-BD at finding globally optimal solutions. The cost of performing a sampling before the execution of LD-BD is that the total computational time increases. In this case, solutions with $\omega = 10$ required on average 3.49 seconds, while solutions with $\omega = 20$ required on average 5.08 seconds. Note that the solution of this problem using the global logic-based solver GL-OA converged to the global optimum in 7.25 seconds, which is slightly slower than a complete enumeration of the problem in the domain of external variables (6.79 seconds). Moreover, global solver L-BB showed the worst performance, converging to the global optimum in 334 seconds. This confirms the advantages of performing a reformulation and using a tailored solver such as LD-BD before solving this optimization problem.

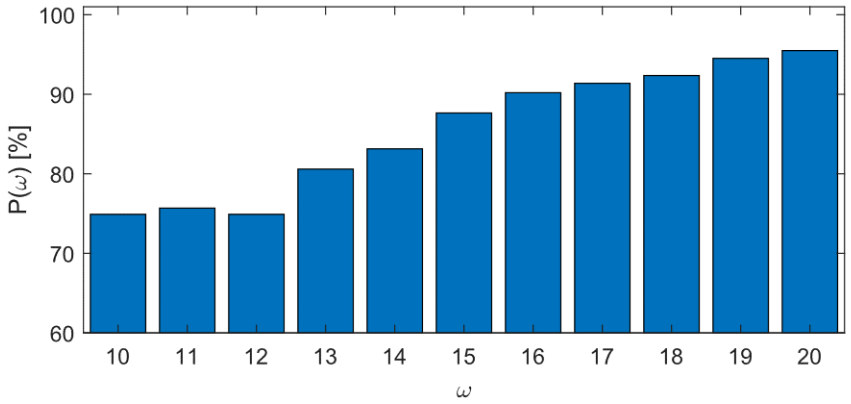


Figure 4.8. Effect of increasing the number of randomly sampled points in D^{init}

Chapter 5
A Hybrid Deterministic-Stochastic Algorithm for the Optimal Design of Process Flowsheets with Ordered Discrete Decisions

In this chapter, a hybrid algorithm that combines stochastic and deterministic optimization methods is proposed to exploit the advantages of stochastic and deterministic techniques and to reduce the computational effort required to optimize the design of chemical process flowsheets involving a mixture of continuous and ordered discrete decisions. Examples of ordered discrete decisions include selecting the number of stages in multi-effect evaporator sequences [269], number of reactors connected in series or parallel [280], or the number of trays and location of interconnecting streams in separation sequences [91]. Chemical engineering software with simulation and Nonlinear Programming (NLP) optimization capabilities is considered for modeling purposes through this chapter.

In the context of simulation-based flowsheet optimization, the utilization of pure stochastic methods (SMs), specifically the Differential Evolution with Tabu List (DETL), has undergone extensive testing in solving highly nonlinear problems characterized by a large number of equations, which include features such as discontinuous and multi-objective functions [281–288]. SMs have proven to be efficient at finding good quality solutions, thanks to their expected convergence to a global optimum in infinite time [32,289]. Nonetheless, the computational effort required by these methods is often significant and is still an open challenge. This bottleneck has prompted the exploration of a parallel-hybrid methodology, conceived with the objective of tackling the existing computational time limitations.

While previous works hybridize deterministic and stochastic algorithms in sequential and nested arrangements (Figures 5.1A and 5.1B), the present work proposes a parallel configuration to perform the hybridization, i.e., the stochastic algorithm runs in one processor whereas the deterministic strategy runs in an alternate processor (Figure 5.1C). The proposed hybrid algorithm combines a SM with a deterministic optimization algorithm that is based on the previously developed deterministic D-SDA [7]. A new version of the traditional D-SDA that integrates this method with a variable bounding (VB) strategy that iteratively generates tighter search bounds and avoids convergence failures is presented herein (DSDA-VB). A key feature of this deterministic algorithm is its capability to optimize discrete degrees of freedom (DOF) within flowsheet simulation software. This capability eliminates the need to rely solely on a SM for optimizing discrete decisions. Rather than focusing on the comparison of

different hybrid strategies, this work highlights the performance of the proposed parallel-hybrid strategy against a pure SM. By reducing computational time, the proposed parallel-hybrid approach aims to make SMs more accessible and applicable in real-world scenarios. Furthermore, the DSDA-VB is proposed as a deterministic MINLP algorithm integrated within the proposed hybrid method. The proposed DSDA-VB strategy allows to iteratively update the search region explored by the SM. The VB strategy also provides adequate variable bounds when the NLP solver used by DSDA-VB is prone to convergence failure, which is a common situation in process flowsheet simulation software.

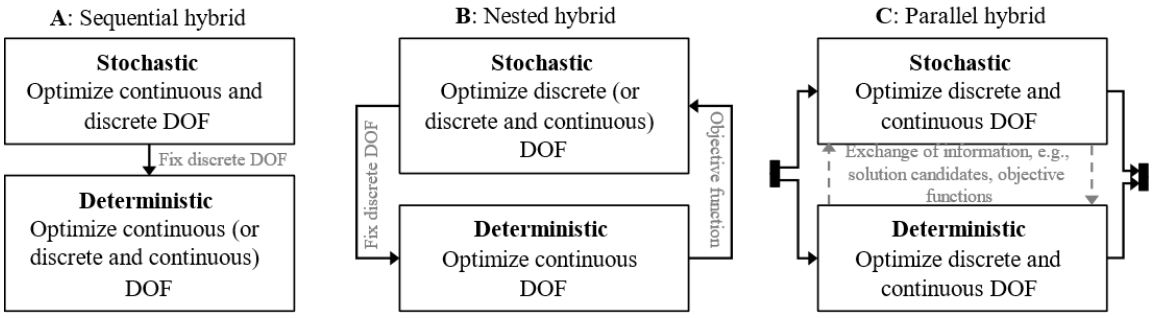


Figure 5.1. Hybrid arrangements for optimal flowsheet design aided with simulation software. A: sequential hybrid, B: nested hybrid, and C: parallel hybrid. DOF means degrees of freedom.

This chapter is organized as follows: Section 5.1 presents a general description of the optimization problem addressed in this chapter. Section 5.2 describes the working principle of the SM and DSDA algorithms. This section also introduces the proposed hybrid algorithm and the DSDA-VB strategy. Section 5.3 illustrates the main features of the proposed method using a thermally coupled system and a sequence of reactive, extractive, and traditional distillation columns as case studies.

5.1 Problem statement

This chapter considers the optimization problem presented in Eq. (5.1), which involves the minimization of an objective function (f), e.g., minimize capital and operating costs. Vector of variables $\boldsymbol{x} \in \mathbb{R}^{n_x}$ represents the continuous DOF of the problem, e.g., output flow specifications, equipment sizes, energy requirements, among others. Variables in $\boldsymbol{y} \in \mathbb{Z}^{n_y}$ are the discrete DOF of the problem, i.e., a given combination of $\boldsymbol{y}$ completely specifies structural decisions of the flowsheet such as number of sequential/parallel units, feed locations, location of interconnecting flows, etc. There are other continuous variables that are handled internally by the simulator. Among these, variables in vector

$\boldsymbol{z} \in \mathbb{R}^{n_z}$ are the outputs of interest among the internally managed variables, e.g., output flows and compositions that are not specified by the user.

$$\begin{aligned} &\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}} f(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \\ &s.t. \\ &\boldsymbol{x}^L \le \boldsymbol{x} \le \boldsymbol{x}^U,\ \boldsymbol{y}^L \le \boldsymbol{y} \le \boldsymbol{y}^U,\ \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \le \boldsymbol{0},\ [\boldsymbol{x},\boldsymbol{y}] \in \widehat{\Omega} \end{aligned} \tag{5.1}$$

Different types of constraints are considered, including bounds on the decision variables ($\boldsymbol{x}^L \le \boldsymbol{x} \le \boldsymbol{x}^U,\ \boldsymbol{y}^L \le \boldsymbol{y} \le \boldsymbol{y}^U$), constraints with a known mathematical expression ($\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \le \boldsymbol{0}$) and the constraints handled internally by black-box simulations ($[\boldsymbol{x},\boldsymbol{y}] \in \widehat{\Omega}$). The upper (superscript U) and lower (superscript L) bounds for continuous and discrete DOF are denoted by $\boldsymbol{x}^L$, $\boldsymbol{x}^U$ and $\boldsymbol{y}^L$, $\boldsymbol{y}^U$ respectively. Given that one of the key features of the proposed hybrid algorithm is the iterative refinement of these bounds, their values will be updated throughout the iterative solution of optimization problems. Thus, bounds $\boldsymbol{x}^{L,0}, \boldsymbol{x}^{U,0}, \boldsymbol{y}^{L,0}$ and $\boldsymbol{y}^{U,0}$ will be referred to as the user-defined bounds of the problem, which impose limits over the iteratively refined algorithmic bounds as follows: $\boldsymbol{y}^{L,0} \le \boldsymbol{y}^L \le \boldsymbol{y} \le \boldsymbol{y}^U \le \boldsymbol{y}^{U,0}$ and $\boldsymbol{x}^{L,0} \le \boldsymbol{x}^L \le \boldsymbol{x} \le \boldsymbol{x}^U \le \boldsymbol{x}^{U,0}$. Vector of constraints $\boldsymbol{g}$ includes those design requirements that are specified by the user, e.g., product compositions. Other constraints are handled internally by the simulator, and these are represented through set $\widehat{\Omega}$; these constraints include mass and energy balances, equilibrium relationships and thermodynamic models, reaction rates, etc. Set $\widehat{\Omega}$ is therefore defined as those combinations in $\boldsymbol{x}$ and $\boldsymbol{y}$ that guarantee convergence of all the constraints within the simulator. Throughout this work, it is assumed that objective function f is single-objective function that is bounded from below, and that the effective domain of f is nonempty. Also, functions f and $\boldsymbol{g}$ are assumed to be twice differentiable with respect to the continuous variables. Concerning the structure of the DOF, the only requirement is that variables $\boldsymbol{y}$ must be ordered, which is a common feature in process flowsheets such as sequences of intensified distillation systems, reactor networks, multi-effect evaporator sequences, among others.

The solution strategy developed in this study aims to handle problems involving multiple interconnected units in a flowsheet, with high dimensionality in the discrete ($\boldsymbol{y}$) and continuous ($\boldsymbol{x}$) DOF. This work emphasizes the application of the proposed strategy to process optimization with flowsheet simulators using rigorous process models, thanks to the flexibility process simulators offer to include multiple operation units in a simulation, i.e., their readily available unit operations, non-ideal

thermodynamic and physical property models allow to simulate processes with multiple interconnected unit operations. Nevertheless, the hybrid algorithm proposed in this work could also be applied in the context of optimization modeling languages, e.g., GAMS or Pyomo, although these would require the users to build their own first-principles process models.

5.2 Mathematical framework

The approach proposed in this work to solve problem (5.1) is a hybrid deterministic-stochastic algorithm that combines an SM with the herein proposed DSDA-VB strategy. An overview of stochastic and deterministic optimization using the D-SDA is provided in sections 5.2.1 followed by the proposed hybrid algorithm presented in section 5.2.2.

5.2.1 Overview of stochastic methods and the DSDA

Stochastic methods (SMs) aim to find the optimal solution of a problem by iteratively generating and evaluating a set of candidate solutions based on random sampling. These algorithms use a probabilistic approach to explore the search space and escape local optima, rather than relying on deterministic approaches that often converge to a local solution. The operation of stochastic optimization algorithms typically involves four steps: initialization, generation, evaluation, and selection. These steps are repeated until a satisfactory solution is found or a stopping criterion is met. The initialization step involves generating an initial set of candidate solutions randomly. The generation step involves creating new candidate solutions by applying random perturbations or mutations to the current population. The evaluation step involves calculating the objective value of each candidate solution using the objective function and constraints defined for the problem. The objective value is used to assess the quality of the solution and compare it to other candidate solutions. The selection step involves choosing the best candidate solutions based on some criteria, such as the highest objective value, lowest cost, or shortest distance. Once the selection step is completed, the algorithm repeats the generation, evaluation, and selection steps until a satisfactory solution is found or a stopping criterion is met, e.g., the number of iterations, the quality of the best solution found, or a predefined threshold [290,291].

The D-SDA is a deterministic decomposition algorithm originally developed for integer programming[6], and recently extended to MINLP optimization problems involving continuous and ordered discrete decisions such as those emerging in vector $\boldsymbol{y}$. To initialize the D-SDA, the user must provide a feasible initialization for the variables in $\boldsymbol{y}$. At each iteration, the D-SDA fixes the values of

discrete decisions at their current trial value (denoted by $\boldsymbol{y}^*$) in Problem (5.1) and solves for the remaining continuous variables $\boldsymbol{x}$ and $\boldsymbol{y}$ using NLP optimization to return an objective function value $f^*(\boldsymbol{y}^*) = f(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{z}^*)$ with optimized continuous variables $\boldsymbol{x}^*$ an $\boldsymbol{z}^*$. The NLP optimization problem that results from fixing discrete decisions in Problem (5.1) is referred to as a subproblem. Discrete variables are then fixed at every discrete point within a neighborhood of $\boldsymbol{y}^*$ denoted as $N_\Theta(\boldsymbol{y}^*)$, where Θ indicates the neighborhood type. For example, this neighborhood typically considers those discrete points whose infinity ℓ_∞ -norm (infinite norm) with respect to $\boldsymbol{y}^*$ is less or equal to 1, and it is denoted by $N_\infty(\boldsymbol{y}^*)$. To reduce the computational effort, subsets of $N_\infty(\boldsymbol{y}^*)$ such as the N_2-neighborhood ($N_2(\boldsymbol{y}^*)$) with $2n_y + 1$ points within an ℓ_2 -norm of $\boldsymbol{y}^*$ are often considered in practice. Each neighbor $\boldsymbol{y} \in N_\Theta(\boldsymbol{y}^*)$ results in an NLP subproblem that returns a new objective function value $f^*(\boldsymbol{y}) = f(\boldsymbol{x}^*(\boldsymbol{y}), \boldsymbol{y}, \boldsymbol{z}^*(\boldsymbol{y}))$, where notation $\boldsymbol{x}^* = \boldsymbol{x}^*(\boldsymbol{y})$ and $\boldsymbol{z}^* = \boldsymbol{z}^*(\boldsymbol{y})$ represents the optimized values of $\boldsymbol{x}$ and $\boldsymbol{z}$ with discrete variables $\boldsymbol{y}$ fixed in Problem (5.1). Given that discrete variables in $\boldsymbol{y}$ are ordered, concepts from discrete convex analysis can be used to define local optimality, as stated in Definition 5.1.

Definition 5.1. Consider function $f^*(\boldsymbol{y}) = f\big(\boldsymbol{x}^*(\boldsymbol{y}), \boldsymbol{y}, \boldsymbol{z}^*(\boldsymbol{y})\big)$, where function $f^*: \mathbb{Z}^{n_y} \mapsto \mathbb{R} \cup \{+\infty\}$ takes the optimal objective function of the subproblem obtained after fixing $\boldsymbol{y}$ in Problem (5.1) for feasible values of $\boldsymbol{y}$ and $+\infty$ otherwise. A point $\boldsymbol{y}^*$ is a *local minimum* of f^* if $f^*(\boldsymbol{y}^*) \leq f^*(\boldsymbol{y})$ for every $\boldsymbol{y}$ that belongs to $N_\Theta(\boldsymbol{y}^*)$ [6].

Based on Definition 5.1, if $f(\boldsymbol{y}^*)$ takes the minimum value within its neighborhood, then $\boldsymbol{y}^*$ is declared as locally optimal and the D-SDA algorithm stops; otherwise, the combination of discrete decisions that minimize the objective function is used to define a steepest-descent direction. NLP problems with discrete variables fixed are then sequentially solved along the steepest-descent direction until no improvement is found. Then, a new iteration of the algorithm begins with a new neighborhood exploration. Previous works have demonstrated that the D-SDA outperforms traditional MINLP and GDP optimization algorithms when optimizing systems involving binary or Boolean variables defined over ordered discrete sets [7,15]. Note that several discrete analogues of convexity such as integral-convexity, $M^\natural$-convexity and $L^\natural$-convexity have been studied in the literature to detect when a local optimum according to Definition 5.1 is a global solution (e.g., see [292–294]); however, function $f^*(\boldsymbol{y})$ does not have an analytical representation because it is obtained via NLP optimization, i.e., the

convexity properties of $f^*(\boldsymbol{y})$ are unknown a priori. For this reason, only local optimality can be guaranteed in this study.

Some of the limitations of stochastic and deterministic methods can be overcome by combining them into a single optimization framework when solving flowsheet design problems. Typically, exceeding a resource limit is considered as the stopping criterion for SMs, e.g., stop once a maximum number of objective function evaluations is attained ($Fmax$). However, selecting the appropriate resource limit (e.g., $Fmax$ value) requires trial-and-error tests, given that the computational performance depends on the specific characteristics of each optimization problem, e.g., number of variables, non-linearities, and variable's bounds. Additionally, the stopping criterion may differ in different runs of the same problem due to the stochastic nature of SM algorithms [289]. A deterministic optimization algorithm such as D-SDA can help to alleviate these issues since it can be used to improve and assess the optimality characteristics of the current best solutions identified by the SM. Hence, if the current SM solution is close to a locally optimal solution, then the execution of the SM can stop. Nevertheless, the D-SDA exhibits other limitations, such as the initialization requirement (i.e., a feasible combination of discrete decisions), or the exponential growth of discrete neighbors when using a ℓ_∞ -norm to define the vicinity of a discrete point [15]. The DSDA may benefit from SMs, which use randomness and probabilistic techniques to search for feasible and optimal solutions. These methods do not guarantee finding optimal solutions, but they can be efficient in exploring large search spaces and finding acceptable feasible solutions. In the case of constrained optimization problems, stochastic optimization methods typically use a penalty function approach, where the objective function is modified to include a term that penalizes infeasible solutions. This penalty term increases as the solution violates constraints, and the objective function is minimized while keeping the penalty as low as possible. Concerning the size of the neighborhoods, subsets of $N_\infty(\boldsymbol{y}^*)$ such as $N_\Theta(\boldsymbol{y}^*) = N_2(\boldsymbol{y}^*)$ have been considered in previous D-SDA works to decrease computational costs, at the expense of scarifying local optimality within the larger $N_\infty(\boldsymbol{y}^*)$ neighborhood. To alleviate this issue, both the SM and the D-SDA can perform the neighborhood exploration, by exploring $N_\Theta(\boldsymbol{y}^*) \subset N_\infty(\boldsymbol{y}^*)$ neighborhoods with DSDA and heuristically using the SM to find candidate solutions within the larger $N_\infty(\boldsymbol{y}^*)$ neighborhood.

One problem that affects both the D-SDA and SMs is the adequate selection of bounds to perform the search (i.e., $\boldsymbol{x}^{L,0}, \boldsymbol{x}^{U,0}, \boldsymbol{y}^{L,0}$ and $\boldsymbol{y}^{U,0}$). Bounds on optimization variables limit the search space of SMs, which can have a significant impact on the optimization process. While tight bounds can lead to premature convergence and limit the exploration of the search space, thereby leading to suboptimal

solutions, overly loose bounds can lead to a broader search space, thus allowing the identification of attractive solutions, at the cost of lengthy (or even prohibitive) computational times. Selecting bounds that balance the trade-off between exploration and exploitation of the search space is thus key to find attractive feasible solutions in acceptable turnaround times. If bounds are not properly selected for SMs, slow convergence, algorithm stagnation or lack of convergence may arise. In the case of the D-SDA, an inadequate selection of bounds for discrete variables (i.e., $\boldsymbol{y}^{L,0}$ and $\boldsymbol{y}^{U,0}$) does not represent a major concern since the algorithm is successively fixing discrete variables and discarding infeasible neighbors. Nevertheless, bounds for continuous variables (i.e., $\boldsymbol{x}^{L,0}$ and $\boldsymbol{x}^{U,0}$) must be carefully selected for the NLP solver used within D-SDA, whose performance may be affected by these bounds. For instance, the NLP optimization performed by process flowsheet simulators such as Aspen Plus typically implements Sequential Quadratic Programming (SQP), which requires good initial estimations and suitable bounds for the continuous DOF. Overly restricted variable bounds may result in optimal solutions lying at their lower/upper bounds, while loose bounds may result in lack of convergence of the SQP algorithm.

5.2.2 Proposed hybrid deterministic-stochastic algorithm

The hybrid algorithm presented in this section addresses the limitations described above for SMs and the D-SDA by iteratively refining variable bounds within a parallel-hybrid algorithm. In this work, the variable's bounds considered by the SM are updated by the deterministic algorithm. That is, iteratively refined variable bounds (i.e., modified algorithmic bounds $\boldsymbol{x}^{L,M}, \boldsymbol{x}^{U,M}, \boldsymbol{y}^{L,M}$ and $\boldsymbol{y}^{U,M}$) are used to iteratively update the bounds on the decision variables ($\boldsymbol{x}^{L}, \boldsymbol{x}^{U}, \boldsymbol{y}^{L}$ and $\boldsymbol{y}^{U}$), instead of using fixed user-defined bounds (i.e., $\boldsymbol{x}^{L} = \boldsymbol{x}^{L,0}, \boldsymbol{x}^{U} = \boldsymbol{x}^{U,0}, \boldsymbol{y}^{L} = \boldsymbol{y}^{L,0}$ and $\boldsymbol{y}^{U} = \boldsymbol{y}^{U,0}$) throughout the execution of the SM. For continuous variables ($\boldsymbol{x}$), this is achieved by embedding the D-SDA within a variable bounding (VB) algorithm that iteratively refines bounds $\boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^{U,M}$ based on the convergence status of the NLP optimizer, i.e., the DSDA-VB algorithm. For discrete variables ($\boldsymbol{y}$), the N_{∞} neighborhood of each locally optimal solution found by DSDA-VB defines refined bounds $\boldsymbol{y}^{L,M}$ and $\boldsymbol{y}^{U,M}$.

Parallelizing the SM and the DSDA-VB offers different advantages. First, both the SM and DSDA-VB are being executed simultaneously, resulting in faster convergence compared to processing the SM and DSDA-VB optimization steps sequentially. Also, the proposed parallelization allows discrete and continuous DOF to be optimized by both the deterministic and stochastic techniques simultaneously, which leads to solutions that satisfy local optimality requirements for both discrete and continuous

DOF. This feature would not be possible using the available hybrid stochastic-deterministic strategies for flowsheet optimization with process simulators. The parallel implementation presented in this work also offers advantages with respect to the independent execution of the D-SDA and an SM, e.g., improvements in neighborhood exploration compared to the traditional D-SDA, and improved convergence of the SM in situations where the traditional SMs may stagnate.

Based on the above, the SM and DSDA-VB algorithms benefit from each other by considering their parallel execution in an iterative manner. As shown in Figure 5.2, the algorithm starts with an initialization that consists of sequential *Steps S1*, *S2.1* and *D1*. Then, the parallel algorithm begins by simultaneously executing *Step S3* of a stochastic algorithm that iteratively executes the SM (see section 5.2.2.2), and *Step D2* of a deterministic algorithm that iteratively executes the DSDA-VB (see section 5.2.2.3). The main iterations of the parallel-hybrid algorithm are denoted by superscript k. The overall idea of the proposed algorithm is to initialize and start the execution of the SM (*Steps S1* and *S2.1*) and have the SM running, while the deterministic algorithm is constantly retrieving a database consisting of all feasible solutions (F_S) detected by the SM. The deterministic algorithm waits until enough feasible solutions are available in F_S. Then the DSDA-VB is initialized (*Step D1*) by assigning an objective function value of $+\infty$ to the best objective function identified thus far by the DSDA-VB ($f_B^0 \leftarrow +\infty$). The DSDA-VB uses the solution with the best objective function value from F_S (i.e., f_S^k) as initialization (*Step D2*), and then performs a local optimization to generate modified bounds $\boldsymbol{x}^{L,M}, \boldsymbol{x}^{U,M}, \boldsymbol{y}^{L,M}$ and $\boldsymbol{y}^{U,M}$ (*Steps D3* and *D4*), which are returned to the SM when there is an improvement in f_B^k (see *Step D5* and purple dashed lines in Figure 5.2). If f_B^k improved in *Step D5*, then the SM search is re-started using the modified bounds obtained by DSDA-VB (*Steps S4* and *S2.2*). This procedure is repeated until the stopping criterion in *Step D6* is satisfied, meaning that both the deterministic (*Step D7*) and stochastic (*Step S5*) algorithms are terminated. According to this stopping criterion, the parallel hybrid strategy stops when the relative difference between f_B^k and f_S^k is below an ϵ-tolerance. Note that iterations k are controlled by the deterministic algorithm i.e., *Steps D2-D6*. Iterations are managed by the deterministic steps because these define when the SM must be reinitialized (see *Step D5* and red dashed lines in Figure 5.2), and when both deterministic and stochastic steps must stop (see *Steps D6* and *D7* and green dashed lines in Figure 5.2). Figure E.1 in appendix E provides in-depth (high level) information about the algorithm presented in Figure 5.2, including information related to solver interactions, variables and constraints considered at each step, and data exchange aspects.

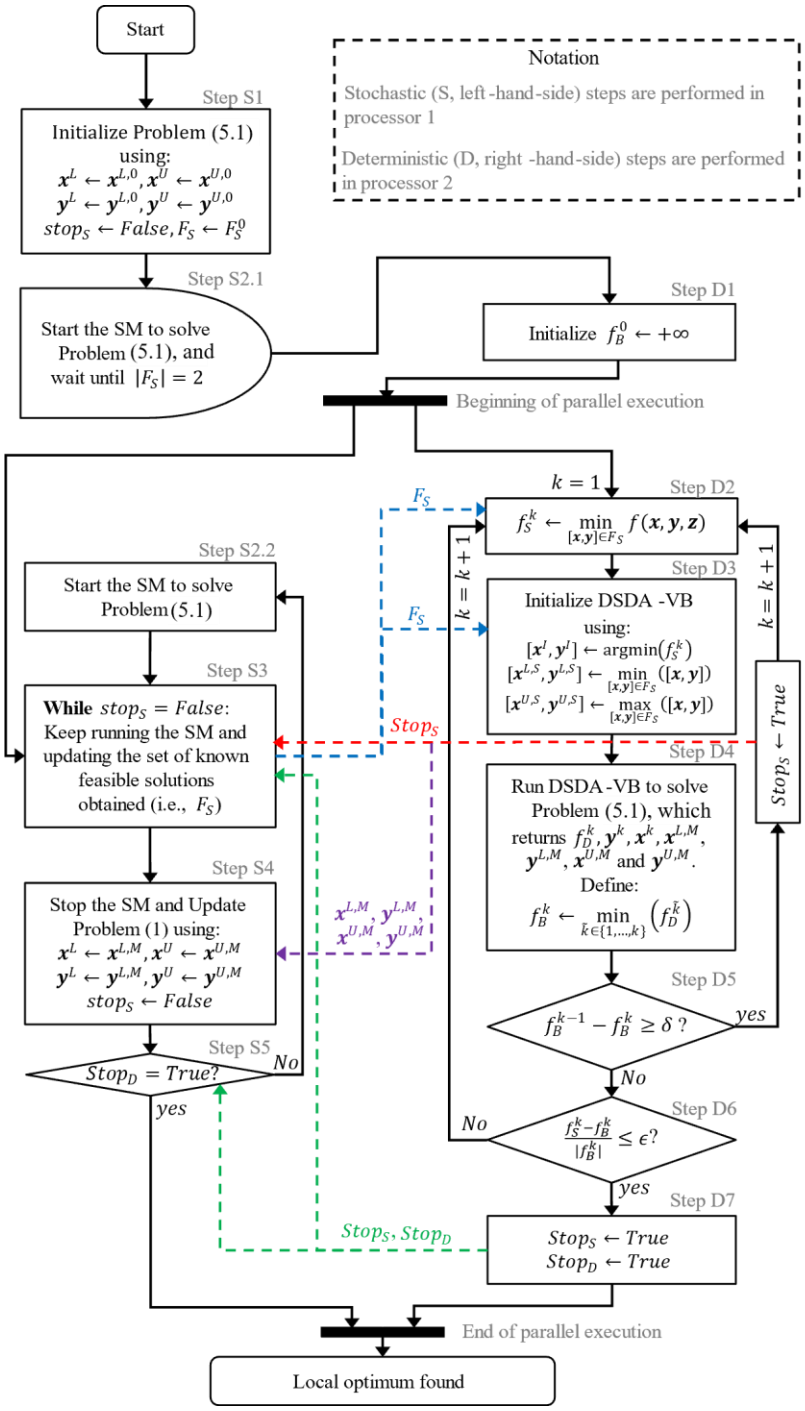


Figure 5.2. Proposed deterministic-stochastic algorithm. Black rectangles at the top and the bottom represent the beginning and the end of the parallelization, respectively. Dashed lines represent the exchange of information.

5.2.2.1 Convergence analysis and summary

The proposed hybrid method enforces requirements over the stochastic and deterministic algorithms to successfully converge to a local optimum. In general, the hybrid stochastic-deterministic strategy introduced above is expected to converge and close the gap within an ϵ-tolerance ($\epsilon \geq 0$) if the three conditions detailed in Theorem 5.1 are satisfied (a detailed proof of this theorem is available in section E.2 of appendix E).

Theorem 5.1. *Assume that: 1) the first run of the SM in Step S1 (iteration $k = 0$) can identify at least two different feasible solutions which are added to F_S; 2) the DSDA-VB in Step D4 can eventually find at least one locally optimal objective function (f_D^k) using feasible information in F_S, and 3) the objective function in optimization problem (5.1) is bounded, i.e., the objective function cannot be improved indefinitely without violating the constraints, and the SM in Steps S2.2 and S3 eventually converges towards the global optimum, subject to the current selection of bounds ($\boldsymbol{x}^{L,M}, \boldsymbol{x}^{U,M}, \boldsymbol{y}^{L,M}, \boldsymbol{y}^{U,M}$). Then, the proposed hybrid stochastic-deterministic algorithm will eventually terminate in a local optimum for any user-defined ϵ-tolerance greater or equal to* 0.

The hybrid strategy discussed above has the potential to improve the performance of both the SM and the DSDA when solving flowsheet design problems. First, the hybrid algorithm in Figure 5.2 proposes a parallel execution of a deterministic and a stochastic algorithm that not only ensures local optimality for continuous variables, but also guarantees local optimality for discrete decisions within N_2-neighbors. To the authors' knowledge, these optimality guarantees cannot be provided by other hybrid algorithms available in the literature, see e.g., [131,133,137,138]. This parallel execution strategy also addresses some of the limitations that SMs or deterministic algorithms such as the D-SDA exhibit when executed separately. For instance, the inability of D-SDA to handle infeasible initializations is addressed by retrieving feasible solutions from the SM through set F_S, and the prohibitive evaluation of N_∞-neighborhoods for problems with many discrete decisions is addressed by delegating the N_∞-neighborhood exploration to the SM. Also, the slow convergence of the SM with respect deterministic strategies is addressed by continuously generating updated search bounds with the DSDA-VB method, and by defining a stopping criterion that incorporates stochastic and deterministic objective function values. In summary, the hybrid method takes advantage of the SM to identify feasible solutions at early stages of the execution (iteration $k = 0$) and explore a neighborhood of the current solution candidate (iterations $k \geq 1$). The SM heuristically seeks the global optimum of the problem within the

neighborhood it is exploring, but it does not provide optimality guarantees. The deterministic DSDA-VB strategy, on the other hand, uses information from the SM to update search limits and, under a deterministic approach, DSDA-VB satisfies the local optimality conditions in Definition 5.1 for the highly nonlinear model in Problem 5.1. This is expected to result in shorter computational times, as illustrated in section 5.3. It is also acknowledged that the proposed hybrid algorithm has limitations. The proposed hybrid strategy still needs improvements to handle multi-objective, discontinuous, and highly multimodal objective functions. Also, updating the search bounds of the SM using DSDA-VB helps the algorithm to converge faster. However, the bounds obtained using the proposed VB strategy only guarantee feasibility and local optimality; hence, any set of bounds returned by DSDA-VB may miss the global optimum. Detailed descriptions of the steps involved in the proposed parallel-hybrid algorithm are discussed below.

5.2.2.2 Steps followed by the stochastic algorithm

Steps S1 and S2.1: In *Step* S1, the stochastic algorithm begins by setting up Problem (5.1) aided with a process simulator for flowsheet calculations and initializing the algorithm using: 1) the user-defined variable bounds (i.e., $\boldsymbol{x}^{L,0}, \boldsymbol{x}^{U,0}, \boldsymbol{y}^{L,0}$ and $\boldsymbol{y}^{U,0}$) for Problem (5.1), 2) a value of $False$ is selected for control flow parameter $stop_S$, which will be latter used to define when to execute or reinitialize the SM, and 3) known feasible DOF combinations for set F_S, e.g., $F_S^0 = \emptyset$, indicating that no feasible solution is known for the problem. After that, an iterative procedure starts with the first execution of the SM in *Step S2.1*.

Step S3 and S4: According to *Step S3*, the execution of the SM keeps updating set F_S with new feasible solutions until the deterministic strategy determines that the execution of the SM must be stopped, i.e., until $stop_S$ takes a value of $True$. As described below in *Step D5*, stopping the SM occurs whenever the deterministic algorithm identifies an improvement in the best locally optimal solution found. Once $stop_S = True$, the current SM run is stopped as indicated in *Step S4*.

Steps S4 and S5: As indicated in *Step S4*, $stop_S$ is set back to $False$ and variable bounds in Problem (5.1) are updated using the modified bounds returned by the deterministic algorithm, denoted by purple dashed lines in Figure 2 ($\boldsymbol{x}^{L,M}, \boldsymbol{x}^{U,M}, \boldsymbol{y}^{L,M}$ and $\boldsymbol{y}^{U,M}$). One of the purposes of these new bounds is to guide the SM towards the best locally optimal solution identified by the deterministic strategy thus far, e.g., by keeping $\boldsymbol{y}^{L,M}$ and $\boldsymbol{y}^{U,M}$ within a N_∞-neighborhood of the best local solution. Before re-starting the execution of the SM using these new bounds, a global stopping criterion is verified in *Step S5*. This

global stopping criterion uses control flow parameter $Stop_D$, which stops the execution of both algorithms based on the best objective functions found by both the deterministic and stochastic algorithms. Flow parameters $Stop_D$ and $stop_S$ (red and green dashed lines in Figure 2) are controlled by the deterministic algorithm (*Steps D1-D7*), which is explained next.

5.2.2.3 Steps followed by the deterministic algorithm

Step D1: The deterministic strategy is constantly retrieving the most updated version of set F_S from either *Step S2*.1 or *Step S3* (see *Steps D1* and *D2*). In *Step D1*, the deterministic strategy must await until at least two different feasible solutions are available in F_S, which is a requirement to initialize the DSDA-VB strategy. These two feasible solutions from the stochastic technique are required to provide a feasible initialization ($[\boldsymbol{x}^I, \boldsymbol{y}^I]$) to *Step D3* and an initialization for continuous variables bounds ($[\boldsymbol{x}^{L,S}, \boldsymbol{x}^{U,S}]$) for the execution of DSDA-VB in *Step D4*. The deterministic strategy is then initialized in *Step D1* by assigning a value of $+\infty$ to the best locally optimal objective function known f_B^0, where subscript B stands for "best". Having $f_B^0 = +\infty$ means that no locally optimal solution has been detected yet. Then, the iteration count k is set at $k = 1$ and the iterations of the parallel hybrid algorithm start.

Steps D2 and D3: In *Step D2*, the solution with the best objective function value is retrieved from F_S and assigned to f_S^k, i.e., f_S^k is the best solution reported by the stochastic algorithm at iteration k. The DSDA-VB algorithm is then initialized at *Step D3* using the DOF associated to f_S^k (i.e., $[\boldsymbol{x}^I, \boldsymbol{y}^I]$), and the maximum ($[\boldsymbol{x}^{U,S}, \boldsymbol{y}^{U,S}]$) and minimum ($[\boldsymbol{x}^{L,S}, \boldsymbol{y}^{L,S}]$) DOF values in F_S.

Step D4 (The DSDA-VB): Selecting adequate variable bounds for deterministic NLP optimization is critical to avoid execution errors and improve solver performance, e.g., good bounds may avoid variable saturation, moving away from an optimal solution, or moving into regions with large function values, large derivative values, or singularities. Despite their relevance, variable bounds typically remain fixed, and the task of defining appropriate bounds is typically delegated to the user. To alleviate this issue, the DSDA-VB algorithm has been developed as an optimization method that iteratively executes the DSDA and a VB strategy that recomputes bounds for continuous variables ($\boldsymbol{x}^{L,M}, \boldsymbol{x}^{U,M}$). The VB strategy includes three distinctive features. First, it helps to circumvent potential convergence issues of the NLP solver. Second, it allows to keep the continuous variables returned by DSDA-VB (i.e., $\boldsymbol{x}^k$) away from their algorithmic bounds $\boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^{U,M}$; otherwise, a solution may be erroneously declared

as locally optimal. Third, the VB strategy helps to indirectly accelerate the SM, whose convergence rate may benefit from exploring a subset of the original search space that is known to contain at least one locally optimal solution. Note that bounds for discrete decisions do not need to be iteratively recalculated within DSDA-VB, because the D-SDA can consider the user-defined bounds $\boldsymbol{y}^{L,0}$ and $\boldsymbol{y}^{U,0}$ whenever it is executed, without affecting its performance. Instead, updated bounds of discrete decisions ($\boldsymbol{y}^{L,M}, \boldsymbol{y}^{U,M}$) are defined as upper and lower bounds of a discrete neighborhood of the local optimum identified by DSDA-VB. Note that these updated bounds of discrete DOF define a N_∞-neighborhood consisting of 3^{n_y} combinations of discrete DOF, which are considered by the SM whenever it is reinitialized. In contrast, each execution of the DSDA within DSDA-VB considers the smaller N_2-neighborhood of size $2n_y + 1$, given that the evaluation of N_∞-neighborhoods with D-SDA may be computationally prohibitive. This means that the SM is allowed to explore many discrete DOF combinations that may improve the objective function, but that may be ignored by the DSDA-VB. In summary, the DSDA-VB executed in *Step D4* of Figure 5.2 is an algorithm that not only optimizes the discrete and continuous DOF of the problem locally, but also returns new bounds for these variables as part of the solution.

The DSDA-VB returns an objective function (f_D^k) and the locally optimal solution found ($[\boldsymbol{x}^k, \boldsymbol{y}^k]$), as well as new variable bounds ($\boldsymbol{x}^{L,M}, \boldsymbol{x}^{U,M}, \boldsymbol{y}^{L,M}$ and $\boldsymbol{y}^{U,M}$) that are tighter than those originally defined by the user, i.e., $\boldsymbol{x}^{L,0} \leq \boldsymbol{x}^{L,M}, \boldsymbol{x}^{U,M} \leq \boldsymbol{x}^{U,0}, \boldsymbol{y}^{L,0} \leq \boldsymbol{y}^{L,M}$ and $\boldsymbol{y}^{U,M} \leq \boldsymbol{y}^{U,0}$. On the one hand, continuous variables bounds ($\boldsymbol{x}^{L,M}, \boldsymbol{x}^{U,M}$) guarantee that the continuous variables reported by DSDA-VB ($[\boldsymbol{x}^k, \boldsymbol{y}^k]$) are not at their bounds and that NLP convergence issues are avoided, e.g., discontinuities in the constraints that cause convergence errors when bounds are inappropriately selected are avoided. On the other hand, updated discrete variable bounds ($\boldsymbol{y}^{L,M}, \boldsymbol{y}^{U,M}$) are restricted within an N_∞-neighborhood of the locally optimal solution $\boldsymbol{y}^k$ returned by DSDA-VB as indicated in Eqs. (5.2A) and (5.2B), e.g., for a problem with a single discrete DOF, the improved bounds for optimal solution $y^k = 3$ are $y^{L,M} = \max(y^{L,0}, 2)$ and $y^{U,M} = \min(y^{U,0}, 4)$. The best objective function found by the deterministic algorithm (f_B^k) is updated after the execution of the DSDA-VB in *Step D4* as $f_B^k = \min_{\tilde{k}\in\{1,\ldots,k\}} \left(f_D^{\tilde{k}}\right)$. A detailed diagram and description of the DSDA-VB algorithm is provided in appendix E (Section E.1, Figure E.2).

$$\boldsymbol{y}^{L,M} = \max\left(\boldsymbol{y}^{L,0}, \min\left(N_\infty(\boldsymbol{y}^k)\right)\right) \tag{5.2A}$$

$$\boldsymbol{y}^{U,M} = \min\left(\boldsymbol{y}^{U,0}, \max\left(N_{\infty}(\boldsymbol{y}^{k})\right)\right) \tag{5.2B}$$

Steps D5, D6 and D7: The remaining steps (*Steps D5, D6* and *D7*) of the deterministic algorithm shown in Figure 5.2 decide when the stochastic algorithm must be reinitialized, and when both algorithms must be stopped through control flow parameters $stop_S$ and $stop_D$, respectively. *Step D5* decides if the SM must be stopped and reinitialized, based on an objective function improvement criterion that uses a small tolerance $\delta > 0$. If the best objective function found by the deterministic strategy in the current iteration (f_B^k) improved with respect to the best objective found in previous iterations (f_B^{k-1}), then control flow parameter $stop_S$ is set to $True$ indicating that the current SM run in the stochastic strategy must be stopped and reinitialized (see *Steps* S3 and S4 of the stochastic algorithm, which use the updated information from the red and purple dashed lines in Figure 5.2). This allows the stochastic algorithm to update the search bounds of the SM and guide it towards the solution with objective f_B^k. If the stochastic technique does not require to be reinitialized, then the global stopping criterion of both algorithms is evaluated in *Step D6*. This global stopping criterion verifies if the relative difference between the best solutions found by the stochastic (f_S^k) and deterministic (f_B^k) algorithms is below an ϵ-tolerance. If the global stopping criterion is satisfied, the stochastic technique has thus converged near the best local solution found by the deterministic algorithm, hence both algorithms stop (*Step* D7). Otherwise, a new iteration of the deterministic strategy begins.

5.3 Computational experiments

This section presents two case studies that highlight the improvements in terms of solution quality and computational times when using the proposed hybrid algorithm to optimize the total annual cost (TAC in $\$/year$) of intensified distillation flowsheets. In order to carry out the stochastic optimization process, the author has selected the Differential Evolution with Tabu List (DETL) as the SM. This algorithm combines the advantages of Differential Evolution (DE), a powerful global search algorithm, with the concept of tabu search, which is a local search method that prevents the algorithm from revisiting previously explored solutions [295]. Figure 5.3 illustrates the method used to implement the parallel algorithm proposed in this work. Two separate computers with a processor Intel® Core™ i7-3770 CPU @ 3.4 GHz and 16 GB of RAM were used to run the tasks related to the stochastic (computer 1) and deterministic (computer 2) algorithms. These computers were allowed to interact by exchanging information through a mappable network folder with 10 GB of storage capacity using Microsoft Excel.

The main codes of the hybrid algorithm (Figure 5.2) were implemented in Matlab. These codes were connected to an Excel spreadsheet that worked as an interface between Matlab and Aspen Plus, which was used for flowsheet simulation and NLP optimization with the SQP method. In the case of the stochastic algorithm, Excel was also used to run an implementation of the DETL code that was developed in previous works [296,297].

The rigorous Mass-Equilibrium-Summation-entHalpy (MESH) model with non-ideal equilibrium relationships available in Aspen plus was considered. Vector of constraints $\boldsymbol{g}$ in Problem (5.1) was handled by DETL using the penalty function method, with a high penalty factor of $1 * 10^{10}$ that is likely to avoid infeasible solutions. The traditional DETL method was used as a benchmark to compare the performance of the proposed hybrid algorithm. To perform a fair comparison, the tuning parameters of DETL were kept fixed for all runs of the hybrid and traditional DETL method, respectively. That is, both algorithms were executed under the same conditions. I adopted standard tuning parameters available in the literature for DETL [295,298]: population size $NP = 10 * (n_x + n_y)$, crossover rate $CR = 0.8$, mutation factor $F = 0.7$, tabu list size $TLS = \lceil NP/2 \rfloor$, and tabu radius $TR = 1 * 10^{-6}$. For the DSDA exploration, N_2-neighborhoods were considered.

The objective function considered for both case studies is shown in Eq. (5.3A), which was obtained using the estimation method presented by Douglas [299] and cost data available from previous works [138]. This function represents the addition of the annualized investment cost ($Investment_c$) and the operating costs associated with utilities ($Operating_c$) for each column ($c \in C$) in the flowsheet. The investment cost in Eq. (5.3B) considers the column diameter in meters (d_c), the total column height in meters (h_c), and the number of catalytic stages (nc_c) if any. The operating cost in Eq. (5.3C) represents the cost of the duties of the reboiler (Qr_c) and the condenser (Qc_c) in kilowatt. This objective function assumes 330 days of operation per year, an annual interest rate of 5%, and payback period of 5 years. Note that search bounds are imposed over d_c, h_c and nc_c whereas $|Qr_c|$ and $|Qc_c|$ are bounded below by 0, i.e., the objective function is bounded.

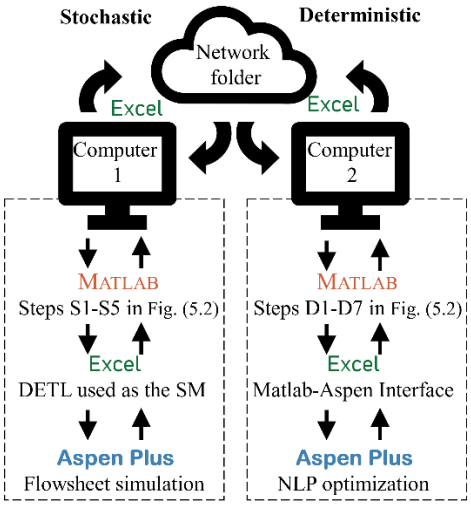


Figure 5.3. Hybrid algorithm implementation and software considered in this work.

$$f = TAC = \sum_{c \in C}(Invesment_c + Operating_c) \quad (5.3A)$$

$$Invesment_c = 10000 + 292.67(3.28d_c)^{1.066}(3.77h_c)^{0.802} + 15.29(3.28d_c)^{1.55}h_c + 131.74(d_c)^2 nc_c \quad (5.3B)$$

$$Operating_c = 146.8|Qr_c| + 24.5|Qc_c| \quad (5.3C)$$

The distillation column models considered in this work involve multiple discrete DOF, continuous DOF, and constraints resulting from multiple interconnected units in a process flowsheet. Rigorous stage-by-stage nonlinear models combining mass/energy balances and non-ideal equilibrium relationships for each unit are considered. The problems considered in this work are non-convex due to the nonlinear relationships between the input variables (such as feed flow rate, reflux ratio, and tray temperatures) and the output variables (such as product purity and recovery). Additionally, the large search spaces considered for discrete decisions and the nonlinear interactions between discrete and continuous variables make the problems described above challenging non-convex combinatorial optimization problems.

5.3.1 Case study 1: A thermally coupled system.

The optimization of the thermally coupled system shown in Figure 5.4 is considered. In this work, stages are numbered from top to bottom, i.e., the condenser corresponds to stage 1. The feed to the first column (C1) has a total molar flow rate of $\mathcal{F}_{total}^{FEED} = 45\ kmol/h$, with mole fractions of $\chi_{isobutane}^{FEED} = 0.4$, $\chi_{n-butane}^{FEED} = 0.2$ and $\chi_{n-pentane}^{FEED} = 0.4$. The first column aims to produce nearly pure isobutane at the top and nearly pure n-pentane at the bottoms. The separation of n-butane from the mixture is performed in another column (C2) that is thermally coupled to C1 through interconnecting vapor and liquid flows that are connected to the last stage of C2. These interconnecting flows avoid the need of a reboiler for the second column, decreasing energy consumption with respect to a traditional distillation sequence [272,300]. The packages available in Aspen Plus were used for thermodynamic calculations. The Chao-Seader property method with Lee-Kesler enthalpy was considered, which uses the Redlich-Kwong equation of state for vapor phase properties, and the Scatchard-Hildebrand model for liquid activity coefficients [301].

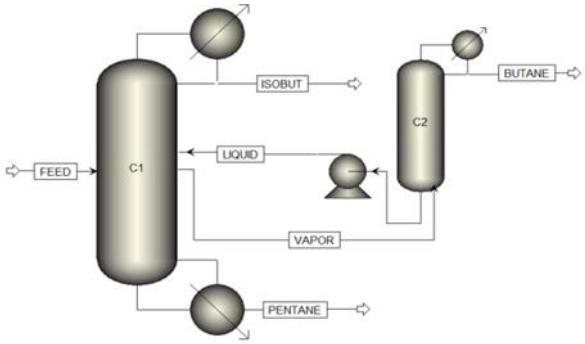


Figure 5.4. Thermally coupled system that separates a mixture of isobutane, n-butane and n-pentane.

The DOF of this problem ($\boldsymbol{x}$ and $\boldsymbol{y}$), and their corresponding bounds ($\boldsymbol{x}^{L,0}, \boldsymbol{x}^{U,0}, \boldsymbol{y}^{L,0}$ and $\boldsymbol{y}^{U,0}$) are summarized in Table 5.1. Note that the vapor flow that goes from C1 to C2 is assumed to be located one stage below the liquid flow that goes from C2 to C1. Other variables of interest that are retrieved from the optimization (i.e., those in vector $\boldsymbol{z}$) are the energy duties, which are required to calculate the objective function. Vector $\boldsymbol{z}$ also contains those variables required to specify the problem's constraints. These variables include the molar fraction and flow of the desired component for each output stream, i.e., molar flow ($\mathcal{F}_{isobutane}^{ISOBUT}$) and mole fraction ($\chi_{isobutane}^{ISOBUT}$) of isobutane at the top of C1; molar flow ($\mathcal{F}_{n-pentane}^{PENTANE}$) and mole fraction ($\chi_{n-pentane}^{PENTANE}$) of n-pentane at the bottoms of C1, and molar flow

($\mathcal{F}_{n-butane}^{BUTANE}$) and mole fraction ($\chi_{n-butane}^{BUTANE}$) of n-butane at the top of C2. The constraints considered for this problem ($\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})$) are shown in Equations (5.4A)-(5.4C), which establish that: 1) a minimum molar recovery of 94% must be attained for all components, 2) a minimum molar product purity of 98%, 99% and 96% is enforced for isobutane, n-pentane and n-butane respectively, and 3) the heigh to diameter ratio of the columns must be less or equal to 20 [302].

$$\mathcal{F}_{isobutane}^{ISOBUT} \geq 0.94\mathcal{F}_{isobutane}^{FEED}, \mathcal{F}_{n-pentane}^{PENTANE} \geq 0.94\mathcal{F}_{n-pentane}^{FEED}, \quad \mathcal{F}_{n-butane}^{BUTANE} \geq 0.94\mathcal{F}_{n-butane}^{FEED} \tag{5.4A}$$

$$\chi_{isobutane}^{ISOBUT} \geq 0.98, \chi_{n-pentane}^{PENTANE} \geq 0.99, \chi_{n-butane}^{BUTANE} \geq 0.96 \tag{5.4B}$$

$$h_c/d_c \leq 20, \forall c \in \{\mathrm{C1}, \mathrm{C2}\} \tag{5.4C}$$

Table 5.1. Variables and their bounds for the first case study, and best solution found.

Continuous DOF ($\boldsymbol{x}$)				
Variable	Units	Lower bounds ($\boldsymbol{x}^{L,0}$)	Upper bounds ($\boldsymbol{x}^{U,0}$)	Best solution found (Figure 5.5A)
Reflux ratio (C1)	-	0.5	10	7.62
Distillate flow (C1)	kmol/h	17.29	19.11	18.18
Distillate flow (C2)	kmol/h	8.69	9.6	8.92
Vapor flow (from C1 to C2)	kmol/h	11.34	34.02	18.84
Diameter (C1)	m	0.1	6	1.68
Diameter (C2)	m	0.1	6	0.37
Discrete DOF ($\boldsymbol{y}$)				
Variable	Units	Lower bounds ($\boldsymbol{y}^{L,0}$)	Upper bounds ($\boldsymbol{y}^{U,0}$)	Best solution found (Figure 5.5A)
Number of stages (C1)	-	5	70	55
Number of stages (C2)	-	5	70	12
Feed stage (C1)	-	2	69	24
Liquid feed stage (from C2 to C1)	-	2	69	46

This case study was used to test the proposed hybrid algorithm and to compare the performance of the proposed algorithm and the traditional DETL method. Given that both methods rely on random exploration, different solutions may be obtained from different runs of the same optimization problem. Thus, this case study was solved four times under the same conditions using both the proposed method and DETL. A relative optimality gap of $\epsilon = 0.05$ was established for the hybrid algorithm; also, δ was set to 0.05, and a CPU time limit of 5000 min was considered for both strategies. More details about the implementation and execution of the hybrid algorithm are available in sections E.1 and E.3 of appendix E.

Despite their inherent random variability, the four trials showed a similar qualitative performance, i.e., the parallel-hybrid method showed faster convergence and better objective function values than the traditional DETL. The evolution of the best objective function over time for these experiments is shown in Figures 5.5A-5.5D. Dotted lines represent the traditional DETL method, while solid and dashed lines represent the proposed DETL-based stochastic method and the proposed DSDA-VB deterministic method, respectively. The best solution identified by the hybrid algorithm is a local optimum and was obtained from the first trial (Figure 5.5A). This solution corresponds to a design with the DOF reported in the last column of Table 5.1, and it is compared with the best (local) solutions from the other runs (Figures 5.5B-5.5D) depicted in Table 5.2. The results from the different runs in Table 5.2 vary slightly, showing average percentage differences between the best-known solution (Figure 5.5A) and the remaining solutions (Figures 5.5B-5.5D) below 1% for all continuous DOF, except for the vapor flow from C1 to C2 and the diameter of C2 which show average percent differences of 1.3% and 2.7%, respectively. The discrete DOF also exhibits similar values between runs, differing only by increments or decrements of 1 stage with respect to the best-known solution. As a result, the resulting objective function values in Table 5.2 only differ by 0.05% on average compared to the best objective (271229 $/year).

Based on Figure 5.5, the proposed hybrid algorithm has a higher convergence speed than the traditional DETL for this case study. The hybrid strategy closed the gap after 2435 minutes on average, while the traditional DETL could not find the best solution reported by the hybrid algorithm after 5000 minutes in any of the trials. For instance, the DETL objective found after 5000 minutes in Figure 5.5A (275018 $/year) is still 1.4% larger than the solution obtained at 1825 minutes by the hybrid algorithm (271229 $/year). Also, when the hybrid algorithm stops in Figures 5.5A-5.5C, the objective function reported by the traditional DETL is on average 26% worse than the hybrid's strategy objective. These

improvements in convergence speed are mainly attributed to the "jump" that the hybrid algorithm performs once the first execution of the DSDA-VB algorithm ($k = 1$) is performed and the first group of improved variable bounds is returned to the SM. This is illustrated in Table 5.3, where after 209, 141, 150, and 252 minutes of execution of the hybrid algorithm, the DSDA-VB improves the DETL objective by 134%, 79%, 99% and 59% for the trials shown in Figure 5.5A, 5.5B, 5.5C, and 5.5D, respectively. Note that while the hybrid algorithm requires $\sim 10^2$ minutes to obtain the objective functions in column f_B^k shown in Table 5.3, the traditional DETL needs $\sim 10^3$ minutes to obtain solutions of this quality. Although the first iteration of the hybrid algorithm leads to a solution for which DOF are locally optimal, the objective function is further improved in subsequent iterations of the hybrid algorithm due to the continuous exploration of the DETL algorithm within the continuously refined bounds obtained from the DSDA-VB strategy. Additional information about the iterations of the hybrid algorithm for optimization runs in Figures 5.5A-5.5D is available in section E.5 of appendix E.

Table 5.2. Best solution found by each run of the hybrid algorithm (Figures 5.5B-5.5C).

Continuous DOF ($\boldsymbol{x}$)				
Variable	Units	Fig. 5.5B	Fig. 5.5C	Fig. 5.5D
Reflux ratio (C1)	-	7.40	7.64	7.61
Distillate flow (C1)	kmol/h	18.18	18.19	18.18
Distillate flow (C2)	kmol/h	8.92	8.92	8.92
Vapor flow (from C1 to C2)	kmol/h	18.99	19.29	18.95
Diameter (C1)	m	1.71	1.68	1.68
Diameter (C2	m	0.37	0.34	0.37
Discrete DOF ($\boldsymbol{y}$)				
Variable	Units	Fig. 5.5B	Fig. 5.5C	Fig. 5.5D
Number of stages (C1)	-	56	55	55
Number of stages (C2)	-	12	11	12
Feed stage (C1)	-	25	25	25
Liquid feed stage (from C2 to C1)	-	47	46	46
Objective function	$/year	271519	271339	271281

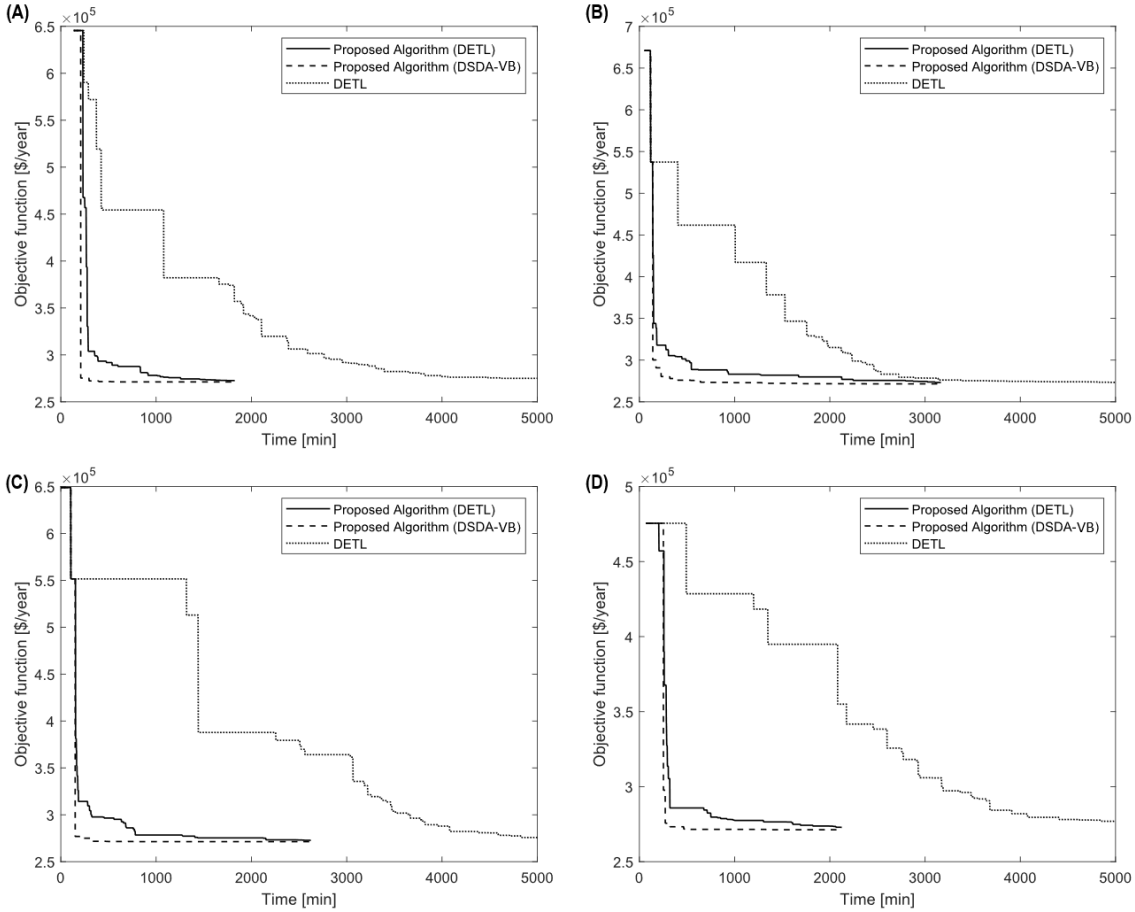


Figure 5.5. Convergence plots for the optimal design of the thermally coupled system.

Table 5.3. First iteration of the hybrid algorithm.

Run	Time [min]	Iteration (k)	f_B^k (DSDA-VB)	f_S^k (DETL)
Figure 5.5A	209	1	275351	645529
Figure 5.5B	141	1	300204.4	537369.4
Figure 5.5C	150	1	276903.6	551489.4
Figure 5.5D	252	1	297621.2	475507.7

The proposed hybrid algorithm is able to continuously identify new locally optimal solutions because the DETL explores solution candidates within N_∞-neighborhoods of discrete decisions of size $|N_\infty| = 3^4 = 81$, while DSDA-VB only guarantees local optimality within N_2-neighborhoods of size $|N_2| = 2(4) + 1 = 9$. N_2-neighbors consider the effect of varying one discrete decision at a time, while N_∞-

neighbors consider interactions between variables, e.g., modifying both the number of stages and feed location of a column simultaneously. As a result, DETL can guide the local search towards solution candidates that are near the current DSDA-VB solution but cannot be identified by this deterministic algorithm. For instance, Figure 5.6A illustrates the convergence plot of a discrete DOF: the number of stages N of column C1. The variables and bounds plotted in Figure 5.6 correspond to the best objective found by each algorithm at each iteration. The bounds for the number of stages in Figure 5.6A start with the user-defined bounds (i.e., $N^{L,0} = 5$ and $N^{U,0} = 70$), but these are updated to $N^{L,M} = 53$ and $N^{U,M} = 54$ after the first local solution with $N = 54$ is identified by DSDA-VB at 209 minutes (see the first row in Table 5.3). After 529 minutes of execution of the hybrid algorithm, the bounds over N are recalculated again; this time around $N = 55$, with $N^{L,M} = 55$ and $N^{U,M} = 56$. This change coincides with increments of the number of stages of C2 and the location of the liquid interconnection stage, i.e., multiple discrete decisions changed at 529 minutes, which is something that could not be achieved by the DSDA-VB alone, which explores the search space using N_2-neighborhoods ($|N_2| = 11$). This indicates that this new solution identified at 529 minutes was attained thanks to the exploration performed by DETL in the larger N_∞-neighbrohood ($|N_\infty| = 243$), which allowed varying multiple discrete DOF simultaneously. Note that although DSDA-VB only performs local optimality verifications within N_2-neighborhoods, it is allowed to explore the user-defined search region of discrete decisions ($\sim 10^7$ discrete solution candidates), while DETL seeks improved solutions within an N_∞-neighborhood of the current local DSDA-VB solution (81 discrete solution candidates).

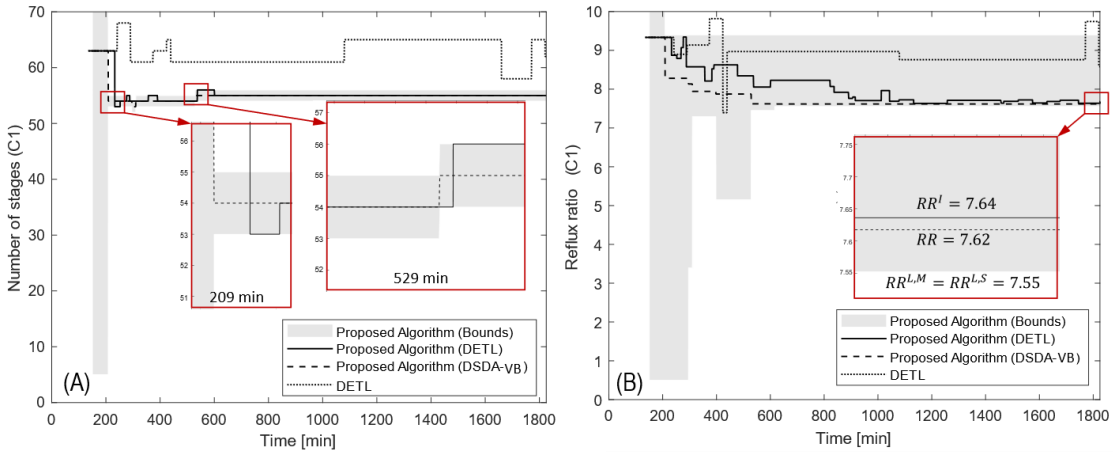


Figure 5.6. Illustrative convergence plots for the number of stages of C1 (A) and the reflux ratio of C1 (B). These plots correspond to the optimization run in Figure 5.5A.

In contrast to the discrete decisions, updating the bounds of the continuous variables does not necessarily limit the search of continuous variables within a vicinity of the current best solution. Figure 5.6B shows the behavior of the reflux ratio of unit C1. The upper bound of the reflux ratio begins at $RR^{U,0}$, and then it its updated and kept at $RR^{U,S}$, which is the maximum feasible reflux ratio stored in F_S. Contrarily, the VB strategy varied the lower bound of the reflux ratio ($RR^{L,M}$) during the first 32 iterations of the algorithm (i.e., the first 615 minutes) and then $RR^{L,M}$ remained constant at $RR^{L,S} = 7.55$. This avoided binding RR at its lower bound in the locally optimal solutions reported by D-SDA, i.e., after 615 minutes, $RR^{L,S} = 7.55$ guarantees that the best reflux ratio (7.62) is locally optimal, given that the optimized reflux ratio is no longer reaching its lower bound. Overall, updating bounds as illustrated in Figure 5.6 allows to guide the hybrid algorithm towards locally optimal solutions faster than the traditional DETL, which requires more time to converge for this case study.

Additional computational experiments showcasing the performance of the proposed parallel-hybrid algorithm against a nested-hybrid algorithm are presented in Figure E.5 of appendix E. The nested-hybrid strategy optimizes discrete DOF using the DETL, which iteratively fixes those discrete DOF and optimizes continuous DOF through the NLP solver available in Aspen. As shown in Figure S5 of the supplementary material, the nested-hybrid approach was 464 minutes on average slower than the parallel-hybrid method at finding the best solution reported by each algorithm. Also, the best parallel-hybrid solution (271229 $/year) is 0.2% better than the best nested-hybrid solution (271669 $/year) in terms of its objective function value. However, the nested-hybrid exhibits two key limitations: this method cannot refine search bounds for continuous DOF and cannot guarantee local optimality in the solutions. In contrast, the proposed parallel-hybrid approach addresses these issues offering both optimality guarantees and convergence improvements over the nested-hybrid methodology. More details about this comparison are provided in section E.4 of appendix E.

5.3.2 Case study 2: a sequence of reactive, extractive and conventional distillation columns

This case study aims to demonstrate the applicability of the proposed algorithm to a more challenging process flowsheet optimization problem that involves a large number of DOF and constraints, i.e., 13 discrete DOF and 15 continuous DOF. This case study was adapted from the work by Torres-Vinces et al. [303]. The sequence of distillation systems presented in Figure 5.7 consists of a reactive distillation (RD) column that has an input of pure 2,3-butanediol (2,3-BD) equal to $1000\ kg/h$. The pure 2,3-BD

follows a series of dehydration reactions in the RD column, which simultaneously generates Methyl Ethyl Ketone (MEK) as the main product in the top stream (T-RD) and separates the unreacted 2,3-BD as a bottom's product. The dehydration reactions in the RD system also generate by-products in the T-RD stream such as 2-methylpropanal (2MPL), 3-buten-2-ol (3B2OL), water, and some 1,3-butadiene (1,3-BD). These by-products are separated from the MEK in subsequent separation stages. The first stage is an extractive distillation (ED) column that uses glycerol as solvent to extract water from stream T-RD. The solvent mixed with water is removed from the bottom of the ED column (steam B-ED), while the remaining components in the mixture leave the system from the top (steam T-ED). The latter stream (T-ED) is further purified in distillation columns C1 and C2, where column C1 generates a bottom product rich in 3B2OL, and C2 generates a top product with 2MPL and some 1,3-BD, and a bottom product with purified MEK. Moreover, stream B-ED is separated in distillation column C3, which recovers the glycerol (bottom product) and produces water as the top product. The non-random two-liquid (NRTL) thermodynamic model was used for the liquid phase, while the Redlich-Kwong equation of state was used for the vapor phase [303].

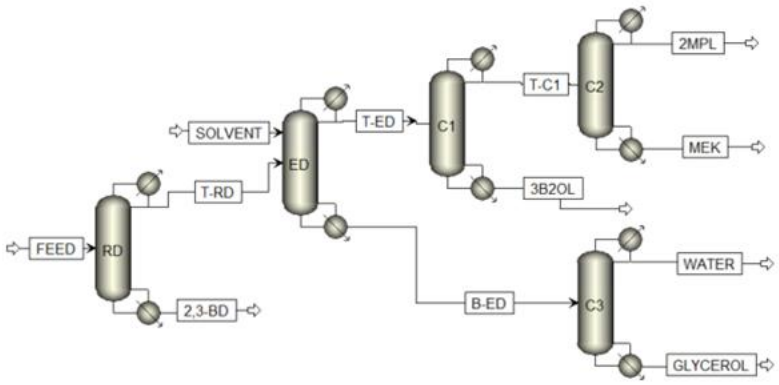


Figure 5.7. Sequence of distillation processes designed for the production of MEK from 2,3-BD, and the separation of by-products.

The DOF of this problem ($\boldsymbol{x}$ and $\boldsymbol{y}$), and their corresponding bounds ($\boldsymbol{x}^{L,0}, \boldsymbol{x}^{U,0}, \boldsymbol{y}^{L,0}$ and $\boldsymbol{y}^{U,0}$) are summarized in Table 5.4. Variables in vector $\boldsymbol{z}$ include the reboiler and condenser duties for all columns, as well as those variables required to specify the problem's constraints, i.e., the mass fraction (ω_{comp}^{str}) and mass flow ($\mathcal{M}_{comp}^{str}$) of the desired component $comp$ for each output stream str. The user-defined constraints that are considered for this problem ($\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$) include those proposed by Torres-

Vinces et al.[303]: a purity constraint of at least 99.5% (wt) is enforced over the MEK output (Eq. (5.5A)); purity constraints of at least 99% (wt) are considered for the 2,3-BD, 3B2OL, and 2MPL streams (Eq. (5.5B)); purity constraints of at least 98% and 99.99% (wt) are imposed over the water and glycerol output streams, respectively (Eq. (5.5C)), and a recovery of at least 98% (wt) is enforced for all products (Eq. (5.5D)). Also, the heigh to diameter ratio of the columns is constrained to be less or equal to 20, as stated in Eq. (5.5E) [302]. In addition, hydrodynamic constraints that aim to maintain proper column hydrodynamic operation (e.g., enough contact between vapor and liquid phases) such as entrainment and downcomer flooding are imposed for all columns. These constraints are handled internally by Aspen Plus, i.e., these constraints are included in the black-box constraints $\widehat{\Omega}$ that appear in problem (5.1). Note that this case study is more challenging than the problems usually considered to test the hybrid algorithms for flowsheet optimization, which usually consider flowsheets involving one or two processing units as in case study 1, e.g., see [33,135,138].

$$\omega_{MEK}^{MEK} \geq 0.995 \tag{5.5A}$$

$$\omega_{comp}^{str} \geq 0.99, \quad \forall (comp, str) \in \{(2{,}3-BD, 2{,}3-BD), (3B2OL, 3B2OL), (2MPL, 2MPL)\} \tag{5.5B}$$

$$\omega_{Water}^{WATER} \geq 0.98, \omega_{Glycerol}^{GLYCEROL} \geq 0.9999 \tag{5.5C}$$

$$\mathcal{M}_{comp}^{str} \geq 0.98 \mathcal{M}_{comp}^{T-RD}, \quad \forall (comp, str) \in \begin{Bmatrix} (MEK, MEK), (2MPL, 2MPL), (3B2OL, 3B2OL), \\ (Water, WATER), (Glycerol, GLYCEROL) \end{Bmatrix} \tag{5.5D}$$

$$h_c/d_c \leq 20, \forall c \in \{\mathrm{RD}, \mathrm{ED}, \mathrm{C1}, \mathrm{C2}, \mathrm{C3}\} \tag{5.5E}$$

Due to the high number of variables and constraints involved, this problem requires much more time to converge than the previous case study. In the present case study, the proposed hybrid algorithm and the DETL strategy were left to run until a time limit of $5 * 10^4$ minutes was reached with $\delta = 0.05$. The convergence plot of both algorithms is shown in Figure 5.8. At $5 * 10^4$ minutes, the relative gap between the traditional DETL and hybrid algorithms objectives is approximately 48%, while the gap between the DETL and DSDA-VB algorithms within the proposed hybrid approach is around 1%. The

best solution reported by the hybrid algorithm has an objective function of $863423\ \$/year$; the DOF obtained for this solution is depicted in the last column of Table 5.4. In contrast, the traditional DETL found a much more expensive design with an objective function of $1282080\ \$/year$.

Table 5.4. Variables and their bounds for the second case study, and best solution found.

Continuous DOF ($\boldsymbol{x}$)				
Variable	Units	Lower bounds ($\boldsymbol{x}^{L,0}$)	Upper bounds ($\boldsymbol{x}^{U,0}$)	Best solution found
Reflux ratio (RD)	-	1	3.2	3.016
Reflux ratio (ED)	-	0.1	0.5	0.489
Reflux ratio (C1)	-	1	14.7	1.306
Reflux ratio (C2)	-	1	5.8	5.607
Reflux ratio (C3)	-	1	1.6	1.000
Distillate flow (from RD to ED)	kg/h	994.01	1004.00	999.258
Distillate flow (from ED to C1)	kg/h	792.57	800.53	797.241
Bottoms flow (C1)	kg/h	89.55	90.45	90.344
Bottoms flow (C2)	kg/h	434.82	439.19	437.230
Bottoms flow (C3)	kg/h	397.01	401.00	399.989
Diameter (RD)	m	0.5	3.5	2.408
Diameter (ED)	m	0.5	3.5	1.372
Diameter (C1)	m	0.5	3.5	2.286
Diameter (C2)	m	0.5	3.5	1.372
Diameter (C3)	m	0.5	3.5	0.500
Discrete DOF ($\boldsymbol{y}$)				
Variable	Units	Lower bounds ($\boldsymbol{y}^{L,0}$)	Upper bounds ($\boldsymbol{y}^{U,0}$)	Best solution found
Number of stages (RD)	-	76	86	79
Number of stages (ED)	-	45	55	45
Number of stages (C1)	-	75	85	75
Number of stages (C2)	-	45	55	45
Number of stages (C3)	-	13	23	13
Feed stage (RD)	-	40	50	49
T-RD feed stage (ED)	-	35	44	38
Solvent feed stage (ED)	-	2	10	2
Feed stage (C1)	-	45	55	53
Feed stage (C2)	-	20	30	20
Feed stage (C3)	-	9	12	10
Location of first reactive stage (RD)	-	2	7	5
Location of last reactive stage (RD)	-	75	85	78

Due to the problem size and the search bounds selected, the performance of the traditional DETL deteriorated significantly for this case study (see the dotted line in Figure 5.8). Although DETL was able to identify feasible solutions during the early stages of its execution, it was unable to improve the objective function within the specified time limit. This performance may be attributed to the low ratio of individuals evaluations that converged to feasible solutions, which in this case resulted in stagnation of the traditional DETL algorithm, i.e., at $5 * 10^4$ minutes the DETL population remains diverse and does not seem to be progressing towards a particular solution. In contrast, the hybrid algorithm managed to handle this convergence problem by automatically identifying variable bounds that, for this case study, allowed the DETL algorithm to improve its objective function and satisfy local optimality requirements for discrete and continuous variables (see solid line in Figure 5.8).

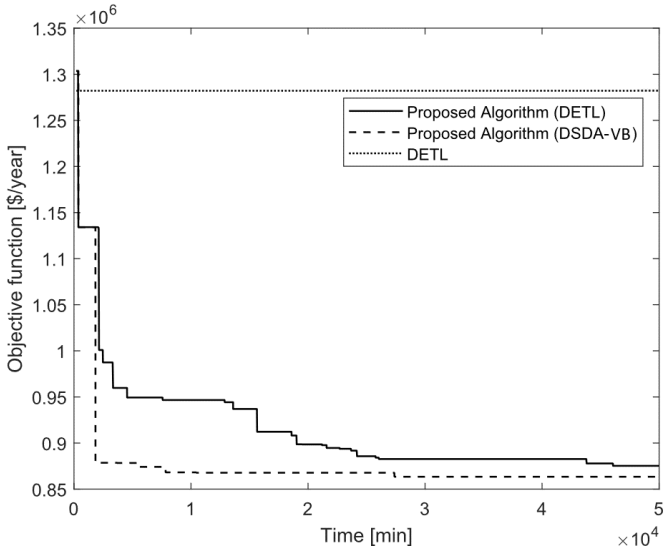


Figure 5.8. Convergence plot for the MEK production process.

Table 5.5 summarizes some of the iterations performed by the hybrid algorithm for this case study. These iterations correspond to those where variable bounds are updated. Each iteration shows the locally optimal objective function obtained by DSDA-VB (f_B^k), the best objective function identified by DETL (f_S^k), and the bounds and optimal solution for the number of stages ($N^{L,M}, N^{U,M}, N^k$) and reflux ratio ($RR^{L,M}, RR^{U,M}, RR^k$) of the RD. According to Table 5.5, the frequency at which variable bounds are updated decreases as iterations progress, e.g., the first group of updated bounds is obtained after 1833 minutes at $k = 1$, while the last group of updated bounds is obtained after 16926 minutes at

$k = 272$. Note that the hybrid algorithm spent 45% of the total execution time between iterations 272 and 442 (i.e., 22622 minutes). During this time interval, DSDA-VB could not identify better solutions, thus bounds were not updated, i.e., between iterations 272 and 442 DETL is aiming to improve the objective function without receiving feedback from DSDA-VB. This suggests that, although the DETL performance improves with respect to its naïve execution, DETL is still the time-limiting factor when integrated with DSDA-VB. Concerning the convergence of variables and their bounds, the behavior obtained is similar to that obtained for the first case study. For example, the lower bounds are updated from the user-defined values (76 and 1) to 80 and 2.96 at $k = 1$, for N and RR, respectively. After the first iteration, variable bounds and values do not change significantly, but they still have an impact on the objective function, e.g., f_B^k and f_S^k improve by 1.8% and 30% between iteration 1 and 442. Detailed data showing the evolution of the objective functions f_B^k and f_S^k is provided in section E.5 of appendix E.

Table 5.5. Convergence of objective functions f_B^k and f_S^k, number of stages (RD) and reflux ratio (RD) for iterations in which variable bounds are updated.

				Number of stages (RD)			Reflux ratio (RD)		
Time [min]	k	f_B^k	f_S^k	$N^{L,M}$	$N^{U,M}$	N^k	$RR^{L,M}$	$RR^{U,M}$	RR^k
1833	1	878540.4	1134177	80	82	81	2.96	3.04	3.01
3692	49	878356.7	987397	80	82	81	2.93	3.03	2.97
5473	56	874157.8	959781.5	80	82	81	2.47	3.04	2.93
7826	161	868193.2	946637	79	81	80	2.89	3.04	2.95
10452	168	867731.8	946637	79	81	80	2.90	3.03	2.97
27378	272	863423.2	882641.1	78	80	79	2.89	3.04	3.02
50000	442	863423.2	875273.2	78	80	79	2.89	3.04	3.02

Chapter 6
Optimal Design and Dynamic Transitions of Multitask Catalytic Distillation Columns

Continuous chemical processes are typically designed to keep a single product at a desired specification; however, allowing multiproduct operation and chemical production at multiple grades is often required to meet customer and market requirements. Furthermore, there exist processes that allow multigrade/multiproduct production in a single unit or process, by performing dynamic transitions between different operating conditions, e.g., polymers such as polyethylene and polypropylene are produced at different melt indexes, by performing grade changeover operations [304–306]. Dynamic modelling and optimization approaches for the design of intensified processes for multitask production are scarce in the literature [307–309]. According to [28,309–316], multitask units are expected to comply with a demand-driven operation that should avoid off-grade production and high transition costs; thus, considering the multipurpose and dynamic behavior of these systems should be an aspect that needs to be considered at the conceptual design stage. The integration of dynamic considerations with process design in RD/CD has been considered in recent works, through the integration of design and control using both sequential [317–319] and simultaneous approaches [320]. The design and control studies available in the literature usually focus on the design of single product RD/CD units, coupled with control design for disturbance rejection or set-point tracking performance. However, works that consider multipurpose operation for the design of RD/CD systems are uncommon, as discussed next.

Reactive distillation has been considered as an attracting unit to allow multitask dynamic production at both experimental and computational levels [27,28,310,321–327]. While there are studies that have considered the optimal design and dynamic transitions involved in multipurpose RD/CD columns, steady state design and process dynamics have always been treated separately [28,310,323–327]. The work by López-Negrete de la Fuente and Flores-Tlacuahuac was one of the first to study dynamic transitions between products in RD using an optimization framework [324]. They studied optimal and empirical transition policies for the start-up and dynamic transitions in reactive distillation, applied to the metathesis of 2-pentene to form 2-butene and 3-hexene. Shah studied the process of polyester synthesis by continuous RD [310]. In that work, the feasibility of multigrade production was demonstrated, while design and optimal production times were studied and optimized sequentially. At the design step, a reactive distillation column that produces two grades of polyester was designed to produce one of the

products first. Since reaction conditions are similar for both products, it was assumed that the same equipment could be used for the second grade. Ramírez-Márquez et al. studied the dynamic behavior of a RD column for the production of multiple silane products [323]. For the design of the unit, different column designs for each desired product were optimized first using a sensitivity analysis approach. The multiproduct column was designed as an overdesigned column using information from the previous single-product designs. Note that in both [310] and [323], the design step was performed without considering dynamics, based on either similarity between products or overdesign considerations.

Regardless of its simplicity, a sequential approach for the optimal design and dynamic transitions of a multitask RD columns does not necessarily guarantee to obtain an optimal design [323,325]. For instance, designing an overdesigned column at steady state does not necessarily guarantee optimality in the operating variables, e.g., reboiler duties, for the different products; also, the column designed at steady state may not be optimal during dynamic operation. Moreover, a design resulting from a sequential approach may be infeasible, i.e., the column may not cover the entire range for process inputs required to achieve the desired product specifications. Rules of thumb are often used to find the nearest feasible design when this occurs, e.g., by adjusting discrete and continuous design variables through sensitivity analysis until a column that can produce every desired product is obtained. To partially overcome these difficulties, Medina-Herrera et al. developed an approach to sequentially design and evaluate the performance of a RD unit subject to dynamic transitions [28,325]. That work proposed an intermediate step in the sequential design process, which consists of using an overdesigned column to initialize an optimization problem that considers every product simultaneously at steady state [325]. That simultaneous steady state design problem solves an economic optimization, where production scenarios are given *a priori*. Those scenarios were determined by known production requirements given by the market, and they are independent of the production order. Through this chapter, this formulation will be referred to as a steady state *multi-scenario* (also called multi-period) formulation. As a result, Medina-Herrera et al. [325] concluded that their design approach may still result in constraint violations and that the design obtained may have implications in transitions times and the economic performance of the process. Also, those authors highlighted the need of an optimal design strategy capable of considering optimal dynamic transitions at an early design stage [28]. Thus, the current state of the art suggests that more rigorous design strategies are still required to integrate both design and transitions optimization into a single step at the conceptual design stage.

This chapter introduces the first deterministic framework that suits the needs of a challenging intensified problem: the optimal design of a catalytic distillation (CD) system, using a formulation that considers the multitask and dynamic behavior of the system. This feature differentiates this work from previous contributions in this area, which have addressed similar problems using stochastic methods, e.g., see [28,325]. The methodology proposed in this chapter is based on the simultaneous economic optimization of the CD unit design, the steady state operation of the process, and the corresponding dynamic transitions between the multiple products/grades. For this purpose, this study does not rely on conventional heuristic-based and trial and error design strategies; instead, the goal is to obtain a design that can be proven to be locally optimal while using a mechanistic process model involving first-principles conservation equations. To the authors' knowledge, this is the first study that explicitly incorporates process dynamics into the optimal design of multitask RD/CD columns, thereby adding novelty to this study. Furthermore, this is the first work that proposes a deterministic methodology that allows to guarantee local optimality when designing a multiproduct/multigrade RD/CD process, either at steady-state or in the dynamic domain.

Through this chapter, I propose a general dynamic *multi-scenario* design approach that allows to simultaneously optimize the design and transient conditions of a multitask RD column. In contrast to the multi-scenario steady state model presented in [325], the dynamic multi-scenario model considered in this work is not constrained to a particular case study and will be posed as a Mixed-Integer Dynamic Optimization (MIDO) problem, which cannot be solved to optimality using conventional MINLP optimization strategies. The resulting formulation contains nonlinearities and rigorous phenomenological models that describe the behavior of the phases within the CD unit, e.g., higher order and heterogeneous kinetic models, non-ideal vapor and liquid mixtures, hydraulic correlations for the calculation of pressure drop, hydrodynamic operation constraints, geometric relationships between design variables, etc. Moreover, these constraints are inherently propagated through time due to the inclusion of process dynamics at the process design stage. Furthermore, the addition of discrete design variables such as the number of trays, location of catalytic trays and feed location imposes nonlinear interactions between continuous and discrete variables. Therefore, a new deterministic optimization framework specifically formulated to handle this optimization problem is studied. This optimization strategy is based on the D-SDA which was recently found to be highly efficient to solve optimal CD and extractive distillation design problems at steady state, both in terms of optimal solution quality and convergence speed [7,122]. The D-SDA is a promising framework to handle CD superstructures that

consider the transient operation of this process, though the method still requires further developments to incorporate dynamics into the optimal design problem. This challenge is properly addressed in this work with an improved version of the D-SDA that considers dynamic transitions, through the implementation of a pruning strategy. The D-SDA with pruning is referred through this chapter as the enhanced D-SDA. Furthermore, a modular strategy is developed in this chapter to guarantee local optimality when process dynamics are considered. The first two modules do not consider dynamics and allow to find a local optimal solution fast, while the third module relies on the enhanced D-SDA to find a locally optimal design that ensures dynamic feasibility and optimality in the operation.

Given that D-SDA tools have only been used to solve single-product steady-state problems in chemical engineering, e.g., see [4,7,122], this study presents the first application of a MINLP optimization strategy based on DCA to solve Mixed-Integer nonlinear Dynamic Optimization (MIDO) *multi-scenario* design problems. The MIDO multi-scenario problem is solved using deterministic optimization. Hence, the modular strategy proposed in this chapter can provide local optimality guarantees and avoid drawbacks often found in stochastic optimization, e.g., the selection of case-study tunning parameters and the possibility of finding different solutions when running the algorithm multiple times. A key feature of the proposed modular design framework is that it incorporates the enhanced D-SDA, which incorporates a pruning strategy that can significantly reduce the required CPU times, which is a key contribution of this research work.

This chapter is organized as follows: In section 6.1, a general description of the MIDO problem addressed in this work is presented. Then, section 6.2 explains the methodology used to solve this problem, based on an improved D-SDA that can handle MIDO problems for this class of systems. The proposed approach is tested using an Ethyl Tert-Butyl Ether (ETBE) case study presented in section 6.3. Results from the case study are presented in section 6.4.

6.1 Problem statement

In this section, a problem statement for the economic optimal design of continuous processes with dynamic transitions is presented. The problem consists of simultaneously finding an economically optimal design and operating conditions of a CD column that produces multiple products (or grades) of a single product, by performing dynamic transitions between different steady states. The general problem statement is shown in Eq. (6.1), where the objective is to minimize the equivalent annual cost for the CD column, subject to the process model and the operational product quality constraints. Both

continuous design and operation ($\boldsymbol{v}$) and binary ($\boldsymbol{y}$) vectors of variables are considered. $\boldsymbol{v}$ includes continuous variables related to design variables, e.g., diameter, stage height, column height, weir length and height, as well as process operation variables such as condenser and reboiler duties, reflux ratio, boil up rate, compressibility factors, kinetic variables, enthalpies, vapor and liquid hold ups, flows, temperatures and compositions along the column, etc. Vector $\boldsymbol{y}$ contains those binary variables used to represent discrete design decisions in a distillation superstructure formulation. Thus, these discrete decisions allow to specify the total number of stages, the location of feedstreams and the reactive/catalytic stages within the column. The constraints are represented by the feasible sets S and $Y \subseteq \{0,1\}^{n_y}$. Set S contains every constraint over $[\boldsymbol{v}, \boldsymbol{y}]$ (both equalities and inequalities) that describes the steady state and dynamic operation of the process and that appear in either phenomenological or empirical constraints, i.e., mass and energy balances, sum of mole fractions equal to one, equations of state, hydraulic correlations, kinetic expressions, saturation limits on the variables, etc. Note that vector $\boldsymbol{y}$ is also part of the feasible set S to account for the interactions between continuous and discrete design variables, e.g., binary variables must deactivate equilibrium equations for those leftover stages that were considered in the superstructure formulation but are not part of the final column design. Set Y defines the search region for the binary variables. For instance, logic constraints imposed over the superstructure are included in Y, e.g., reactive stages must be kept in between the reflux and the boil up stages.

$$\min_{\boldsymbol{v},\boldsymbol{y}} \left(C_{INV} + \int_0^{t_p} (C_{OP}(t) - C_{IN}(t)) dt \right) \tag{6.1}$$

$$s.t.$$

$$[\boldsymbol{v}, \boldsymbol{y}] \in S, \boldsymbol{y} \in Y$$

In the objective function shown in Eq. (6.1), C_{INV} $[\$/year]$ is the annualized investment cost, $C_{OP}(t)$ $[\$/min]$ are the time dependent operation costs, whereas $C_{IN}(t)$ $[\$/min]$ are the time dependent incomes of the process[f]. For instance, if the output product flow calculated from the model is introduced in the objective in mol/min, then the multiplication of this flow by its selling price in $\$/mol$ will result in incomes $C_{IN}(t)$ with units of $\$/min$. A finite time horizon $[0, t_p]$ must be selected

[f] There is a need to consider costs with different time units. C_{INV} is in $\$/year$ using an annualizing factor (AF in $[-]/year$). Both $C_{OP}(t)$ and $C_{IN}(t)$ appear inside an integral with the operating factor (t_p in $min/year$) as an upper limit that represents the yearly operation of the process in minutes (min) per year. Thus, $C_{OP}(t)$ and $C_{IN}(t)$ are in $\$/min$ such that the integral term in Eq. (1) is: $(\$/min)(min/year) = \$/year$.

to perform the dynamic optimization of the process; this time horizon includes every dynamic transition and the periods of steady-state operation that will be optimized. A typical value of t_p used for continuous process is $t_p = 480{,}000\ min/year$, to guarantee an allotment rate of 91% [328,329]. The remaining downtime is usually left for maintenance or catalyst replacement/regeneration [329].

In the following section, a deterministic methodology that allows to efficiently solve the optimization problem in Eq. (6.1) is presented. This approach is restricted to continuous processes that perform transitions between different steady states during operation. For these processes, steady state information can be disaggregated from dynamic considerations. Thus, this approach cannot be directly applied to batch systems. Moreover, the framework is limited to the optimization of a single operation unit whose output stream changes from one product to another in a sequential fashion. The processing units used to handle off-grade products in-between transitions are not considered in the present analysis. Hence, it is assumed that off-grade production can be used for blending purposes or recycled to other sections of a chemical plant. Moreover, closed loop operation is not considered; thus, this work assumes that it is possible to design a controller that can follow the control actions obtained from optimal open-loop control solutions. Since I am using a rigorous dynamic model that is expected to make reasonable predictions of an actual process, a model-based multivariable controller such as a Model Predictive Controller could be designed and expected to accurately follow the optimal open-loop control actions obtained from the proposed framework.

6.2 Methodology

The optimization problem in Eq. (6.1) indicates that the equivalent annual cost of the process will be optimized over a user defined optimization time span given by $[0, t_p]$. This life span is expected to be realistic, e.g., set t_p to $480{,}000\ min/year$. Therefore, the resulting dynamic optimization problem will consider a large number of constraints and variables to represent the dynamic behavior of the process over $[0, t_p]$. Also, the current formulation in Eq. (6.1) does not allow to distinguish steady state production periods from dynamic production periods, whose times are usually minimized to avoid off-grade production [28,309–316,324]. As shown in section 6.2.1, the objective function has been adjusted such that only sections of $[0, t_p]$ are considered for processes with repetitive sets of dynamic transitions (i.e., a wheel-type of continuous process) and to recognize that the dynamics of the process can be neglected during steady state production periods. These modifications also allow to include transition times into the problem's formulation. Section 6.2.2 presents a dynamic *multi-scenario* formulation that complies

with the modifications in the objective function, where each scenario represents a time slot with a dynamic transition and a steady state production period. These modifications in the objective function and the constraints allow to formulate a simpler optimization problem at steady state that disregards dynamic variables and constraints; this problem is referred to as the steady state *multi-scenario* problem (see section 6.2.3). Solving this simpler problem simultaneously with the original MIDO problem is beneficial when implementing a discrete search procedure based on neighborhood comparison like that performed by the D-SDA. This is because solving steady state *multi-scenario* subproblems provides useful information to the dynamic *multi-scenario* subproblems, which allows to decrease the combinatorial complexity of the dynamic problem, i.e., it decreases the number of feasible discrete combinations to be evaluated. In particular, the steady state *multi-scenario* problem continuously generates lower bounds for the dynamic *multi-scenario* problem, thus adding cuts to the feasible region of discrete decisions and decreasing the computational time (see theorem presented in section 6.2.4). In turn, these developments serve as a basis for the proposed solution procedure: the enhanced D-SDA, which will be discussed in section 6.2.4.

6.2.1 Objective function

The objective function in Eq. (6.1) can be rewritten to consider that the operation within $[0, t_p]$ may be cyclical, with a sequence of tasks that repeats w times over the period $[0, t_p]$. Thus, $[0, t_p]$ can be divided into w intervals $[0, t_p/w] \cup [t_p/w, 2t_p/w] \ldots \cup \left[(w-1)t_p/w, t_p\right]$. It is assumed that every interval has the same initial and final state, therefore the value of the integral for every interval is the same and Eq. (6.1) can be rewritten as Eq. (6.2). Thus, the cyclical operating behavior can be modeled with variables over $[0, t_p/w]$ at the expense that additional constraints must be imposed to the problem (see section 6.2.2).

$$\min\left(C_{INV} + w\int_0^{\frac{t_p}{w}} C_{OP}(t) - C_{IN}(t)dt\right) \tag{6.2}$$

The interval $[0, t_p/w]$ can be further divided into subintervals $[t_0, t_1] \cup [t_1, t_2] \ldots \cup [t_{n_K-1}, t_{n_K}]$ that represent n_K time slots, with $t_0 = 0$ and $t_p/w = t_{n_K}$ (see the illustrative example in Figure 6.1). These time slots may include product changeovers, start-ups, shut-downs; each with their corresponding steady-state production periods. As a convention, each subinterval is divided into a dynamic transition period that goes from t_{k-1} until t_k^{TR}, followed by a steady-state production period until t_k, i.e.,

$[t_{k-1}, t_k] = \left[t_{k-1}, t_k^{TR}\right] \cup \left[t_k^{TR}, t_k\right], \forall k \in K = \{1,2,3,\dots,n_K\}$. The term t_k^{TR} is treated as an optimization variable that represents the point in time when the system's transients have decayed[g]. Henceforth, these subintervals will be denoted as time slots or *scenarios*, e.g., $k = 1$ represents the first time slot/scenario. Since there are many subintervals involved, this formulation is referred to as the *multi-scenario* formulation. Thus, Eq. (6.2) is rewritten as follows:

$$\min\left(C_{INV} + w\sum_{k\in K}\int_{t_{k-1}}^{t_k^{TR}} C_{OP}(t) - C_{IN}(t)dt + \int_{t_k^{TR}}^{t_k} C_{OP}(t) - C_{IN}(t)dt\right) \tag{6.3}$$

There are transition costs during dynamic periods; the product cannot generate revenue during these periods because it has not achieved its desired specification. Thus, $\int_{t_{k-1}}^{t_k^{TR}} C_{IN}(t)dt = 0$. Also, operating costs and incomes remain constant at $C_{OP}(t_k)$ and $C_{IN}(t_k)$ during the steady state production stage. Under these considerations, and after a rearrangement of the objective function, Eq. (6.3) can be reformulated as follows:

$$\min f_{OBJ} = f_{OBJ}^{SS} + f_{OBJ}^{TR} \tag{6.4.A}$$

$$f_{OBJ}^{SS} = C_{INV} + w\sum_{k\in K}\left[\left(C_{OP}(t_k) - C_{IN}(t_k)\right)(t_k - t_{k-1})\right] \tag{6.4.B}$$

$$f_{OBJ}^{TR} = w\sum_{k\in K}\left[\int_{t_{k-1}}^{t_k^{TR}}(C_{OP}(t) - C_{OP}(t_k) + C_{IN}(t_k))dt\right] \tag{6.4.C}$$

Note that the objective function in Eqs. (6.4.A)-(6.4.C) is only the result of an algebraic rearrangement from Eq. (6.3); hence, there were no further assumptions considered. As shown in Eqs. (6.4.A) and (6.4.B), f_{OBJ}^{SS} is referred to as the steady-state section of the objective function since it ignores the process dynamics. On the other hand, the function f_{OBJ}^{TR} (Eq. (6.4.C)) represents the dynamic section of the objective function. Note that f_{OBJ}^{SS} (Eq. (6.4.B)) takes into account the transition costs in the expression: $C_{OP}(t_k)\left(t_k^{TR} - t_{k-1}\right)$. This is an approximation since C_{OP} must be calculated considering the process dynamics over the time span $[t_{k-1}, t_k^{TR}]$, i.e., $\int_{t_{k-1}}^{t_k^{TR}} C_{OP}(t)dt$. This approximation is used in Eq. (6.4.B) to account for the transition costs in the steady-state calculation.

[g] Note that $t_k^{TR} - t_{k-1}$ is the time needed to move from one operating point to another (i.e., a dynamic transition).

Nevertheless, f_{OBJ}^{TR} (Eq. (6.4.C)) adds the actual transition costs of the process through the integral $\int_{t_{k-1}}^{t_k^{TR}} C_{OP}(t)dt$. Note that f_{OBJ}^{TR} (Eq. (6.4.C)) excludes the approximated transition costs considered in Eq. (6.4.B) through the expression: $\int_{t_{k-1}}^{t_k^{TR}} -C_{OP}(t_k)dt = -C_{OP}(t_k)\left(t_k^{TR} - t_{k-1}\right)$. Moreover, the present study assumes that there is no product revenue during the dynamic transitions. To account for this condition, f_{OBJ}^{SS} (Eq. (6.4.B)) assumes that the potential revenue generated by the product sold during transitions is subtracted from this cost function through the term: $-C_{IN}(t_k)\left(t_k^{TR} - t_{k-1}\right)$. Similarly, f_{OBJ}^{TR} (Eq. (6.4.C)) considers that products cannot be sold during transitions through the term $\int_{t_{k-1}}^{t_k^{TR}} C_{IN}(t_k)dt = C_{IN}(t_k)\left(t_k^{TR} - t_{k-1}\right)$. That is, $-C_{IN}(t_k)\left(t_k^{TR} - t_{k-1}\right)$ (from f_{OBJ}^{SS}) plus $C_{IN}(t_k)\left(t_k^{TR} - t_{k-1}\right)$ (from f_{OBJ}^{TR}) is equal to 0, indicating that there is no revenue during dynamic transitions. This form of the objective function is motivated by the fact that the first term f_{OBJ}^{SS} is independent of variables related to the dynamic performance of the process and only depends on the following steady state information: investment costs C_{INV}, steady state operating costs $C_{OP}(t_k), \forall k \in K$ and revenues $C_{OP}(t_k), C_{IN}(t_k), \forall k \in K$. Note that $t_k, \forall k \in K$ and w are herein treated as parameters while transition times $t_k^{TR}, \forall k \in K$ are considered as decision variables.

The new theoretical foundations developed in this work are based on the usual conjecture that $f_{OBJ}^{TR} \geq 0$, i.e., dynamic transitions represent an economic cost. Note that in cases where $\int_{t_{k-1}}^{t_k^{TR}} C_{OP}(t_k)dt \leq \int_{t_{k-1}}^{t_k^{TR}} C_{OP}(t)dt\,, \forall k \in K$, the objective function is further simplified to $f_{OBJ}^{TR} = w\sum_{k\in K}\left[\int_{t_{k-1}}^{t_k^{TR}} C_{IN}(t_k)dt\right]$, and the conjecture becomes a proposition. On the contrary, if $\int_{t_{k-1}}^{t_k^{TR}} C_{OP}(t_k)dt > \int_{t_{k-1}}^{t_k^{TR}} C_{OP}(t)dt\,, \forall k \in K$, then the analysis becomes more complicated. Nevertheless, it seems highly unlikely to obtain a better solution for an optimal design problem when its dynamics are included than when dynamics are excluded, because dynamic transitions are usually expected to generate additional costs due to the off-specified production period, as discussed in previous works [28,309–314,330]. A reason for this common assumption is that chemical processes are designed to be profitable; hence, $C_{IN}(t_k)$ is generally greater than $C_{OP}(t_k)$, which implies that $f_{OBJ}^{TR} \geq 0$ [331]. Another reason for $f_{OBJ}^{TR} \geq 0$ is that both $\int_{t_{k-1}}^{t_k^{TR}} C_{OP}(t_k)dt$ and $\int_{t_{k-1}}^{t_k^{TR}} C_{OP}(t)dt$ are approximations of the transition costs; therefore, they could return similar numerical values. Hence, the absolute value

of their difference is expected to be much smaller than $\int_{t_{k-1}}^{t_k^{TR}} C_{IN}(t_k)dt$. Either way, having $f_{OBJ}^{TR} \geq 0$ seems to be a reasonable conjecture. The benefits of isolating f_{OBJ}^{SS} from f_{OBJ}^{TR} are illustrated in sections 6.2.3 and 6.2.4.

6.2.2 Constraints in the feasible region *S*

The objective of this section is to introduce the constraints that appear in the feasible region S shown in Eq. (6.1) by following the *multi-scenario* representation shown previously. The variables of the problem can be divided into different categories. First, there are state variables that depend on time denoted by $[\boldsymbol{\chi}(t), \boldsymbol{\zeta}(t)]$, with differentiable variables $\boldsymbol{\chi}(t) = [\chi_1(t), \chi_2(t), \dots, \chi_{n_\chi}(t)] \in \mathbb{R}^{n_\chi}$ and algebraic variables $\boldsymbol{\zeta}(t) = \left[\zeta_1(t), \zeta_2(t), \dots, \zeta_{n_\zeta}(t)\right] \in \mathbb{R}^{n_\zeta}$. Also, there are time dependent manipulated variables or inputs $\boldsymbol{\mu}(t) = \left[\mu_1(t), \mu_2(t), \dots, \mu_{n_\mu}(t)\right] \in \mathbb{R}^{n_\mu}$. In addition, there are continuous variables that are independent of time, and these are represented by the vector $\boldsymbol{\varphi} = \left[\varphi_1, \varphi_2, \dots, \varphi_{n_p}\right] \in \mathbb{R}^{n_\varphi}$. Binary variables are described by $\boldsymbol{y} = [y_1, y_2, \dots, y_{n_y}] \in \{0,1\}^{n_y}$. Both $\boldsymbol{\varphi}$ and $\boldsymbol{y}$ are included to account for the effect of design variables, either continuous or discrete. The first group of constraints that define S are shown in Eqs. (6.5.A)-(6.5.C), which pose the problem as a Mixed-Integer Dynamic Optimization (MIDO) problem.

$$\boldsymbol{g}^D(\boldsymbol{\varphi}, \boldsymbol{y}) \leq 0 \tag{6.5.A}$$

$$\boldsymbol{h}^{SS}(\boldsymbol{\chi}(t_k), \boldsymbol{\mu}(t_k), \boldsymbol{\zeta}(t_k), \boldsymbol{\varphi}, \boldsymbol{y}) \leq 0, \quad \forall k \in K \tag{6.5.B}$$

$$\boldsymbol{h}^{TR}\left(\boldsymbol{\chi}(t), \frac{d\boldsymbol{\chi}(t)}{dt}, \boldsymbol{\mu}(t), \boldsymbol{\zeta}(t), \boldsymbol{\varphi}, \boldsymbol{y}\right) \leq 0, \quad \forall t \in \left(t_{k-1}, t_k^{TR}\right], \forall k \in K \tag{6.5.C}$$

The constraints are separated into design constraints over discrete and continuous design variables (vector functions $\boldsymbol{g}^D$ in Eq. (6.5.A)), steady state constraints (vector function $\boldsymbol{h}^{SS}$ in Eq. (6.5.B)) and dynamic constraints (vector function $\boldsymbol{h}^{TR}$ in Eq. (6.5.C))[h]. Constraints in $\boldsymbol{g}^D$ are time independent and define the dimensions of the distillation equipment, e.g., length to diameter ratio, available catalyst space, definition of active and downcomer areas, etc. $\boldsymbol{g}^D$ also includes those equations that connect

[h] Note that equality constraints can be written in terms of inequalities, thus, this formulation is general.

discrete decisions with continuous design variables, e.g., the total column height is defined in terms of the total number of stages and the stage height. In addition, time dependent modeling constraints are included in $\boldsymbol{h}^{SS}$ and $\boldsymbol{h}^{TR}$, where $\boldsymbol{h}^{SS}$ imposes restrictions over steady state times, i.e., $t_k \forall k \in K$, while $\boldsymbol{h}^{TR}$ imposes similar constraints during dynamic transition periods, i.e., $\left(t_{k-1}, t_k^{TR}\right], \forall k \in K$. Both $\boldsymbol{h}^{SS}$ and $\boldsymbol{h}^{TR}$ have in common that they consider the conservation equations that characterize the system, e.g., mass and energy balance over distillation trays both at steady ($\boldsymbol{h}^{SS}$, with time derivatives equal to 0) and dynamic ($\boldsymbol{h}^{TR}$) states. Also, both $\boldsymbol{h}^{SS}$ and $\boldsymbol{h}^{TR}$ include those intermediate expressions that represent physical phenomena involved in the process, e.g., calculation of compressibility coefficients, activity coefficients, reaction and phase equilibrium constant, reaction rates, vapor and liquid enthalpies, etc. Moreover, there are constraints that appear in $\boldsymbol{h}^{SS}$ but not in $\boldsymbol{h}^{TR}$ and vice versa. For instance, product purity requirements that enforce a desired composition over a product are only required during steady state operation ($\boldsymbol{h}^{SS}$), while off-specified products are allowed during dynamic transitions. Similarly, constraints over process' inputs that allow the implementation of a control policy are only required during transient operation ($\boldsymbol{h}^{TR}$). These constraints are shown in Eqs. (6.6.A)-(6.6.C), and these enforce that a digital system performs calculations periodically (discretely) during finite sampling periods, while control actions remain constant between sampling points [332,333].

$$\alpha_{k,j} = t_{k-1} + \sum_{j'=1}^{j} \Delta t_{k,j'}, \forall k \in K, \forall j \in J_k, \alpha_{k,n_{J_k}} = t_k^{TR}, \forall k \in K \tag{6.6.A}$$

$$\boldsymbol{\mu}\left(\alpha_{k,j}\right) = \boldsymbol{\mu}(t), \forall k \in K, \forall j \in J_k, \forall t \in \left(\alpha_{k,j} - \Delta t_{k,j}, \alpha_{k,j}\right) \tag{6.6.B}$$

$$\Delta \boldsymbol{t}_{min,k} \leq \Delta \boldsymbol{t}_k \leq \Delta \boldsymbol{t}_{max,k}, \forall k \in K \tag{6.6.C}$$

Eqs. (6.6.A) and (6.6.B) are included to ensure that control actions remain constant over a finite sampling period, where the size of this period may be bounded above and below as described in Eq. (6.6.C). In Eq. (6.6.A), $\alpha_{k,j}$ represents a sampling point $j \in J_k = \{1,2,3, \ldots, n_{J_k}\}$, where J_k is the set of sampling points of scenario k; $\Delta t_{k,j}$ is the size of the sampling interval of sampling point j and scenario k, and $\Delta \boldsymbol{t}_k = \left[\Delta t_{k,1}, \Delta t_{k,2}, \ldots, \Delta t_{k,n_{J_k}}\right]$ (see a graphic representation of these concepts in Figure 6.1). Note that I also define $\alpha_{k,n_{J_k}} = t_{k-1} + \sum_{j \in J_k} \Delta t_{k,j} = t_k^{TR}$ to link t_k^{TR} with $\Delta t_{k,j}$. Eq. (6.6.A) allows to define Eq. (6.6.B), which can be interpreted as a zero-order hold that keeps input variables constant

over sampling intervals. The size of the sampling interval is bounded as described by Eq. (6.6.C). Having $\Delta \boldsymbol{t}_k, \forall k \in K$ as variables allows to consider as decision variables the times at which the transitions may occur ($t_k^{TR}, \forall k \in K$).

In addition to the expressions defined above, the dynamic *multi-scenario* formulation also requires additional constraints in S to couple the different scenarios, by ensuring that the time-dependent variables of the problem remain continuous over time, e.g., the product composition profile must be continuous over $[0, t_p]$. Hence, additional constraints are still required to guarantee that the system will be near enough to the desired steady state t_k at $t_k^{TR}, \forall k \in K$. Thus, switchability constraints for each scenario (Eqs. (6.7.A) and (6.7.B)) and continuity constraints between scenarios (Eqs. (6.7.C) and (6.7.D)) are introduced, i.e.,

$$\left(\frac{d\chi_i}{dt}\Big|_{t_k^{TR}}\right)^2 \leq \varepsilon_1, \forall k \in K, \forall i \in \{1,2,\dots,n_\chi\} \tag{6.7.A}$$

$$\boldsymbol{\mu}\left(t_k^{TR}\right) = \boldsymbol{\mu}\left(t_k^{TR} - \Delta t_{k,n_{J_k}}\right), \forall k \in K \tag{6.7.B}$$

$$\boldsymbol{\mu}\left(t_k^{TR}\right) = \boldsymbol{\mu}(t_k), \forall k \in K \tag{6.7.C}$$

$$-\varepsilon_2 \leq \left[\boldsymbol{\chi}_I\left(t_k^{TR}\right), \boldsymbol{\zeta}_I\left(t_k^{TR}\right)\right] - \left[\boldsymbol{\chi}_I(t_k), \boldsymbol{\zeta}_I(t_k)\right] \leq \varepsilon_2, \forall k \in K \tag{6.7.D}$$

$$\boldsymbol{\mu}(t_0) = \boldsymbol{\mu}(t_k), k = n_K \tag{6.7.E}$$

$$[\boldsymbol{\chi}(t_0), \boldsymbol{\zeta}(t_0)] = [\boldsymbol{\chi}(t_k), \boldsymbol{\zeta}(t_k)], k = n_K \tag{6.7.F}$$

The first two constraints, Eqs. (6.7.A) and (6.7.B), are based on a framework to evaluate the switchability characteristics of a process design [334]. The first constraint, Eq. (6.7.A), allows differential variables to become approximately 0 at t_k^{TR}, by selecting a small enough (positive) value of ε_1. That is, for a given time slot k, ε_1 is small enough to allow $[\boldsymbol{\chi}(t), \boldsymbol{\zeta}(t)]$ to remain approximately constant over $[t_k^{TR}, t_k]$, or at least to guarantee that after t_k^{TR} the system's variables and derivatives will converge to their steady state at t_k, while keeping the inputs $\boldsymbol{\mu}(t)$ constant from t_k^{TR} until t_k. Eq. (6.7.B) keeps the last two control actions constant and allows to make the optimization procedure more efficient [334].

These equations are combined with Eq.(6.7.C) to account for the *multi-scenario* property of the proposed optimization model. Eq. (6.7.C) indicates that control actions remain constant from t_k^{TR} to t_k, i.e., within time slots. Eq. (6.7.D) impose additional constraints that complement Eq. (6.7.C) for problems involving multiple steady states, i.e., Eq. (6.7.D) must be imposed only if multiplicity occurs in the process (see [335] for more details about the "multiplicity" concept). The purpose of Eq. (6.7.C) is to acknowledge that a system may have multiple steady states at t_k, for the same steady state input $\boldsymbol{\mu}(t_k)$. Thus, $\boldsymbol{\chi}_I(t_k)$ and $\boldsymbol{\zeta}_I(t_k)$ are those state variables in $[\boldsymbol{\chi}(t), \boldsymbol{\zeta}(t)]$ that, when fixed at steady state, guarantee that there is only one possible steady state for the system. Consequently, the value for these variables at the end of the transition period, i.e., $\left[\boldsymbol{\chi}_I\left(t_k^{TR}\right), \boldsymbol{\zeta}_I(t_k^{TR})\right]$, must be within a ε_2-neighborhood of $[\boldsymbol{\chi}_I(t_k), \boldsymbol{\zeta}_I(t_k)]$ to guarantee an agreement between the steady states achieved at t_k^{TR} with Eq. (6.5.C) ($\boldsymbol{h}^{TR}$) and the steady states defined by Eq. (6.5.B) ($\boldsymbol{h}^{SS}$) (see for example the composition profile in Figure 6.1). Note that Eq. (6.7.D) is not defined as an equality constraint since the time derivatives at t_k^{TR} are not exactly 0 due to Eq. (6.7.A); therefore, the system is not expected to be exactly at the desired steady state. Instead, the independent state variables are forced within the ε_2-neighborhood that is expected to drive the system towards the steady state imposed by Eq. (6.5.B) ($\boldsymbol{h}^{SS}$). For systems without multiple steady states, Eq. (6.7.D) is not required in the formulation, although constraints of this type can be used to ensure that some variables achieve their desired steady state values with tolerance ε_2. For instance, the quality and quantity of the process' product may be expected to satisfy their corresponding specification within some limits at t_k^{TR}, in order to make the products saleable and satisfy demand requirements. Furthermore, Eqs. (6.7.E) and (6.7.F) are used to specify that the initial conditions of the system are equal to their corresponding final state. Note that Eqs. (6.7.E) and (6.7.F) must be imposed for processes with frequency values w greater than 1, for which dynamic transitions occur cyclically. If $w = 1$, the system may be forced to start and finish its operation at the same state using these constraints; otherwise, Eqs. (6.7.E) and (6.7.F) must be replaced by the expected initial conditions of the system. In summary, Eqs. (6.5.A)-(6.7.F) guarantee that at t_{k-1} and from t_k^{TR} to t_k the system is at the desired steady state for every scenario k and that continuity is satisfied between each time slot and between operation cycles. A schematic example of the *multi-scenario* formulation with special emphasis in the feasible region S is shown in Figure 6.1.

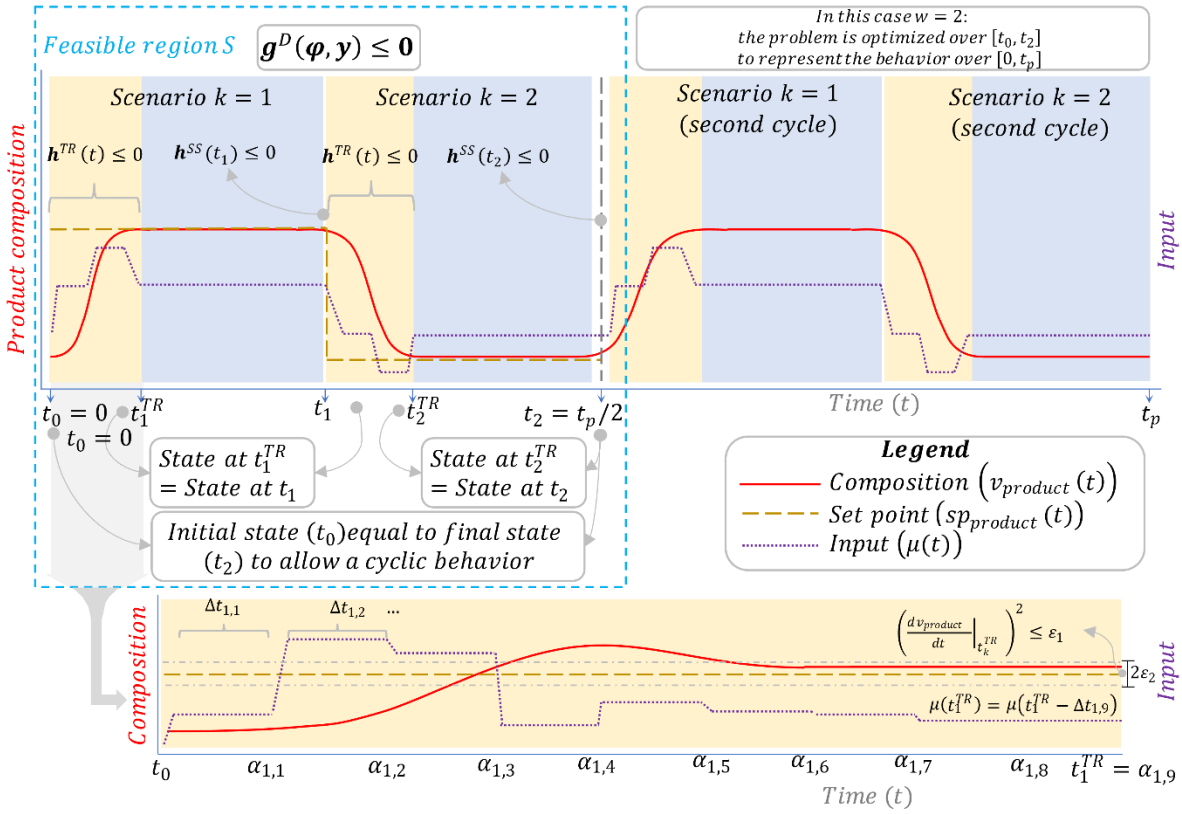


Figure 6.1. Schematic example of the dynamic multi-scenario formulation.

6.2.3 Problem partitioning and special features

The optimization problem defined by the objective function in Eq. (6.4.A) and constraints in Eqs. (6.5.A)-(6.5.C),(6.6.A)-(6.6.C) and (6.7.A)-(6.7.F) is computationally challenging. This is due to the size of the problem with multiple dynamic optimization problems that must be solved simultaneously and discrete variables that inherently make the problem non-convex[i]. Nevertheless, this problem has a particular structure; i.e., the special form of the discrete variables for distillation column design and the fact that the objective function and its constraints can be partitioned (under certain assumptions) into a steady state domain and a dynamic domain.

The problem statement of Eq. (6.1) is now rewritten as in Eq. (6.8), where the vector of continuous variables is $[\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}]$, with a vector of time dependent variables at steady state ($\boldsymbol{v}^{SS} = [\boldsymbol{v}_1^{SS}, \boldsymbol{v}_2^{SS}, \dots, \boldsymbol{v}_{n_K}^{SS}]$), a vector of time dependent variables at transient state for each scenario ($\boldsymbol{v}^{TR} =$

[i] according to the usual definition of convexity [36].

$[\boldsymbol{v}_1^{TR}, \boldsymbol{v}_2^{TR}, \dots, \boldsymbol{v}_{n_K}^{TR}, \boldsymbol{\chi}(t_0), \boldsymbol{\mu}(t_0), \boldsymbol{\zeta}(t_0)]$), and the previously defined time independent variables ($\boldsymbol{\varphi}$). The scenario-dependent vectors in $\boldsymbol{v}^{SS}$ and $\boldsymbol{v}^{TR}$ are defined as $\boldsymbol{v}_k^{SS} = [\boldsymbol{\chi}(t_k), \boldsymbol{\mu}(t_k), \boldsymbol{\zeta}(t_k)] \forall k \in K$ and $\boldsymbol{v}_k^{TR} = \left[\boldsymbol{\chi}\left(\left(t_{k-1}, t_k^{TR}\right]\right), \boldsymbol{\mu}\left(\left(t_{k-1}, t_k^{TR}\right]\right), \boldsymbol{\zeta}\left(\left(t_{k-1}, t_k^{TR}\right]\right), \Delta\boldsymbol{t}_k, t_k^{TR}\right], \forall k \in K$[j]. The feasible region S for the variables of the problem is partitioned into two subregions. The first subregion (S^{SS}) comprises Eqs. (6.5.A) and (6.5.B), which are only affected by the values of $\boldsymbol{v}^{SS}, \boldsymbol{\varphi}$ and $\boldsymbol{y}$. The second subregion S^{TR} includes the remaining constraints defined by Eqs. (6.5.C) and (6.7.A)-(6.7.F). Recall that Eqs. (6.6.A)-(6.6.C) are part of the vector function $\boldsymbol{h}^{TR}$ defined in Eq. (6.5.C).

Dynamic multi-scenario optimization problem (6.8)

$$\min_{\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}, \boldsymbol{y}} f_{OBJ} = f_{OBJ}^{SS}(\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}) + f_{OBJ}^{TR}(\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR})$$

$$s.t.$$

$$[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}] \in S^{SS}, [\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}, \boldsymbol{y}] \in S^{TR}, \boldsymbol{y} \in Y$$

S^{SS}: Eqs. (6.5.A) and (6.5.B)

S^{TR}: Eqs. (6.5.C) and (6.7.A)-(6.7.F)

The first special feature of the optimization problem in Eq. (6.8) is the form of the discrete design variables $\boldsymbol{y}$. For CD design, either binary or Boolean variables are used to integrate the nonlinear balance equations of the problem with the selection of the total number of trays, the location of feeds, the location of catalytic stages, etc. From the point of view of existing MINLP and logic-based optimization solvers, the representation of these discrete decisions with binary/Boolean variables does not take into consideration the physical meaning of these discrete decisions. For instance, the notion of the total number of stages is not considered when using a binary or Boolean formulation; instead, the selection of the number of trays is implicitly represented as a set of units in a superstructure that may activate or deactivate the presence of mass transfer or equilibrium equations between phases. This situation motivated previous works to develop the reformulation with *external variable*, as an strategy capable of recognizing the ordered nature of the discrete decisions arising in distillation design, and using this special feature to apply a D-SDA that aims to search for local solutions faster than existing MINLP optimization approaches [4]. In this work, it is assumed that the independent binary variables ($\boldsymbol{y}_I$) in $\boldsymbol{y}$ can be reformulated with external variables ($\boldsymbol{e}$) as $\boldsymbol{y}_I = \boldsymbol{y}_I(\boldsymbol{e})$, and therefore every binary

[j] Notation $\boldsymbol{\chi}\left(\left(t_{k-1}, t_k^{TR}\right]\right)$ indicates that variables in vector function $\boldsymbol{\chi}(t)$ are evaluated over the range $(t_{k-1}, t_k^{TR}]$.

variable can be expressed in terms of the external variables, i.e., $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{y}_I(\boldsymbol{e}))$. This is possible for distillation design since discrete decisions are typically binary variables that define aspects such as the number of stages and the location of feeds. Note that the proposed approach can be extended to other superstructure systems with similar characteristics. For instance, systems such as dividing wall columns or extractive columns can also be considered since the external variables reformulation can be readily applied to those systems.

The D-SDA applies first a reformulation step over those binary variables that were used to represent integer decisions at the modeling stage. The resulting reformulated discrete variables are denoted as *external variables* which may include, e.g., number of stages, feed location, catalytic stages' location, etc. For instance, the feed location, which is commonly modeled using binary variables over multiple prospective stages, is reformulated into a single *external variable* that explicitly indicates the feed input stage. The D-SDA then decomposes the optimization problem into master and subproblem optimization layers. In the master layer, the feasible region of the *external variables* is explored using a discretized version of the continuous steepest descent algorithm. In the subproblem layer, the objective function of the upper layer is calculated by solving the resulting optimization problem with *external variables* fixed at the current value, e.g., an NLP optimization solved using conventional NLP solvers. As a stopping criterion, the D-SDA verifies discrete neighborhoods to select the direction of steepest descent and to check for local optimality. For instance, consider that the optimization routine is currently at a feasible solution with *external variables* fixed at the value indicated by vector $\boldsymbol{e}$. Then, a neighborhood is defined as those fixed combinations of *external variables* that surround $\boldsymbol{e}$. If a neighboring combination of discrete variables near $\boldsymbol{e}$ is better than any feasible solution inside a neighborhood of $\boldsymbol{e}$, the optimizer continues the search along the steepest descent direction (defined by $\boldsymbol{\delta}$), until there is no further improvement. Then, the neighborhood verification and the steepest descent optimization is repeated for a new combination of *external variables*. If there is no direction of improvement, the algorithm stops, and the current solution is declared as the local optimum. The neighborhood used during the search is user-defined; however, points within a ∞-norm of $\boldsymbol{e}$, i.e. $N_\infty(\boldsymbol{e})$, have been consistently used to guarantee local optimality in MINLP problems [4,7]. This is because a ∞-norm can be used to guarantee global optimality for the majority of the discrete functions studied in discrete convex analysis, e.g., integrally convex functions, separable convex functions, $L^\natural$-convex functions, $M^\natural$-convex, etc. [6]. Nonetheless, caution must be taken with the CPU usage of problems with many external variables, e.g., preliminary tests have shown that time may increase significatively for steady state design problems

involving more than 8 external variables. This is because the size of $N_\infty(\boldsymbol{e})$ discrete neighborhoods increases exponentially as the number of external variables increases [6]. Accordingly, alternative reduced neighborhoods are recommended to decrease the computational costs, at the expense of weaker optimality guarantees for the solution [4].

The second feature of the optimization problem presented in Eq. (6.8) is the nature of its objective function (Eq. (6.4.A)). As indicated above, f_{OBJ}^{SS} only depends on steady state information, i.e., f_{OBJ}^{SS} depends on $C_{INV} = f(\boldsymbol{\varphi}, \boldsymbol{y}), C_{OP}(t_k) = f(\boldsymbol{\chi}(t_k), \boldsymbol{\mu}(t_k), \boldsymbol{\zeta}(t_k)), C_{IN}(t_k) = f(\boldsymbol{\chi}(t_k), \boldsymbol{\mu}(t_k), \boldsymbol{\zeta}(t_k)), \forall k \in K$. The minimization of f_{OBJ}^{SS} is equivalent to the steady state optimal design of a multiproduct/multigrade CD unit, subject to steady state and dynamic constraints for the different scenarios; however, f_{OBJ}^{SS} only depends on steady state and design variables $[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}]$. Thus, time dependent transient variables ($\boldsymbol{v}^{TR}$) do not affect the steady state decisions in the objective function. Consequently, to reduce the complexity of this problem, dynamic decisions are not considered when performing the sole minimization of f_{OBJ}^{SS}. This optimization problem is shown in Eq. (6.9), which will be useful to propose a general methodology to solve the dynamic *multi-scenario* optimization problem.

Steady-state multi-scenario optimization problem (6.9)

$$\min_{\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}} f_{OBJ}^{SS}(\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y})$$

$$s.t.$$

$$[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}] \in S^{SS}, \boldsymbol{y} \in Y$$

As discussed in chapter 3 and recent works, the D-SDA is an efficient deterministic approach to solve the optimal deign of single-product CD and extractive columns, modeled with rigorous nonlinear models [7,122]. Thus, the main purpose of the next section is to present an enhanced version of the D-SDA, specifically designed to solve the simultaneous design and dynamic transitions problem presented in section 6.1. The improved D-SDA takes advantage of the ordered nature of discrete decisions in CD design, which also allows to exploit the special characteristic of the objective function presented in Eq. (6.4.A). As a result, the rigorous solution of the optimization problem presented in Eq. (6.8) becomes tractable.

6.2.4 Improved D-SDA

This section presents the developments performed in the D-SDA to address the optimal solution of problems involving dynamic decisions, as shown in Eq. (6.8). To apply the D-SDA, the optimization problem must be decomposed as shown in Eqs. (6.10.A) and (6.10.B), where binary variables are reformulated with external variables $\boldsymbol{e} = \left[e_1, e_2, \dots, e_{n_e}\right] \in E$, with $E \subseteq \mathbb{Z}^{n_e}$. Feasible region E is an equivalent representation of Y, in the field of the external variables, i.e., $\boldsymbol{e} \in E \Leftrightarrow \boldsymbol{y}(\boldsymbol{e}) \in Y$. Thus, if it is not possible to formulate E as a tight representation of Y (e.g., $\boldsymbol{e} \in E$ but $\boldsymbol{y}(\boldsymbol{e}) \notin Y$), then $\boldsymbol{y}(\boldsymbol{e}) \in Y$ should be kept in the formulation of the master problem in Eq. (6.10.A). The objective function of Eq. (6.10.A) is $f_{SUB}(\boldsymbol{e}) = f_{OBJ}(\boldsymbol{v}^o, \boldsymbol{y}(\boldsymbol{e}))$, which is the optimum value of the subproblem in Eq. (6.10.B) found at $\boldsymbol{v}^o = [\boldsymbol{v}^{SSo}, \boldsymbol{v}^{TRo}, \boldsymbol{\varphi}^o]$ with $\boldsymbol{e}$ fixed. As a convention, $f_{SUB}(\boldsymbol{e}) = +\infty$ for subproblems declared as infeasible either by the master problem or the NLP solver in the subproblem.

Master problem (6.10.A)

$$\min_{\boldsymbol{e}} f_{SUB}(\boldsymbol{e})$$

$$s.t.$$

$$\boldsymbol{e} \in E, \boldsymbol{y}(\boldsymbol{e}) \in Y$$

Dynamic multi-scenario subproblem (6.10.B)

$$f_{SUB}(\boldsymbol{e}) = \min_{\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}} f_{OBJ}(\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e}))$$

$$s.t.$$

$$[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})] \in S^{SS}, [\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})] \in S^{TR}$$

In this work, the conjecture that $f_{OBJ}^{TR} \geq 0$ is fundamental to propose an efficient optimization strategy to solve the simultaneous design and dynamic transitions problem. This conjecture allows to formulate Theorem 6.1 (proof available section F.1 of appendix F), which enables the computation of a lower bound for the optimization subproblem shown in Eq. (6.10.B).

Theorem 6.1 *Consider the following problem, which provides a solution for the steady state objective function* ($f_{SUB}^{SS}(\boldsymbol{e}) = f_{OBJ}^{SS}(\boldsymbol{v}^{SS*}, \boldsymbol{\varphi}^*, \boldsymbol{y}(\boldsymbol{e}))$) *and continuous variables* ($[\boldsymbol{v}^{SS*}, \boldsymbol{\varphi}^*]$) *of the optimization problem in Eq. 6.10.B for some value of* $\boldsymbol{e}$.

Steady-state multi-scenario subproblem (Lower layer) (6.11)

$$f_{SUB}^{SS}(\boldsymbol{e}) = \min_{\boldsymbol{v}^{SS},\boldsymbol{\varphi}} f_{OBJ}^{SS}(\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e}))$$

$$s.t.$$

$$[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$$

Assume that $f_{OBJ}^{TR}(\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}) \geq 0$ *for any* $[\boldsymbol{v}, \boldsymbol{y}(\boldsymbol{e})]$ *such that* $[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$ *and* $[\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})] \in S^{TR}$*. Then,* $f_{SUB}^{SS}(\boldsymbol{e})$ *is a lower bound for the optimization problem in Eq. 6.10.B evaluated at* $\boldsymbol{e}$*, that is:*

$f_{SUB}^{SS}(\boldsymbol{e}) \leq f_{OBJ}\big(\boldsymbol{v}, \boldsymbol{y}(\boldsymbol{e})\big), \forall [\boldsymbol{v}, \boldsymbol{y}(\boldsymbol{e})]$ *such that* $[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$ *and* $[\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})] \in S^{TR}$

Theorem 6.1 provides information to calculate a lower bound for the dynamic *multi-scenario* subproblem in Eq. (6.10.B) through the solution of the steady state *multi-scenario* subproblem in Eq. (6.11). Note that the steady state and dynamic *multi-scenario* problems share the master problem in Eq. (6.10.A), i.e., the combination of Eqs. (6.10.A) and (6.10.B) results in the dynamic *multi-scenario* problem stated in Eq. (6.8), while the combination of Eqs. (6.10.A) and (6.11) results in the steady state *multi-scenario* problem defined in Eq. (6.9). Consequently, both optimization problems can be merged into a single optimization procedure using the master problem in Eq. (6.10.A), where steady state information from Eq. (6.11) can used to generate lower bounds for the dynamic subproblems in Eq. (6.10.B). When this theory is applied to the D-SDA, an additional pruning procedure like those applied by branch and bound techniques can be implemented. In particular, the new pruning rule allows to check the lower bound of neighboring candidate solutions and compare them with the current feasible solution of the dynamic *multi-scenario* problem. If a neighbor cannot produce a better solution than the current feasible solution, then this neighbor is discarded, i.e., the NLP dynamic subproblem is not solved. This is advantageous because the subproblem in Eq. (6.11) is much easier to solve than the dynamic subproblem in Eq. (6.10.B) in terms of computational time, e.g., the solution of the NLP problem in Eq. (6.11) may take a few seconds, while the solution of the NLP in Eq. (6.10.B) may be computationally demanding. A figure that illustrates this pruning procedure is shown in Figure 6.2. In this example, the objective function f_{SUB} is calculated at $e_1 = 4$. To decide the optimization direction, f_{OBJ}^{SS} is calculated for the neighbors $e_1 = 3$ and $e_1 = 5$. In this case, $f_{SUB}^{SS}(5) \geq f_{SUB}(4)$, therefore, the neighbor $e_1 = 5$ can be pruned since it implies that the dynamic decisions, and therefore the transient costs will increase the overall costs for this process. On the contrary, $f_{SUB}^{SS}(3) < f_{SUB}(4)$, thus, this

neighbor cannot be discarded and the dynamic subproblem must be solved to obtain $f_{SUB}(3)$ and continue the optimization process. Based on these developments, a novel modular solution procedure that incorporates an enhanced D-SDA is discussed next.

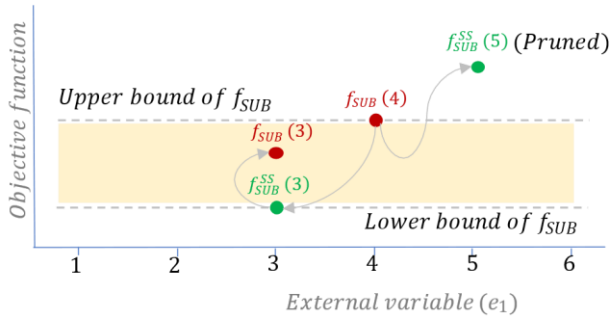


Figure 6.2. Illustration of the pruning procedure.

As shown in Figure 6.3, the proposed solution procedure consists of a series of modules solved sequentially, where each module builds on an initialization provided by its predecessor. The first 2 modules serve as initialization for Module 3, which is the rigorous simultaneous solution of the dynamic *multi-scenario* problem (Eq. (6.8)). The objective of Module 1 and Module 2 is to provide a feasible and near-optimal solution for Module 3. The idea of using this modular approach is to reduce computational costs, to acknowledge that steady state information can be used to initialize the dynamic *multi-scenario* problem, and to guarantee feasible initializations for the D-SDA and enhanced D-SDA algorithms illustrated in Figure 6.3. Another advantage of the proposed modular procedure is to make both steady-state and dynamic tools available for optimal process design. That is, if the user is only interested in a robust methodology for steady state design while considering every product simultaneously, Modules 1 and 2 are used to find an optimal steady-state design. In this situation, the D-SDA would still be advantageous compared to the previously used stochastic optimization strategies (see [325]), given that the D-SDA can find a proven locally optimal solution without the need of tunning parameters that may depend on the case study. Also, the D-SDA does not require to run the problem multiple times since it is posed as a deterministic optimization method. Accordingly, the D-SDA offers a reliable and attractive alternative to handle a problem that had been previously solved using stochastic optimization methods. Furthermore, Module 3 can be considered to obtain optimal dynamic information such as optimal control actions and their corresponding optimal transition times, and to acknowledge that the process' design and its steady-state operating points can be further optimized if process dynamics are considered in the analysis, at the expense of longer computational times.

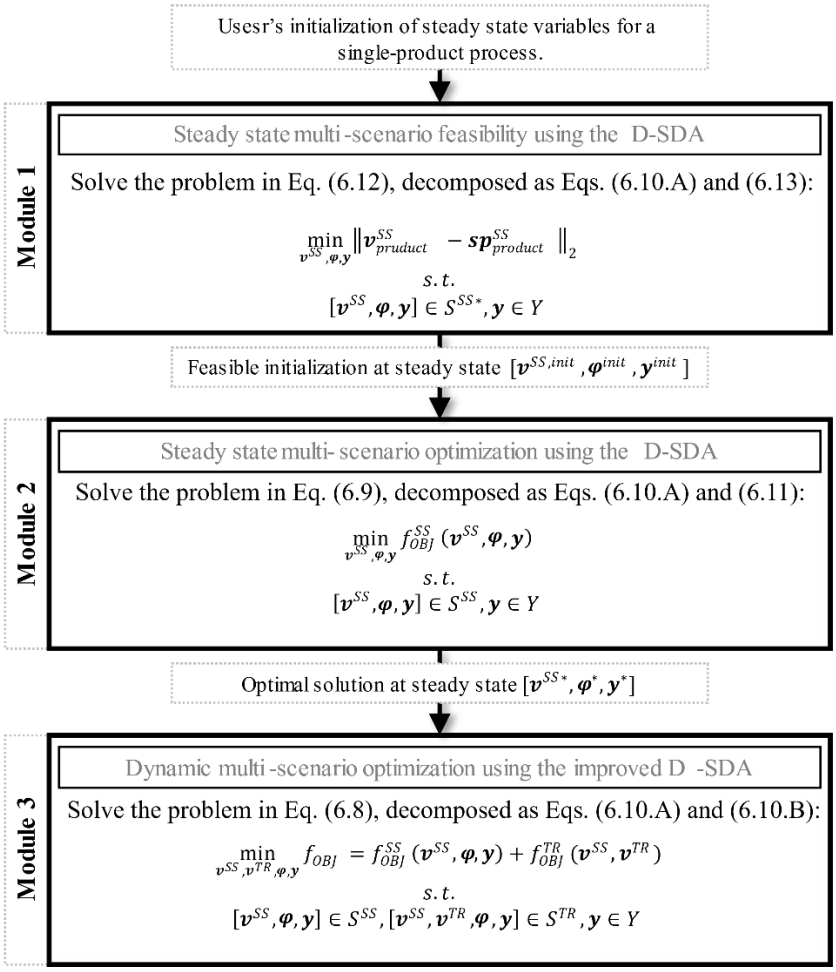


Figure 6.3. Proposed methodology for the solution of the simultaneous design and dynamic transitions problem. This results in the optimal transient solution $[\boldsymbol{v^{SSo}}, \boldsymbol{v^{TRo}}, \boldsymbol{\varphi^o}, \boldsymbol{y^o}]$.

As shown in Figure 6.3, an initialization for the variables that appear in the design of a typical CD column is provided first. For instance, this initialization may result from a simple steady state simulation of the process with a single product. This initialization allows Module 1 to find a feasible solution for the steady state *multi-scenario* problem in Eq. (6.9). The D-SDA can be used to find a feasible design from Module 1, by solving the feasibility optimization problem in Eq. (6.12). As introduced in section 6.2.2, there is a time dependent variable $v_{product}(t)$ that defines the composition/specification of the product/s of interest over time, and it must follow the desired setpoint $sp_{product}(t)$. The values of $v_{product}(t)$ and $sp_{product}(t)$ at steady state can be organized in vectors $\boldsymbol{v}^{SS}_{pruduct}$ and $\boldsymbol{sp}^{SS}_{product}$ (see Eq. (6.12)). The objective of problem (6.12) is to minimize the error

between the steady state values of the product grade and its desired composition, until a design that guarantees the desired compositions at steady state is obtained. Thus, S^{SS*} contains steady state constraints in S^{SS}, except for those that define the product specification of $v_{product}(t)$ with respect to its setpoint $sp_{product}(t)$ at steady state (that is, product specification constraints were moved to the objective function). The decomposed version of the problem in Eq. (6.12) that allows the solution of this problem using the D-SDA can be obtained by using the master problem in Eq. (6.10.A) and the subproblem in Eq. (6.13).

Steady-state multi-scenario feasibility problem

$$\min_{\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}} \left\| \boldsymbol{v}^{SS}_{pruduct} - \boldsymbol{sp}^{SS}_{product} \right\|_2 \tag{6.12}$$

$s.t.$

$[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}] \in S^{SS*}, \boldsymbol{y} \in Y$

$\boldsymbol{v}^{SS}_{pruduct} = \left[v_{product}(t_1), v_{product}(t_2), \dots, v_{product}\left(t_{n_K}\right)\right]$

$\boldsymbol{sp}^{SS}_{product} = \left[sp_{product}(t_1), sp_{product}(t_2), \dots, sp_{product}\left(t_{n_K}\right)\right]$

Steady-state multi-scenario feasibility subproblem (Lower layer) (6.13)

$$f^{FE}_{SUB}(\boldsymbol{e}) = \min_{\boldsymbol{v}^{SS},\boldsymbol{\varphi}} \left\| \boldsymbol{v}^{SS}_{pruduct} - \boldsymbol{sp}^{SS}_{product} \right\|_2$$

$s.t.$

$[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})] \in S^{SS*}$

The initialization step (Module 1) is completed once the objective function in Eq. (6.12) is equal to 0. If a solution is found but condition $\left\| \boldsymbol{v}^{SS}_{pruduct} - \boldsymbol{sp}^{SS}_{product} \right\|_2 = 0$ cannot be satisfied after multiple initialization trials, then the user-defined CD column model cannot satisfy the required product specification. In this case, the designer should check problem's constraints to verify if it is possible to consider other design alternatives to attain the required product specification, e.g., consider a double feed arrangement instead of a single feed, or increase the maximum allowed number of reactive stages. If modifications to the CD column design are not possible, then other process intensification strategies should be considered. If optimality is attained, Module 1 returns a design that satisfies the desired product specifications at steady state and serves as a feasible initialization for Module 2. Including Module 2 was motivated by the results found by Medina-Herrera et al., who solved an optimization problem of this type using a genetic algorithm [325]. That study concluded that the steady state production

periods dominate the process economics, when transitions are scaled-up to real scheduling conditions [28,325]. Consequently, the design obtained from Module 2 is expected to find a design with improved economic characteristics, without incurring in computationally taxing dynamic optimization problems. In contrast to previous works, I demonstrate in this work that it is possible to solve an optimization problem of this type using the deterministic (gradient-based) D-SDA. The rigorous dynamic *multi-scenario* optimization is performed in Module 3, which is solved using the enhanced D-SDA presented in Algorithm 6.1. Note that if the designer is not interested in considering process dynamics at the design stage, the process design can be terminated after Module 2. If that is the case (i.e., process dynamics are not considered in the analysis), this modular approach can be used to solve the steady state multi-scenario problem to optimality, thus making this the first approach in the literature to guarantee local optimality both at steady-state and under dynamic operation.

In the first step of the enhanced D-SDA (S1) the optimal solution at steady state $[\boldsymbol{v}^{SS*}, \boldsymbol{\varphi}^*, \boldsymbol{e}^*]$ is used to initialize the optimization procedure, by setting $\boldsymbol{e} \leftarrow \boldsymbol{e}^*$. In S1, it is assumed that it is possible to find a feasible solution for the dynamic *multi-scenario* problem, when discrete variables remain fixed at their optimal steady state value, i.e., $f_{SUB}(\boldsymbol{e}^*)$ is assumed to be finite. That is, this algorithm considers that there exists at least one design with reachable[k] steady state operating points for the discrete configuration defined by $\boldsymbol{e}^*$. A design that satisfies this characteristic will be denoted to as a "reachable design". Note that having successfully solved Modules 1 and 2 guarantees steady state operability, which is a necessary condition for reachability [315,336]. The iterative procedure starts with step S2, where the neighborhood of $\boldsymbol{e}$ is defined as $N_\infty(\boldsymbol{e})$, which contains those discrete solutions within a ∞-distance of $\boldsymbol{e}$. Similarly, $D_\infty(\boldsymbol{e})$ contains possible optimization directions to continue the search. Depending on the problem characteristics, a smaller neighborhood can be used to speed up the algorithm, at the expense of the possibility of obtaining non-optimal solutions [4,7,122]. Thereafter, the next step consists of pruning multiple neighbors in $N_\infty(\boldsymbol{e})$ using a pruning procedure (S3). To do so, the steady state *multi-scenario* subproblem (Eq. (6.11)) is solved for those points within a neighborhood of $\boldsymbol{e}$, initialized from the steady state information of $\boldsymbol{e}$, i.e., $[\boldsymbol{v}^{SS}(\boldsymbol{e}), \boldsymbol{\varphi}(\boldsymbol{e})]$. Then, the current optimal solution $f_{SUB}(\boldsymbol{e})$ is compared with those steady state solutions within the neighborhood to obtain recalculated sets of neighbors $N'_\infty(\boldsymbol{e})$ and directions $D'_\infty(\boldsymbol{e})$, based on the pruning methodology depicted in Figure 6.2.

[k] Reachable means that there exists at least one input vector $\boldsymbol{u}(t)$ that allows to attain the desired steady state operating points, from their corresponding initial conditions. In other words, reachable means that there exists at least one feasible solution for the dynamic optimization problem under consideration.

The recalculate neighborhood excludes those neighbors $\boldsymbol{\gamma}, \forall \boldsymbol{\gamma} \in \{\boldsymbol{\gamma} \in E: \|\boldsymbol{\gamma} - \boldsymbol{e}\|_\infty \leq 1\}$ for which the optimal solution of the steady state *multi-scenario* problem $f_{SUB}^{SS}(\boldsymbol{\gamma})$ is greater than the current dynamic *multi-scenario* solution $f_{SUB}(\boldsymbol{e})$. Thus, the number of dynamic NLP subproblems to be solved in S4 may be reduced. The main stopping criterion of the problem is verified in step S4, where the rigorous objective function f_{SUB} is calculated for the remaining neighbors after the pruning procedure. If the current solution $f_{SUB}(\boldsymbol{e})$ is optimal with respect to $f_{SUB}(\boldsymbol{\gamma}), \forall \boldsymbol{\gamma} \in N'_\infty(\boldsymbol{e})$, the solution is declared as a local optimum and the algorithm stops. Otherwise, the steepest descent direction $\boldsymbol{\delta}$ is selected in step S5, and the value of the external variables is updated as $\boldsymbol{e} \leftarrow \boldsymbol{e} + \boldsymbol{\delta}$. The algorithm then proceeds to the line search step, where it repeats the "pruning and stopping criterion" procedure, except that the current solution is now compared with one neighbor at a time, i.e., $f_{SUB}^{SS}(\boldsymbol{e} + \boldsymbol{\delta})$ and $f_{SUB}(\boldsymbol{e} + \boldsymbol{\delta})$, along the direction of steepest descent. Note that $f_{SUB}(\boldsymbol{e} + \boldsymbol{\delta})$ is calculated in step S7, only if this candidate was not pruned based on $f_{SUB}(\boldsymbol{e} + \boldsymbol{\delta})$ in step S6. The line search procedure over the direction of steepest descent continues and the *external variables* are recalculated as $\boldsymbol{e} \leftarrow \boldsymbol{e} + \boldsymbol{\delta}$ until no improvement is detected. This takes the algorithm back to step S2, where a new iteration begins. An illustrative example of the enhanced D-SDA is shown in Figure 6.4.

A key feature of the D-SDA and the enhanced D-SDA is that both incorporate initializations subroutines that aim to store previous values of continuous variables to be used as initializations in further steps. These routines allow NLP algorithms to handle highly nonlinear and large-scale optimization problems, for which good feasible initializations are critical for convergence. Apart from the initialization subroutine incorporated in Algorithm 6.1, section F.2 of appendix F presents a strategy to initialize dynamic optimization subproblems from steady state solutions. As in previous works, better performance to solve large scale and non-linear CD design problems is attained when using feasible-path-based solvers such as CONOPT, CONOPT4 [11], and MSNLP (with either CONOPT or CONOPT4 as NLP solver). Thus, these reinitialization subroutines are especially suitable for these solvers. In general, local optimality is guaranteed if the $N'_\infty(\boldsymbol{e})$ neighborhoods are used, and a global NLP solver is used to solve the subproblems. Despite the benefits in using global solvers, it is recommended to use local NLP solvers to obtain improved solutions in shorter computational times when dealing with complex highly nonlinear systems involving multiple external variables, such as those present in CD columns.

Algorithm 6.1. The improved D-SDA in Module 3 of Figure 6.3

Step 4. Initialization

Use the optimal solution at steady state $[\boldsymbol{v}^{SS*}, \boldsymbol{\varphi}^*, \boldsymbol{e}^*]$ to initialize problem's variables:

- $\boldsymbol{e} \leftarrow \boldsymbol{e}^*$
- $[\boldsymbol{v}^{SS*}(\boldsymbol{e}), \boldsymbol{\varphi}^*(\boldsymbol{e})] \leftarrow [\boldsymbol{v}^{SS*}, \boldsymbol{\varphi}^*]$
- Calculate the first dynamic *multi-scenario* subproblem $f_{SUB}(\boldsymbol{e}) = f_{OBJ}(\boldsymbol{v}^{SSo}(\boldsymbol{e}), \boldsymbol{v}^{TRo}(\boldsymbol{e}), \boldsymbol{\varphi}^o(\boldsymbol{e}), \boldsymbol{e})$ initialized from $[\boldsymbol{v}^{SS*}(\boldsymbol{e}), \boldsymbol{\varphi}^*(\boldsymbol{e})]$[a,b].
- Then, go to S2.

Step 5. Define the neighborhood

- Create sets $N_\infty(\boldsymbol{e}) = \{\boldsymbol{\gamma} \in E: \|\boldsymbol{\gamma} - \boldsymbol{e}\|_\infty \leq 1\}$ and $D_\infty(\boldsymbol{e}) = \{\boldsymbol{d}: \boldsymbol{\gamma} - \boldsymbol{e} = \boldsymbol{d}, \forall \boldsymbol{\gamma} \in N_\infty(\boldsymbol{e})\}$. Then, go to S3.

Step 6. Steady state neighborhood verification and pruning

- Solve $f_{SUB}^{SS}(\boldsymbol{\gamma}) = f_{OBJ}^{SS}(\boldsymbol{v}^{SS*}(\boldsymbol{\gamma}), \boldsymbol{\varphi}^*(\boldsymbol{\gamma}), \boldsymbol{\gamma}), \forall \boldsymbol{\gamma} \in N_\infty(\boldsymbol{e})$ initialized from $[\boldsymbol{v}^{SS*}(\boldsymbol{e}), \boldsymbol{\varphi}^*(\boldsymbol{e})]$.
- Recalculate the neighborhood and directions as: $N'_\infty(\boldsymbol{e}) = \{\boldsymbol{\gamma} \in E: \|\boldsymbol{\gamma} - \boldsymbol{e}\|_\infty \leq 1, f_{SUB}^{SS}(\boldsymbol{\gamma}) < f_{SUB}(\boldsymbol{e})\}$ and $D'_\infty(\boldsymbol{e}) = \{\boldsymbol{d}: \boldsymbol{\gamma} - \boldsymbol{e} = \boldsymbol{d}, \forall \boldsymbol{\gamma} \in N_\infty(\boldsymbol{e})\}$.
- From the solution obtained ($f_{SUB}^{SS}(\boldsymbol{\gamma})$), store $[\boldsymbol{v}^{SS*}(\boldsymbol{\gamma}), \boldsymbol{\varphi}^*(\boldsymbol{\gamma})], \forall \boldsymbol{\gamma} \in N'_\infty(\boldsymbol{e})$[c]. Then, go to S4.

Step 7. Stopping criterion

- Solve $f_{SUB}(\boldsymbol{\gamma}) = f_{OBJ}(\boldsymbol{v}^{SSo}(\boldsymbol{\gamma}), \boldsymbol{v}^{TRo}(\boldsymbol{\gamma}), \boldsymbol{\varphi}^o(\boldsymbol{\gamma}), \boldsymbol{\gamma})$ initialized from $[\boldsymbol{v}^{SS*}(\boldsymbol{\gamma}), \boldsymbol{\varphi}^*(\boldsymbol{\gamma})]$, $\forall \boldsymbol{\gamma} \in N'_\infty(\boldsymbol{e})$[a].
- If $f_{SUB}(\boldsymbol{e}) \leq f_{SUB}(\boldsymbol{\gamma}), \forall \boldsymbol{\gamma} \in N'_\infty(\boldsymbol{e})$, stop: $\boldsymbol{e} = \boldsymbol{e}^o$ represents the locally optimal value of external variables, with continuous variables $[\boldsymbol{v}^{SSo}(\boldsymbol{e}^o), \boldsymbol{v}^{TRo}(\boldsymbol{e}^o), \boldsymbol{\varphi}^o(\boldsymbol{e}^o)]$. Else, continue with S5.

Step 8. Selection of steepest descent direction

- Select a vector $\boldsymbol{\delta}$ from $D'_\infty(\boldsymbol{e})$ such that $f_{SUB}(\boldsymbol{e} + \boldsymbol{\delta}) \leq f_{SUB}(\boldsymbol{\gamma}), \forall \boldsymbol{\gamma} \in N'_\infty(\boldsymbol{e})$. Then, update *external variables* as $\boldsymbol{e} \leftarrow \boldsymbol{e} + \boldsymbol{\delta}$ and go to S6.

Step 9. Line search: pruning criterion

- Solve $f_{SUB}^{SS}(\boldsymbol{e} + \boldsymbol{\delta}) = f_{OBJ}^{SS}(\boldsymbol{v}^{SS*}(\boldsymbol{e} + \boldsymbol{\delta}), \boldsymbol{\varphi}^*(\boldsymbol{e} + \boldsymbol{\delta}), \boldsymbol{e} + \boldsymbol{\delta})$ initialized from $[\boldsymbol{v}^{SS*}(\boldsymbol{e}), \boldsymbol{\varphi}^*(\boldsymbol{e})]$. From the solution obtained, store $[\boldsymbol{v}^{SS*}(\boldsymbol{e} + \boldsymbol{\delta}), \boldsymbol{\varphi}^*(\boldsymbol{e} + \boldsymbol{\delta})]$.
- If $f_{SUB}^{SS}(\boldsymbol{e} + \boldsymbol{\delta}) \geq f_{SUB}(\boldsymbol{e})$, go to S2. Else, go to S7.

Step 10. Line search: stopping criterion

- Solve $f_{SUB}(\boldsymbol{e} + \boldsymbol{\delta}) = f_{OBJ}(\boldsymbol{v}^{SSo}(\boldsymbol{e} + \boldsymbol{\delta}), \boldsymbol{v}^{TRo}(\boldsymbol{e} + \boldsymbol{\delta}), \boldsymbol{\varphi}^o(\boldsymbol{e} + \boldsymbol{\delta}), \boldsymbol{e} + \boldsymbol{\delta})$ initialized from $[\boldsymbol{v}^{SS*}(\boldsymbol{e} + \boldsymbol{\delta}), \boldsymbol{\varphi}^*(\boldsymbol{e} + \boldsymbol{\delta})]$[a].
- If $f_{SUB}(\boldsymbol{e} + \boldsymbol{\delta}) \geq f_{SUB}(\boldsymbol{e})$, go to S2. Else, update *external variables* $\boldsymbol{e} \leftarrow \boldsymbol{e} + \boldsymbol{\delta}$ and go to S6.

a: I suggest following the guidelines in section F.2 of appendix F, where the suggested procedure required to initialize and solve the dynamic *multi-scenario sub*problems is discussed.

b: Notation $[\boldsymbol{v}^{SSo}(\boldsymbol{\gamma}), \boldsymbol{v}^{TRo}(\boldsymbol{\gamma}), \boldsymbol{\varphi}^o(\boldsymbol{\gamma})]$ indicates that these optimal variables were obtained for external variables $\boldsymbol{\gamma}$

c: Notation $[\boldsymbol{v}^{SS*}(\boldsymbol{\gamma}), \boldsymbol{\varphi}^*(\boldsymbol{\gamma})]$ indicates that these optimal variables at steady state were obtained for external variables $\boldsymbol{\gamma}$

Recall that binary variables can be directly calculated as a function of $\boldsymbol{e}$, e.g., $\boldsymbol{y}^* = \boldsymbol{y}(\boldsymbol{e}^*)$.

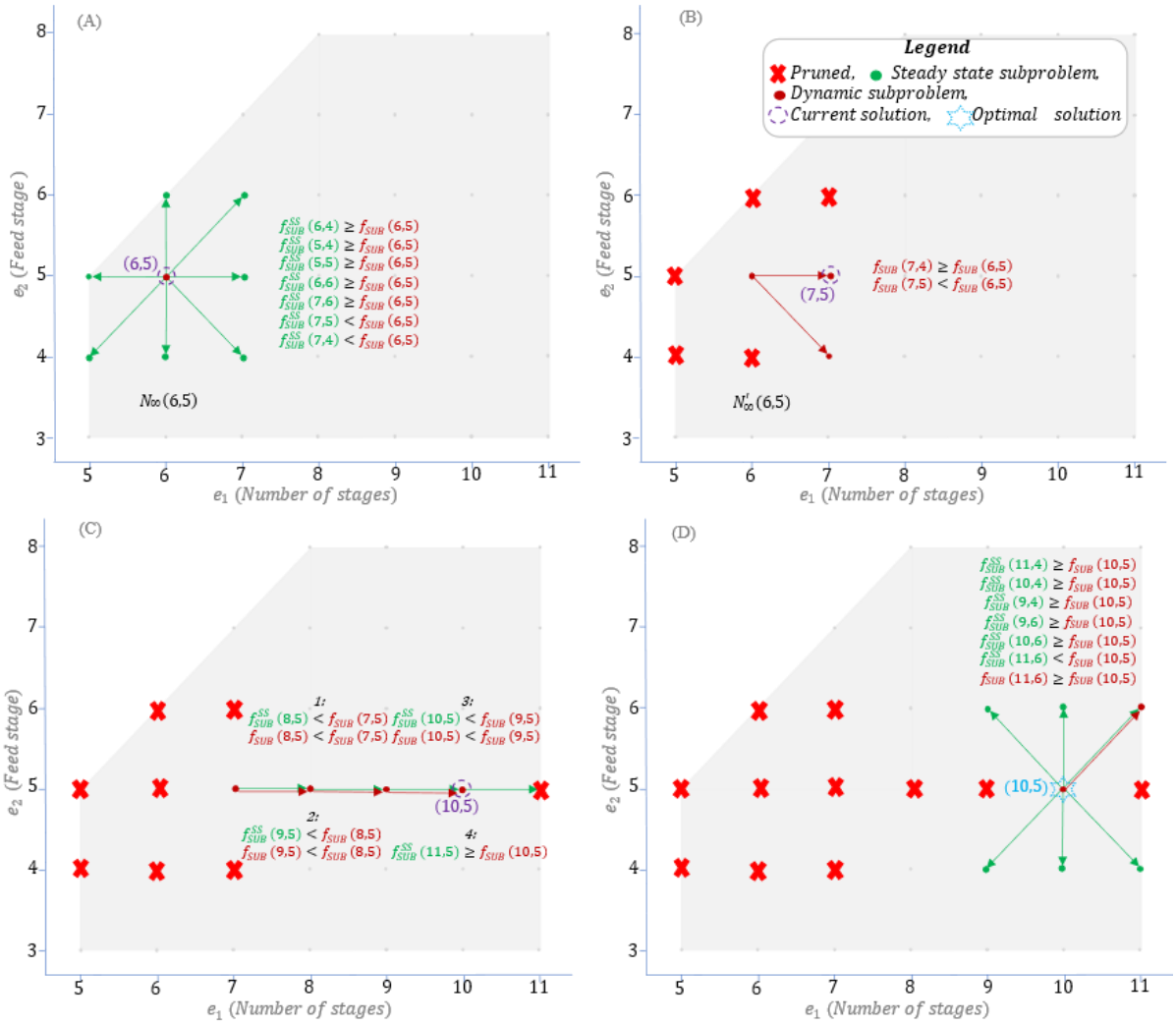


Figure 6.4. Illustrative representation of the enhanced D-SDA algorithm. A: Initialization (S1), definition of the initial neighborhood (S2) and steady state neighborhood verification (S3); B: pruning (S3), stopping criterion (S4) and selection of steepest descent direction (S5) C: line search (S6 and S7); and D: the next and last iteration starts at S2.

6.3 Case study

Currently, ASTM international standards suggest that ETBE must be produced at a nearly pure grade for blending with automotive and aviation fuels for spark-ignition engines [337,338]. Nonetheless, some studies in the literature have suggested the production of ETBE at non-standard compositions [339,340]. These works have investigated the potential benefits of producing ETBE at lower grades to use it in spark-ignition engines. These benefits include improving the spark-ignition engine's performance,

minimizing emissions of volatile organic compounds, and optimizing production costs. The ETBE grades available in the market and the alternative ETBE grades proposed in the literature suggest that the CD process can be further enhanced to attain the feasible production of ETBE at multiple grades. Thus, the methodology presented in the previous section was tested using a case study that consists of a CD column that produces ETBE at three different grades (63%, 83% and 95% [% mol/mol]). ETBE products can be obtained from the exothermic reaction of isobutene and ethanol, which occurs over an acid resin catalyst (see Figure 6.5).

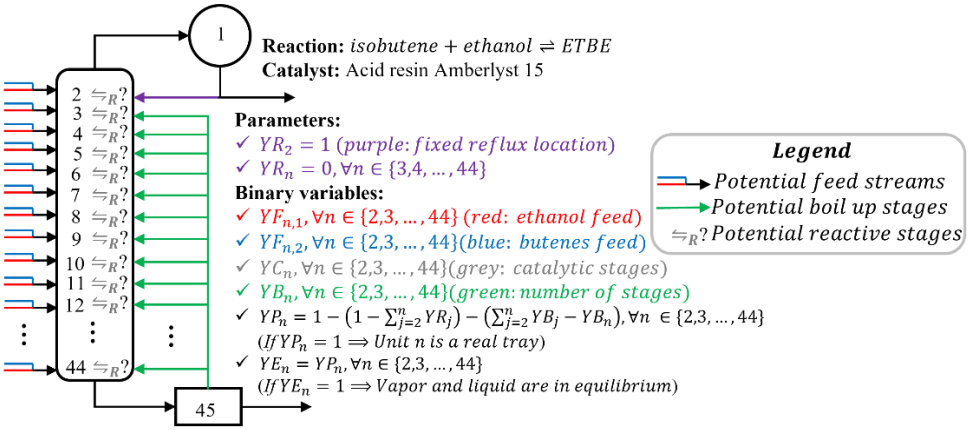


Figure 6.5. Schematic representation of the ETBE superstructure used for the case study.

In this case study, 3 catalytic stages are considered. These catalytic stages must be optimally arranged along the column, and these are allowed to follow a non-consecutive patter. A split feed model is considered, with a feed of pure ethanol ($IN_1 = 1.712\ mol/min$ at $342K$) and a second feed of butenes ($IN_2 = 5.774\ mol/min$ at $323K$) composed by 30% n-butene and 70% isobutene, where the n-butene is an inner component. These feed conditions were obtained from the experimental and numerical studies performed over a tray CD column by Miranda et al.[241] and Gómez et al. [138], respectively. Further details on the process operating conditions are provided in sections 4 and 5 in[241] and [138], respectively. Phase equilibrium was modeled using the gamma-phi formulation. Activity coefficients were calculated using the Non-Random Two-Liquid model to consider the nonideality of the liquid components in solution. For the vapor phase, fugacity coefficients for the components in solution were calculated using the Soave-Redlich-Kwong equation of state. The information required by these thermodynamic models was obtained from Aspen Plus. For the reaction, a pseudo-homogeneous Langmuir, Hinshelwood, Hougen and Watson kinetic model was implemented. The corresponding

kinetic parameters were taken from Jensen and Datta [341]. Similar assumptions were also considered in previous studies and deemed suitable for this system [7].

The discrete decisions of this problem are: the total number of stages, the optimal location of both feeds, and the optimal arrangement of catalytic stages. In a CD superstructure model, these discrete decisions are represented through binary variables, as shown in Figure 6.5. The independent binary variables $\boldsymbol{y}_I$ reformulated as *external variables* $\boldsymbol{e}$ are: i) variables $YF_{n,1}, \forall n \in \{2,3,\dots,44\}$, which are used to decide the location of the ethanol feed; ii) variables $YF_{n,2}, \forall n \in \{2,3,\dots,44\}$, which determine the location of the butenes feed; iii) variables $YC_n, \forall n \in \{2,3,\dots,44\}$ that represent the location of the three catalytic stages; iv) variables $YB_n, \forall n \in \{2,3,\dots,44\}$ that determine the optimum number of trays. In addition, there are dependent binary variables YP_n and $YE_n, \forall n \in \{2,3,\dots,44\}$, which indicate that only those stages within the reflux and the boil-up location will be part of the final design (i.e., real trays); hence, equilibrium relationships must be satisfied for those trays.

The independent continuous design variables (which are part of $\boldsymbol{\varphi}$) considered for this case study are: the column diameter, the height of the stages and the weir height. The remaining continuous design variables in $\boldsymbol{\varphi}$ are calculated from these independent variables. Regarding the manipulated variables, the feed streams remain fixed during the calculations; therefore, this CD column has two extra degrees of freedom when using a MESH model. In this study, these degrees of freedom (DOF) are the reboiler duty and the reflux ratio, which are usually selected as control variables for these systems after performing a Relative Gain Array (RGA) analysis [318]. Thus, the continuous degrees of freedom of this problem are the column diameter, the height of the stages, the weir height, the reboiler duty and reflux ratio at each sampling interval, and the size of the sampling periods. A detailed description of the problem's variables after performing the *external variables* reformulation is presented in Table 6.1 (the reader is referred to appendix F for more information). The objective function of this case study is described in section 6.3.1, followed by the description of the problem's constraints (section 6.3.2).

Table 6.1. Problem variables

Vector	Variables
Continuous variables ($\boldsymbol{\varphi}$)	Continuous design variables: - *Column diameter, stage height, weir height, weir length, total column height, active area, downcomer area, hole area.*
Differential variables ($\boldsymbol{\chi}(t)$)	Mole and energy accumulation at every stage: - *Mole accumulation per component per stage, total mole accumulation per stage, internal energy per stage.*
Algebraic variables ($\boldsymbol{\zeta}(t)$)	Variables that are forced to achieve the desired steady state $\boldsymbol{\zeta}_I(t)$: - *ETBE bottoms molar composition* (it is required to satisfy the product specification at t_k^{TR}). - *ETBE output flow* (This is the only algebraic variable that appear in the objective function; therefore, its expected steady state value should be properly estimated at t_k^{TR})
	Other algebraic variables: - *Vapor and liquid flows, vapor and liquid mole fractions, temperatures, pressures, compressibility coefficients, vapor and liquid densities, saturation pressures, surface tensions, activity coefficient of the liquid, reaction equilibrium constants, reaction rates, fugacity coefficients of the vapor, vapor and liquid enthalpies, vapor and liquid velocities, pressure drops, liquid and vapor accumulations, etc. (these variables appear at each stage).* - *Boil-up rate, distillate flow rate, condenser duty.*
External variables ($\boldsymbol{e}$)	Binary variables are reformulated with external variables (see Appendix F, section F.3), i.e., - *The total number of stages* - *The location of both feeds (ethanol and butenes)* - *The arrangement of 3 catalytic stages*
Time variables	- *Times where transition occurs* (see section 6.2.2) - *Size of sampling periods* (see section 6.2.2)

6.3.1 Objective function

This process is expected to operate $t_p = 480{,}000\, min/year$ (operation guarantees an allotment of 91% over the year). The remaining part of the year (9%) the operation is assumed to be shut-down for maintenance, e.g., yearly catalyst replacement due to its deactivation [342]. The following *scenarios* are considered in this case study: 1) transition from 63% ETBE to a steady state operation at 83% ETBE, 2) transition from 83% ETBE to a steady state operation at 95% ETBE, 3) transition from 95% ETBE

to a steady state operation at 83% ETBE, and 4) transition from 83% ETBE to a steady state operation at 63% ETBE. This implies that four scenarios are considered for the dynamic *multi-scenario* model. Nevertheless, the steady state at 83% ETBE resulting from transitions 1) and 3) are enforced to be the same, which simplifies the steady state *multi-scenario* model from four to three scenarios. Moreover, w is set at 1 to indicate that these dynamic transitions do not repeat cyclically over the year. Also, the system needs to produce each product (63%, 83% and 95%) over intervals of time of the same length over $[0, t_p]$, i.e., the production schedule must satisfy equal production times for each product. To guarantee this requirement, the time slots are constructed based on the following schedule: $t_0 = 0$, $t_1 = 80{,}000\ min/year$, $t_2 = 240{,}000\ min/year$, $t_3 = 320{,}000\ min/year$, $t_4 = t_p = 480{,}000\ min/year$.

The objective function used in this work is adapted from previous works [7]. This function is calculated using Eqs. (6.14.A)-(6.14.C) to obtain the annualized investment costs, time-dependent operating costs and time-dependent process incomes, respectively. In Eq. (6.14.A) $C_0 = 10000\ \$/year$ represents the fixed annualized investment cost of the column and the installation of the condenser and the reboiler; $AF = 0.23/year$ is an annualizing factor; D_c is the column diameter, and h_s is the height of the stage. In Eq. (6.14.B) $\bar{C}_1 = 0.015\ \$/mol$ and $\bar{C}_2 = 0.00825\ \$/mol$ are the costs of feeds, while $\bar{C}_r = 5.1 * 10^{-6}\ \$/kJ$ and $\bar{C}_c = 8.5 * 10^{-7}\ \$/kJ$ are the reboiler's and condenser's costs affected by the reboiler and the condenser duties ($Q_r(t)$ and $Q_c(t)$ in kJ/min), respectively. In Eq. (6.14.C), $L_{ETBE}(t)$ is the output molar flow of ETBE in mol/min and $\bar{C}_{ETBE}(t)$ is the selling price of ETBE, which was calculated assuming a linear relationship between the price and the ETBE composition according to Eq. (6.14.D).

$$C_{INV} = C_0 + AF\left[3.08\sum_{n\in INT} YC_n + 3.75\left(117.85D_c^{1.066}\left(1.15(\sum_{n=2}^{44} h_s YP_n)\right)^{0.802} + 4.7D_c^{1.55}\sum_{n=2}^{44} h_s YP_n\right)\right] \quad (6.14.A)$$

$$C_{OP}(t) = IN_1\bar{C}_1 + IN_2\bar{C}_2 + Q_r(t)\bar{C}_r + Q_c(t)\bar{C}_c \quad (6.14.B)$$

$$C_{IN}(t) = L_{ETBE}(t)\bar{C}_{ETBE}(t) \quad (6.14.C)$$

$$\bar{C}_{ETBE}(t) = \begin{cases} 0.025\ \$/mol\ , t = t_1\ (83\%) \\ 0.029\ \$/mol, t = t_2\ (95\%) \\ 0.025\ \$/mol\ , t = t_3\ (83\%) \\ 0.019\ \$/mol, t = t_4\ (63\%) \end{cases} \quad (6.14.D)$$

6.3.2 Problem's constraints

This case study is based on the NLP dynamic CD model presented by Bernal et al.[343], with enhancements and modifications required to express the steady state section of the model as an MINLP. Constraints that define the feasible regions Y and S are shown in Table 6.2 and Table 6.3, respectively.

Table 6.2. Constraints of the problem: feasible region Y

Type	Constraints
Constraints over discrete design variable	Logic constraints that define Y (these constraints are equivalently defined in the domain of external variables E in section F.3, appendix F): *The boil-up stage is below the reflux stage, there is only one boil-up stage, there is one feed stage per feed stream, feed stages are below the reflux stage, the boil-up stage is below the feed stages, the catalytic stages are in between feed stages; there is a fixed number of catalytic stages, the ethanol feed is above the butenes feed* (Eqs. (F.22)-(F.33) in Appendix F).

6.4 Results and discussion

Two sequential optimal design approaches were tested first for this case study and used to motivate the need of *multi-scenario* formulations. From the results, the best sequential methodology was compared with the simultaneous design and dynamic transitions of the ETBE column obtained from the methodology proposed in this work (section 6.4.2). The problems presented in this section were modeled and implemented in GAMS 34.3.0 and solved using a PC with 16 cores, 32 logical processors and 96 GB of RAM memory. The solution of every NLP subproblem was performed using CONOPT4, and these were fully discretized using the orthogonal collocation method, based on Lagrange interpolation polynomials [344].

Table 6.3. Constraints of the problem: feasible region *S*

Type	Constraints
Constraints over discrete and continuous variables ($\boldsymbol{g}^D$)	Constraints over design variables: - *Definition of: active area, downcomer area, weir length, hole area, and total column* heigh (Eqs. (F.34)-(F.37) in Appendix F)[a,b]. - *geometric inequality relationships between weir height and stage height, available catalyst space and length to diameter ratio* (Eqs. (F.38)-(F.41) in Appendix F)[a,b].
Constraints at steady state ($\boldsymbol{h}^{SS}$), i.e., valid at $t_k, \forall k \in K$	Balances at every stage (including condenser and reboiler) at steady state: - *Total mole balances, partial mole balances, summation equations, thermodynamic equilibrium relationships, and energy balances* (Eqs. (F.42)-(F.63) in Appendix F)[a].
	ETBE purity requirements (see Appendix F, Eq. (F.100)): - *Minimum desired ETBE composition at each scenario.*
	Tray capacity limits: - *Constraints required to avoid: entrainment flooding, downflow flooding, weeping, and catalyst packing flooding* (Eqs. (F.64)-(F.68) in Appendix F)[a,b].
Constraints during dynamic transitions ($\boldsymbol{h}^{TR}$), i.e., valid over $\left(t_{k-1}, t_k^{TR}\right], \forall k \in K$	Balances at every stage (including condenser and reboiler) at transient state: - *Total mole balances, partial mole balances, summation equations, thermodynamic equilibrium relationships, and energy balances* (Eqs. (F.69)-(F.90) in Appendix F)[b]
	Constraints over problem's inputs: - *Upper and lower bounds over the reboiler duty and the reflux ratio* (Appendix F, Eqs. (F.104) and (F.105)). - *Constraints required to simulate zero-order holds with variable length* (see Eqs. (6.6.A)-(6.6.C)).
Constraints that appear both in $\boldsymbol{h}^{TR}$ and $\boldsymbol{h}^{SS}$, i.e., valid over $\left(t_{k-1}, t_k^{TR}\right] \cup \{t_k\}, \forall \in K$	Intermediate expressions: - *Definition of: compressibility coefficients, vapor and liquid densities, saturation pressures, surface tensions, activity coefficient of the liquid, reaction equilibrium constants, reaction rates, fugacity coefficients, vapor and liquid enthalpies, vapor and liquid velocities, pressure drops, etc.* (Eqs. (F.91)-(F.99) in Appendix F)[a,b].
Constraints that integrate the different scenarios	Switchability constraints: - *Constraints required to stabilize the system at a new steady state* (Eqs. (6.7.A) and (6.7.B)).
	Continuity constraints and initial conditions: - *Constraints required to ensure continuity along* $[t_k^{TR}, t_k]$ (Eqs. (6.7.C) and (6.7.D)). - *Constraints required to calculate the initial state of the system* (Eqs. (6.7.E) and (6.7.F)), *and constraints required to ensure the same*

	steady state during ETBE production at 83% (Appendix F, Eqs. (F.102) and (F.103)).

a: the reader is referred to [343] for more details.

b: the reader is referred to [12]for more details.

An important advantage of D-SDA method is that it relies on the solution of multiple NLP subproblems that in some cases can be solved in parallel. Nevertheless, parallel computing techniques have not been tested for the D-SDA. In other cases, multiple threads can be assigned to a single optimization with CONOPT4 to improve the search efficiency, for problems with more than 100,000 variables or constraints. In this case, I used the 28 logical processors available to efficiently solve time consuming steps of the NLP dynamic subproblems using multithread and parallel computing strategies. This shortened by approximately a half the time required to solve dynamic NLP subproblems (see Appendix F, section F.2).

6.4.1 Sequential design methodologies: steady state design step

There is no general framework in the literature to sequentially design multitask CD columns at steady state. Therefore, two sequential approaches for the steady state design step of the multigrade CD process are presented in this section. Note that these approaches assume that the dynamic transitions of the process can be optimized in a subsequent step; thus, they do not consider process dynamics during the design stage. In addition, these approaches do not comprise every possible grade/product into a single optimization procedure; instead, they use heuristic procedures to find a design that guarantees multitask production. Hence, only the steady state constraints and objective function for a single scenario at a time are successively needed to obtain a design as follows: the first approach (*Seq 1*) consists of optimizing the design of different columns that produce each ETBE grade separately; next, an overdesigned column is proposed such that it uses the largest equipment sizes among those identified from the previous step. Such methodology was proposed in a previous study [323]. The second alternative (*Seq 2*) is to design a single column whose characteristics are expected to satisfy the requirements for every product, depending on the features of the case study under consideration. For instance, the production of polyester at two different grades was considered in an earlier study [310]. Although the reactants to produce both polyester grades differ, the reaction conditions for both products are similar, thus, the design found to produce one of the grades was assumed to satisfy the requirements for the second grade.

For the overdesign methodology (*Seq 1*), the ETBE column was optimized separately to achieve the desired specifications of 63%, 83% and 95% for each column. These optimizations were performed using the original D-SDA presented in chapter 3. The optimization required 2596.89 seconds, 948.60 seconds, and 3322.87 seconds for the 63%, 83% and 95% columns, respectively. The designs obtained are shown in Figure 6.6, which have optimal reboiler duties of 341.77 kJ/min, 363.16 kJ/min and 393.73 kJ/min, and optimal reflux ratios of 4.99, 4.37 and 4.40 for Figure 6.6A, Figure 6.6B and Figure 6.6C respectively. As expected, the optimal design for the 83% ETBE column is the same reported in[12], where a single product CD column to produce ETBE at 83% was optimally designed. A common characteristic of the three designs in Figure 6.6 is that the optimal arrangement of feeds and catalyst location follows a pattern where catalytic trays are accommodated alternately instead of the common consecutive arrangement, and the ethanol and butenes are always fed on reactive stages. These CD designs also allow to infer some trends when performing a sensitivity analysis over the desired ETBE composition, e.g., the economically optimal number of stripping stages increases when increasing the desired ETBE composition while the number of rectification stages decreases; less energy consumption is required by lower purity products when optimizing process economics; it is economically attractive to decrease the total number of stages to increase ETBE purity [7,345,346].

According to the overdesign methodology recommended in the literature (*Seq 1*), the highest number of reaction, rectification and stripping trays from the design alternatives in Figure 6.6A, Figure 6.6B and Figure 6.6C are selected to obtain the overdesigned configuration [323,325]. For example, the rectifying section in Figure 6.6A has 5 stages, and this is the largest number of rectification stages among the designs presented in Figure 6.6; hence, 5 rectifying stages are included in the overdesigned column. Also, the largest values for independent design variables (highlighted in bold in Figure 6.6) are specified for the overdesigned column, which is expected to satisfy every ETBE grade [323,325]. The resulting overdesigned column is presented in Figure 6.6D. The next step is the verification of the column obtained in Figure 6.6D, with respect to the constraints imposed over the individual designs in Figure 6.6. This step is required because the column obtained from *Seq 1* is simply the result of merging the designs in Figure 6.6, following the heuristic that an equipment with a certain amount of overdesign may satisfy the changes in the ETBE specification through time [347]. Among the geometric constraints for the individual designs (Figure 6.6), a maximum length to diameter ratio of 20 was stablished to limit the total column height [348]. In this case, it was found that the overdesigned column (Figure 6.6D) exceeds the maximum length to diameter ratio established for the individual designs in Figure 6.6 (see

Table 6.4); thus, the overdesigned column from *Seq 1* does not comply with the allowable foundation requirements for structural integrity and safety, e.g., it may not withstand high wind loads. This result illustrates that a sequential design strategy for CD columns may result in an infeasible design; hence, systematic and efficient strategies such as that presented in this work are indeed needed to avoid trial and verification procedures that may return infeasible or suboptimal CD designs.

Table 6.4. Length to diameter ratio

Design	Value	Upper bound
63% ETBE (Figure 6.6A)	20 (Feasible)	20
83% ETBE (Figure 6.6B)	20 (Feasible)	20
95% ETBE (Figure 6.6C)	20 (Feasible)	20
Overdesigned (Figure 6.6D)	25 (Infeasible)	20

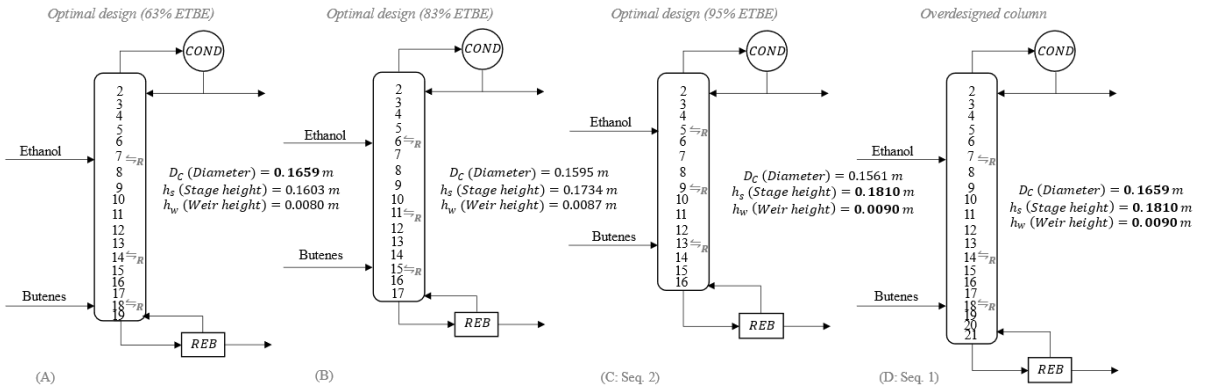


Figure 6.6. A,B,C: optimal design of ETBE columns at different compositions required for the sequential overdesign approach (*Seq. 1*). The design in C is the result from *Seq. 2*. The design in D is the result from *Seq. 1*.

The second approach adopted (*Seq 2*) requires the design of a column that is expected to satisfy the composition requirements for every ETBE grade. From previous studies, a column that can produce the highest ETBE grade is expected to satisfy the purity requirements for the other lower grades, because there is a monotonic behavior between reducing the output ETBE purity and simultaneously decreasing the reboiler duty and the reflux ratio in CD for ETBE production. This behavior was verified by manipulating and optimizing the operating variables of this unit (i.e., reboiler duty and reflux ratio) and it was found that the design in Figure 6.6C (*Seq 2*) can produce ETBE at every desired product

composition, i.e., the column that produces ETBE at 95% would result in a feasible design for the other ETBE grades. Nevertheless, this design has not yet considered neither dynamics nor the production of every ETBE grade simultaneously; thus, it is not expected to be optimal or even feasible in the transient domain. Therefore, this design will be compared in the following section with that obtained using the methodology proposed in Figure 6.3, which simultaneously optimize the steady state and dynamic economic performance of the unit for every ETBE grade.

6.4.2 Simultaneous design methodology

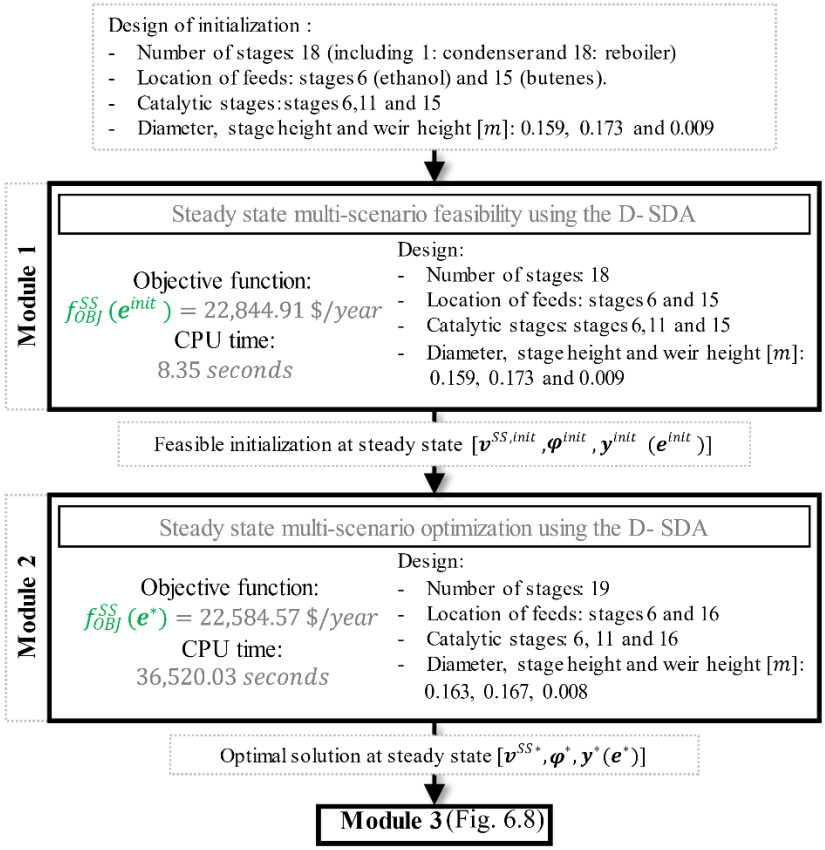


Figure 6.7. Results for the ETBE Case Study: Module 1 and Module 2. The initialization of Module 1 was taken from a previous work that considered the design of a ETBE column to produce ETBE at 83%.

The results of the simultaneous design and economic optimization of dynamic transitions for the ETBE column were found by following the modular strategy presented in Figure 6.3. The results of the first two modules are shown in Figure 6.7. These results are shown to emphasize that the first two

modules are able to generate a design that considers the changes in ETBE market demands at steady state. Also, optimality conditions guarantee that there will not be a better steady-state design in a vicinity of the solution found by the D-SDA. This optimality guarantee was not provided in previous works that considered every product simultaneously at steady state, for optimal RD process design [325]. The next step consists of using the steady state *multi-scenario* solution to initialize the MIDO problem in Module 3 presented in Figure 6.3. This problem was solved using the enhanced D-SDA presented in this work (Algorithm 6.1). The discrete decisions of the problems presented in Figure 6.5 can be represented with 172 independent binary variables, i.e., 43 binary variables in $YF_{n,1}, \forall n \in \{2,3,\dots,44\}$, 43 binary variables in $YF_{n,2}, \forall n \in \{2,3,\dots,44\}$, 43 binary variables in $YC_n, \forall n \in \{2,3,\dots,44\}$, and 43 binary variables in $YB_n, \forall n \in \{2,3,\dots,44\}$. These independent binary variables are reduced to 6 external variables only after applying the reformulation with *external variables* (see Appendix F, section F.3). The external variables represent the total number of stages, the location of two feeds, and the location of three catalytic stages.

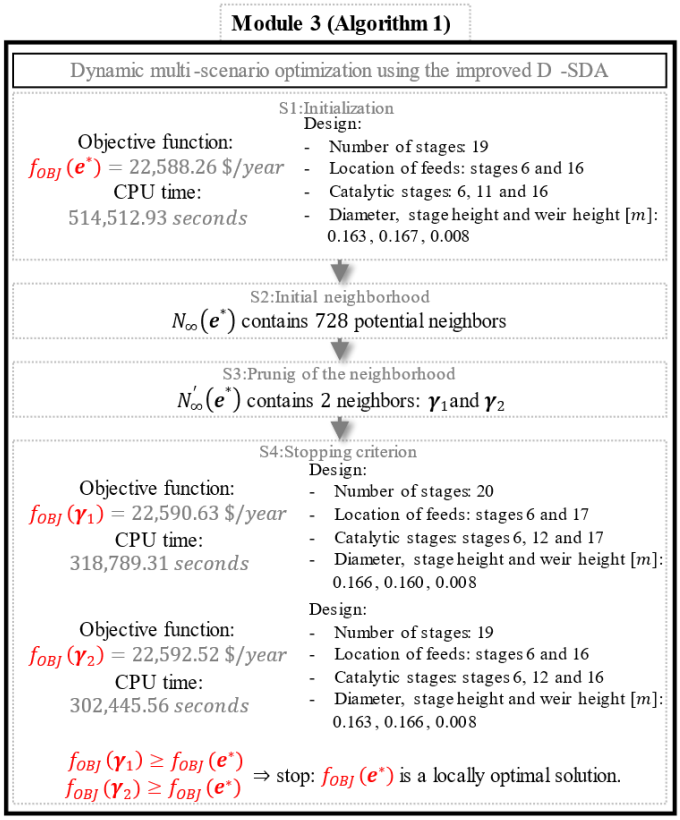


Figure 6.8. Results of module 3, which implements the enhanced D-SDA.

Module 3 returns additional information that cannot be obtained from Modules 1 and 2 alone. For instance, it provides the expected transition times for the product changeovers; the recommended optimal profiles for manipulated variables during transitions; the expected amounts of off-grade product generated during transitions, and a design that is economically optimal and dynamically feasible under transitions in product demands. The results obtained from Module 3 are shown in Figure 6.8. Note that design variables (number of stages, feed stages, diameter, etc.) did not changed when comparing the optimal solutions from Modules 2 and 3 in Figure 6.7 and Figure 6.8. This indicates that, specifically for this ETBE case study, the solution obtained from a steady state *multi-scenario* problem is also optimal with respect to a dynamic *multi-scenario* problem. This also suggests that the initialization step composed by Module 1 (8.35 seconds) and Module 2 (10 hours) is crucial to avoid lengthy and intensive computational calculations in Module 3 (13 days and 10 hours in total), i.e., Module 3 could have been initialized from other design that would have resulted in a total CPU time larger than 13 days and 10 hours. Note that this result does not indicate that Modules 1 and 2 are indeed sufficient to solve this optimization problem, because they do not guarantee neither dynamic feasibility nor optimality. This can only be achieved using the enhanced D-SDA in Module 3; that is, using Modules 1 and 2 alone does not guarantee that the design obtained will be optimal in the transient domain.

Table 6.5. Optimal design variables and costs: sequential and simultaneous approaches.

Optimization strategy	Sequential approach (*Seq 2*)	Simultaneous approach
Discrete configuration	Figure 6.9A	Figure 6.9B
Column diameter [m]	0.156	0.163
Stage height [m]	0.181	0.167
Weir height [m]	0.009	0.008
Annualized investment cost [$/year]	10,220.94	10,238.73
Annualized operating cost [$/year]	36,202.57	36,261.64
Annualized ETBE profit [$/year]	23,769.53	23,912.11
Objective function [$/year]	22,653.97	22,588.26

The optimal designs obtained are shown in Table 6.5, where the “sequential approach” corresponds to the methodology used to find the feasible design obtained from section 6.4.1 (*Seq 2*) while the “simultaneous approach” is that obtained using the approach proposed in this study, i.e., after running

Modules 1, 2 and 3. The discrete configuration of both designs is shown in Figure 6.9. The steady state operating variables and transition times (presented as $t_k^{TR} - t_{k-1}, \forall k \in K$) for each *scenario* and each approach are presented in Table 6.6. The resulting ETBE composition profile is presented in Figure 6.10. Also, the reboiler duty input profiles (Q_r), and reflux ratio input profiles are shown in Figure 6.11, for the simultaneous approach.

Table 6.6. Steady state operating variables (inputs) and transition times.

	Sequential approach (*Seq 2*)			Simultaneous approach		
Scenario	Reboiler duty [kJ/min]	Reflux ratio	Transition time [min] (ITAE[a])	Reboiler duty [kJ/min]	Reflux ratio	Transition time [min] (ITAE[a])
Scenario 1 (83% ETBE)	354.57	4.19	11.90 (210)	370.99	4.50	7.09 (58)
Scenario 2 (95% ETBE)	393.73	4.40	26.36 (542)	424.15	4.85	29.42 (598)
Scenario 3 (83% ETBE)	354.57	4.19	12.43 (241)	370.99	4.50	19.27 (184)
Scenario 4 (63% ETBE)	318.83	4.47	6.68 (46)	334.09	4.82	19.54 (117)

a: Integral of Time-weighted Absolute Error.

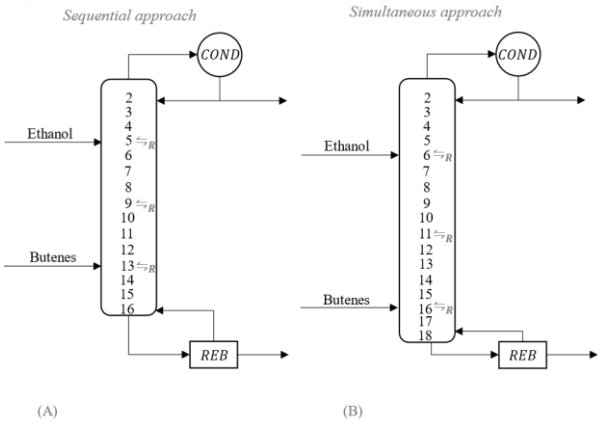


Figure 6.9. Discrete configuration of the design obtained with the sequential (A) and the simultaneous (B) approaches.

As shown in Table 6.4, the optimal solution obtained from the simultaneous approach ($f_{OBJ} = 22{,}588.26\ \$/year$) is very close to the optimal steady state objective function $f_{OBJ}^{SS} =$

$22{,}584.57\ \$/year$ in Module 2. This is because transition times for this system are in the order of magnitude of minutes (see Table 6.6), while the complete economic performance of the process was optimized with respect to one year of operation. Thus, the costs added by the dynamic transitions, which mainly results from the inclusion of transition periods where ETBE cannot be sold, are orders of magnitude lower when compared to the steady state operating costs, i.e., energy costs of the condenser and the reboiler duties. Consequently, the steady state operating conditions and the column design were not affected by the transient operation of the column. This is because the steady state production periods dominate the objective function since the system is mostly operating at steady state. Thus, it would be conflicting to change steady state operating conditions to improve the dynamic performance. These results confirm the findings in a previous study which suggested that even if transition times seem long, dynamic transition costs are irrelevant when incorporated into a realistic production schedule [28]. Additional case studies were investigated in appendix A to further confirm this observation.

When comparing the optimal solutions from the sequential approach (*Seq 2*) and the proposed methodology, the economic benefit of using the simultaneous approach with respect to the sequential approach was of 0.3%. This slight increase in profits can be explained as follows: 1) in contrast to sequential methodologies, the simultaneous approach always guarantees feasibility and local optimality at steady state, e.g., recall that the overdesign-based sequential approach (*Seq 1*) in section 6.4.1 resulted in an infeasible solution, without even considering the process transient operation; 2) local optimality under dynamic operation is ensured, as long as there exist a reachable design for the discrete configuration resulting from Module 2. Note that the assumption of reachability in sequential approaches is stronger (i.e., more difficult to satisfy) than in the simultaneous approach presented in this study. This is because all degrees of freedom involving continuous design variables and steady state operating conditions (i.e., diameter, stage height, weir height, and steady state values for reboiler duty and reflux ratio) remain fixed after completing the steady state design in a sequential procedure; hence, only the time-dependent manipulated variables remain available for control and set point tracking. In practice, design decisions and steady state operating conditions are affected by the transient operation of the process (i.e., process dynamics) and external perturbations entering the process, which may drive the system away from a nominal steady-state condition, e.g., the production of higher purity products is more difficult to control, hysteresis and instability may influence equipment requirements, the choice of reflux ratio and the reboiler design may affect controller pairing selections, etc. [346,349]. Therefore, the proposed simultaneous approach partially alleviates the reachability assumption by

explicitly accommodating the dynamic feasibility requirements of the process during the design stage. This is attained through the manipulation of continuous design variables and steady state operating variables at every iteration of the enhanced D-SDA (including its initialization step S1). Hence, the only feature that is attractive from the sequential approach is that it required less time than the simultaneous approach, i.e., the simultaneous approach (Modules 1, 2 and 3 in Figure 6.3) required approximately 13 days, while the sequential approach only required 3 days and a half. Accordingly, the steady-state *multi-scenario* design approach (Modules 1 and 2), and the dynamic *multi-scenario* design approach (Modules 1, 2 and 3) must be used to generate feasible and optimal designs of CD columns that generate multiple products or multiple grades of the single product. These problems are more challenging and CPU intensive but can be properly addressed using deterministic D-SDA optimization techniques in acceptable solution times for complex MIDO problems such as the present ETBE case study.

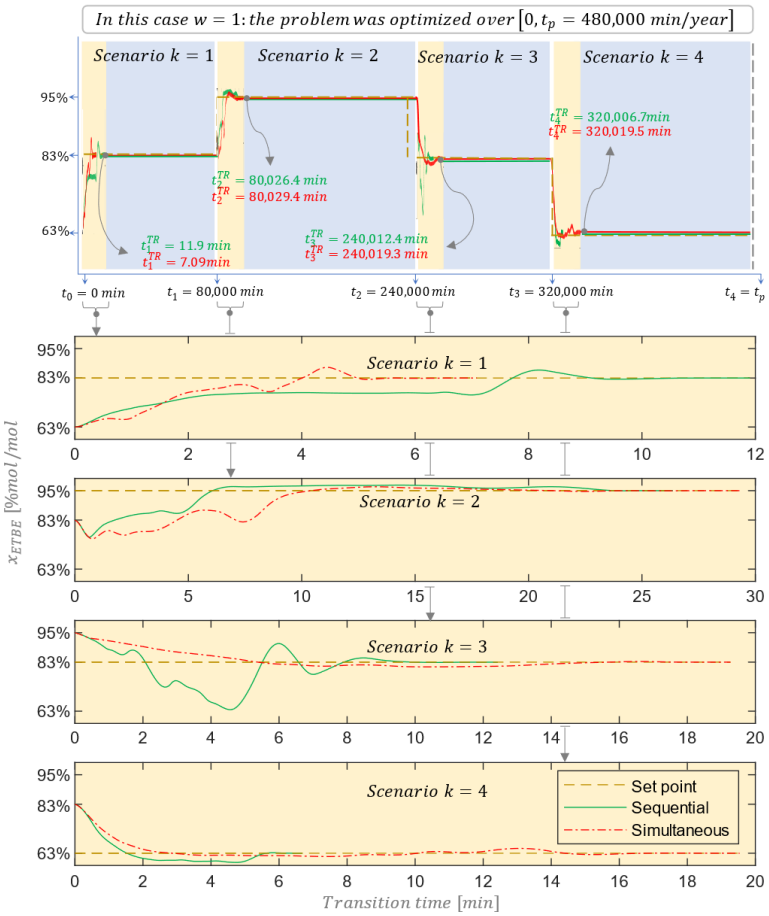


Figure 6.10. Composition of ETBE product during the production schedule.

The results of the ETBE composition profiles (Figure 6.10) show that both sequential and simultaneous approaches allowed the system to reach the desired steady states for the different transitions. For every case, the economic objective function decreased the transition times to achieve the desired product specification as fast as possible to avoid economic losses due to unspecified product (see Table 6.6). The simultaneous approach resulted in a better economic objective function, even though it required more time in the transitions stage than the sequential approach, i.e., $57\,min$ (sequential) vs $75\,min$ (simultaneous) in total. Since the sequential approach could not stablish an optimal design with optimal steady state operating conditions that consider every grade and every transition simultaneously at its first step (optimal design), the second step in this approach (optimization of the dynamic transitions) was expected to focus on minimizing off-grade production for economic performance. In contrast, the need to minimize transitions times to improve the total annualized cost was not a priority for the simultaneous approach since it can modify other variables with larger effects on the dynamic economic objective function, e.g., optimal steady state operating conditions that consider every ETBE grade simultaneously.

The simultaneous approach also shows additional advantages from a process control standpoint. The minimum Integral of Time-weighted Absolute Error (ITAE) is a recommended performance metric, due to its sensitivity when compared to other metrics [350,351]. Furthermore, it is a suitable metric for set-point changes, because its contribution of initial errors is less emphasized than late errors [352]. In this case, the ITAE was calculated using the relative error of the ETBE composition with respect to the desired setpoints ($error(t)$). When adding the ITAEs for the different *scenarios* to obtain an overall measure of control performance, the sequential approach resulted in an $ITAE$ of 1039, while a value of 957 was obtained for the simultaneous approach. On balance, the simultaneous approach resulted in slower transitions than the sequential approach, but with a smaller $ITAE$ value that indicates better set-point tracking performance.

$$ITAE = \int_0^\infty t|error(t)|dt \tag{6.15}$$

The control actions on the manipulated variables are presented in Figure 6.11 for the simultaneous approach. These control profiles are nontrivial since they cannot be obtained from a steady state analysis. The reboiler duty (Q_r) is the key input because it is directly related to the economic performance of the process. Thus, reboiler duty profiles must attain a trade-off between allowing low

energy costs and achieving the required product specifications. Moreover, the average steady state inputs for the simultaneous approach ($Q_r = 376$ and $reflux\ ratio = 4.7$) where larger than those obtained by the sequential approach ($Q_r = 356$ and $reflux\ ratio = 4.4$). Although this seems contradictory, the larger energy requirements by the simultaneous design are responsible of minimizing the total annualized cost of the system. This is because ETBE incomes are considered as part of the objective function. Hence, larger energy requirements are required to evaporate larger amounts of liquid recycled to the column, which guarantees a larger amount of ETBE produced by this CD system[12].

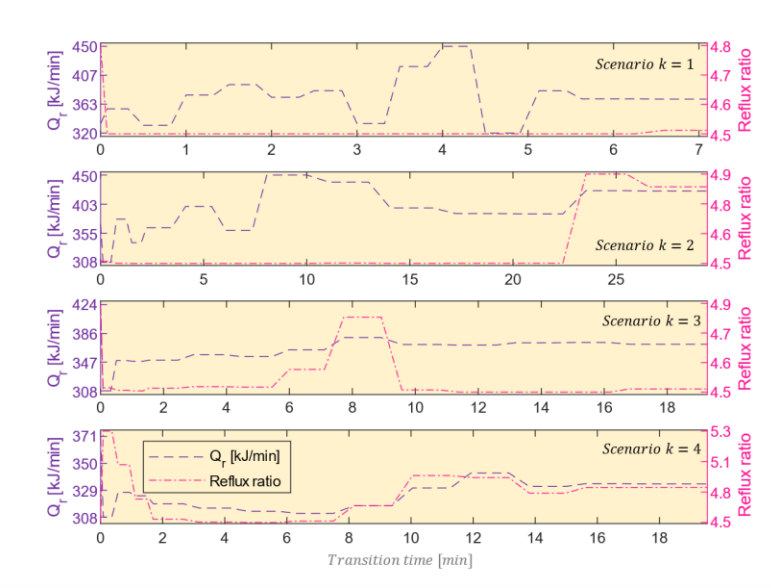


Figure 6.11. Inputs during dynamic transitions: simultaneous approach. Only the transition period of each scenario is plotted.

In this work, the enhanced D-SDA allowed to decrease the required number of discrete neighbors to be solved for the optimality verification. If this problem would have been solved using the conventional D-SDA, it would have been computationally intractable to solve the 728 subproblems that constitute a neighborhood for a problem with 6 *external variables*. This is because the conventional D-SDA would not have applied the pruning technique proposed in this work. Instead, it would have solved every dynamic *multi-scenario* subproblem within $N_\infty(\boldsymbol{e}^*)$, i.e., 728 subproblems. Since each of the dynamic NLP subproblems required on average 4.4 days, the solution of 728 subproblems would take approximately 9 years to complete. Even if $N_\infty(\boldsymbol{e}^*)$ is reduced to those neighbors that are feasible at steady state, the solution time would be almost 2 years, which is still computationally prohibitive.

Hence, the pruning strategy developed in this work for the enhanced D-SDA is instrumental to efficiently handle optimal CD design. Note that the dynamic section of the objective function (f_{OBJ}^{TR}) resulted to be greater than 0 for every dynamic NLP subproblem solved in the present case-study. This validates the assumption that dynamic transitions add costs to the problem due to a product out of specification that cannot be sold during transitions; thus, allowing the implementation of the pruning procedure.

In summary, it is possible to distinguish between the theoretical and practical contributions of this chapter. The first theoretical contribution is the development of a general modeling framework for the optimal economic design of multipurpose continuous CD systems, that allows to consider every product simultaneously in a multi-scenario modeling framework and to incorporate process dynamics into the design if needed. The second theoretical contribution is a pruning procedure that allows to incorporate dynamics into the optimal design of the unit. On the other hand, the practical contribution of this work is the development of a modular optimal design methodology that allows to generate near-optimal designs if the modular procedure is completed until Module 2, or to ultimately return an optimal design that is dynamically feasible if the complete procedure shown in Figure. 6.3 is performed. In particular, the results showed that, for the ETBE case study, an optimal design can be obtained using the D-SDA and the steady-state multi-scenario approach (Module 2). Moreover, our implementation of Module 3 provides guarantees that the design found from Module 2 is also an optimal dynamically feasible design in the presence of transitions between the different ETBE products considered in this study. The enhanced D-SDA also returns optimal control profiles for the manipulated variables, which are not trivial and cannot be obtained from steady-state calculations. Hence, the execution of Module 3 provides new insights on the expected dynamic behavior of the CD process under product transitions at the conceptual design stage. Note that the outcomes from our case study may differ when the optimal design and dynamic transitions of other PI applications is considered, e.g., when considering closed-loop design and control of the ETBE CD column, as discussed in appendix A.

Chapter 7

Discrete-Time Network Scheduling and Dynamic Optimization of Batch Processes with Variable Processing Times through Discrete-Steepest Descent Optimization

This chapter deals with the SSDO of a network batch process using a discrete-time scheduling formulation, with a novel feature which is the incorporation of variable processing times. The processing time of a batch (also referred to as cycle time) is the total time required to perform a batch operation, e.g., in a reaction task it may include the reaction time, and the time needed to charge and discharge the reactor[353]. The effect of variable processing times is crucial for the SSDO of batch plants, e.g., short processing times may improve scheduling economics by increasing throughput, while long processing times may reduce the consumption of heating and cooling resources during the dynamic operation. The importance of variable processing times has also been highlighted in previous works. For instance, Moreno-Benito et al.[354] studied an industrial-size batch copolymerization process to produce an acrylic fiber with a specific composition and quality. For this problem, an adequate selection of processing times was detrimental to generate a polymer that satisfies quality requirements, e.g., an adequate molecular weight distribution. The operational difficulties that may arise if processing times are not adequately selected in the batch copolymerization process include additional downstream costs, e.g., additional polymer-solvent separations that could have been avoided by selecting processing times that result in high conversions.

The way time variables are incorporated in the SSDO formulation highly depends on the time representation. Continuous-time formulations handle task-timing and processing time decisions as continuous variables, while discrete-time formulations usually handle them as discrete decisions after a discretization of the time domain. Although several works have addressed the SSDO of network batch processes with variable processing times using continuous-time scheduling formulations[60,61,73,355–358], the solution of this problem using a discrete-time scheduling formulation has not been widely studied. The studies reported in the literature involving discrete-time SSDO either assume fixed processing times[94,359], or consider a limited search space for processing times through *task modes*, i.e., a task is split into different operation modes, each with an assigned discrete processing time [360]. To the authors' knowledge, the study by Nie et al.[62] is the only work in the literature that has considered a discrete-time

network-based SSDO problem with variable processing times; however, variable processing times that may vary continuously are not allowed in their formulation and the infeasibilities that variable processing times may generate are not adequately handled by their solution framework. The case study considered in that work only decides between fast or slow operation modes of tasks, which are assumed to yield feasible schedules. This way of approximating variable processing times may result in suboptimal or infeasible operation of the plant if a wider range of processing times are needed to adequately model the operation, e.g., very short processing times may not be sufficient to meet the required conversion in a batch reactor, while lengthy processing times may lead to economic losses. Overall, discrete-time network scheduling formulations have been widely used to handle scheduling problems of industrial relevance, due to their versatility and scalability [40,361]. Nevertheless, the application of discrete-time formulations in network SSDO is limited, especially when considering variable processing times.

This chapter proposes a modeling strategy to address the solution of the top-down SSDO of batch processes with variable processing times, using a single-grid discrete-time network scheduling formulation. A modified version of the discrete-time state task network (STN) scheduling model presented by Shah et al.[362] is proposed in this work, which includes variable processing times and nonlinear process dynamics. The proposed formulation allows the user to explicitly define upper and lower bounds for processing times, which are used internally by the model to define the constraints needed to incorporate variable processing times within the formulation. In addition, the proposed model allows to approximately consider processing times as continuous variables that vary between fixed lower bounds and variable discrete upper bounds that coincide with the pre-specified time intervals in the formulation, i.e., the proposed model allows to select between a purely discrete-time formulation or a mixed-time representation [363]. Note that the challenges associated to the practical implementation of SSDO solutions for network batch plants with variable processing times are out of the scope of this work, e.g., processing times that are higher than expected during operation due to unforeseen disturbances, or model parameter uncertainty. The LD-SDA has the potential to handle the ordered discrete decisions that emerge in the discrete-time SSDO of batch network processes, which is illustrated in this chapter.

This chapter is organized as follows: section 7.1 summarizes the SSDO optimization problem under consideration. Section 7.2 presents the proposed mathematical framework to address the SSDO problem, i.e., problem formulation and solution strategy. Section 7.3 applies the proposed framework

to two case studies to test their performance and compare their solution with sequential scheduling and dynamic optimization solutions, and a general purpose MINLP solver. Concluding remarks and future work are provided in section 7.4.

7.1 Problem statement

The problem studied in this work involves the optimization of an objective function, e.g., cost minimization or profit maximization, of different batch processes interconnected within a process network. The problem incorporates scheduling decisions such as batching and assignment (i.e., determine the number of times a task is performed and the specific tasks assigned to each unit), sequencing and timing (i.e., order, starting time, and processing time required for the execution of tasks), batch sizes (i.e., determine the amount of material that should be processed when a task is executed), and inventories (i.e., amount of raw, intermediate or final products available at a given point in time). These scheduling decisions are linearly constrained using a discrete-time STN formulation. The STN representation is a graphical depiction of processing network environments in chemical process scheduling [360]. Network production environments generalize sequential production environments, in the sense that a job going through multiple operations is not defined. Instead, network scheduling models enable more complex features often found in chemical processes, such as tasks with multiple inputs and outputs where material splitting and mixing is allowed, e.g., mixing reactants in a reactor, or separating a mixture in a distillation column. Thus, network environments do not impose constraints on how input and output materials are handled; instead, a production network is defined where materials can flow from storage vessels to processing units and vice-versa, allowing multiple flow inputs and outputs for storage vessels and processing units[41]. Among the available models for STNs, discrete-time material-based approaches where states are explicitly modeled and their inventory is monitored over time have been extensively implemented in both academy and industry, being the model in [362] one of the most studied representations in the field. This modeling technique offers multiple advantages, e.g., it shows better performance than continuous-time models when handling large-scale STN instances [364]; it can be easily extended to consider common processing characteristics such continuous processes, periodic process or mixed batch-continuous processes [365], and it has been adapted to tackle challenging problems such as simultaneous scheduling and dynamic optimization under uncertainty[94]. Due to its versatility and success in addressing large-scale scheduling instances, a discrete-time STN model has been selected in this chapter.

Differential equations and algebraic constraints are used to model the transient behavior of tasks, thus adding manipulated, state, algebraic, and controlled variables to the formulation. The modeling of variable processing times is the key feature that differentiates the problem solved in this work when compared to previous network SSDO models. The discrete-time nature of the STN formulation requires additional binary/Boolean variables to model tasks with variable duration thus making the solution of those problems challenging. To address this issue, this work introduces a GDP approach to facilitate the modeling of variable processing times in discrete-time STNs, where disjunctive terms are incorporated to select the appropriate processing time of tasks. Variable processing times interact with the transient operation of the batch processes involved through linking constraints. For example, the processing time of a distillation task in the scheduling constraints must include its operating time (i.e., distillate collected divided by its flowrate), which appears in the differential equations used to model the distillation system. As a result, a tradeoff may exist between the time decisions in the scheduling time-scale (i.e., discrete-time scheduling grid) and the control time-scale (i.e., discretized time domains to represent differential equations). The proposed mathematical formulation considered in this work is described next.

7.2 Mathematical framework

The details of the discrete-time STN and the modifications proposed to model variable processing times are described first in section 7.2.1, followed by a mathematical formulation of the problem in section 7.2.2. The steps required to solve the resulting problem using the LD-SDA are explained afterwards, i.e., problem reformulation and decomposition (section 7.2.3) followed by an explanation of the LD-SDA steps to address the proposed SSDO problem (section 7.2.4).

7.2.1 Discrete-time STN with variable processing times for SSDO

The scheduling constraints of the SSDO formulation proposed in this work build upon the single-grid and discrete-time STN scheduling model considered by Velez and Maravelias[361], which is based on the formulation originally proposed by Kondili et al.[360] and Shah et al.[362]. The scheduling horizon of duration η is discretized using $n_T + 1$ time slots of length $\hat{\delta} = \eta/n_T$, resulting in discretization index $t \in T = \{0,1,\dots,n_T\}$. The STN consists of *tasks* denoted by index $i \in I$ (e.g., reaction and separation), *units* denoted by index $j \in J$ (e.g., batch reactor and distillation column), and *states* denoted by index $m \in M$ (e.g., input, intermediate, and final product storage). The production environment is defined in

terms of binary parameters $I^{-}_{i,m}/I^{+}_{i,m}, \forall i \in I, \forall m \in M$, which take the value of 1 when task i consumes/produces material from/to state m; and binary parameters $I^{\times}_{i,j}, \forall i \in I, \forall j \in J$, which indicate if task i can be performed at unit j. Parameter $I^{\times}_{i,j}$ is used to define set $IJ^{\times} = \{(i,j): I^{\times}_{i,j} = 1\}$, which represents a task-to-units mapping through tuples (i,j), i.e., if (i,j) belongs to $IJ^{\times}$, it means that task i is allowed to be performed in unit j. Based on this sets, the main scheduling variables are binary variable $x_{i,j,t}$, which takes the value of 1 when unit j processes task i starting at time t; integer batching variable $n_{i,j}$, which represents the number of times task i is processed in unit j; positive real variable $b_{i,j,t}$, which defines the batch size of task i when processed at unit j starting at time t, and positive real variable $s_{m,t}$, which tracks the inventory of state m at time t.

The traditional discrete-time STN formulation assumes discrete processing times as parameters ($\tau_{i,j}$) and represents the main limitation to incorporate variable processing times in the formulation. Parameter $\tau_{i,j}$ represents the number of points in the discrete-time grid required to perform task i in unit j. Fixed processing times ($\tau_{i,j}$) cannot be simply specified as variables in the traditional discrete-time STN formulation, because they appear as indices affecting both variables $x_{i,j,t}$ and $b_{i,j,t}$, e.g., $b_{i,j,t-\tau_{i,j}}$. To circumvent this issue, the following variables are introduced: 1) positive real variables $\tau^{P}_{i,j,t}$ (with units of time) to represent the processing time of task i in unit j when executed at time t; 2) Boolean variables $Y_{\tau_{i,j},i,j}$ to indicate if the execution of task i in unit j requires $\tau_{i,j}$ discrete time points, and 3) auxiliary continuous variables $x^{sum}_{i,j,t} = x^{sum}_{i,j,t}(x_{i,j,t-\tau_{i,j}+1}, x_{i,j,t-\tau_{i,j}+2}, \dots, x_{i,j,t})$ and $b^{shift}_{i,j,t} = b^{shift}_{i,j,t}\left(b_{i,j,t-\tau_{i,j}}\right)$, which aim to couple the newly added variables with the STN variables. A modified discrete-time STN model that includes these variables is shown in Eqs. 7.1A-7.1H. Note that I have selected a GDP formulation to represent the newly added variables and constraints, to avoid relying on any specific reformulation, e.g., big-M.

$$b^{min}_{j} x_{i,j,t} \leq b_{i,j,t} \leq b^{max}_{j} x_{i,j,t}, \ \ \forall (i,j) \in IJ^{\times}, \ \forall t \in T \tag{7.1A}$$

$$\sum_{\substack{i \in I \\ I^{\times}_{i,j}=1}} x^{sum}_{i,j,t} \leq 1, \ \ \forall j \in J, \forall t \in T \tag{7.1B}$$

$$s_{m,t} = s_m^{init} - \sum_{\substack{i \in I \\ I_{i,m}^- = 1}} \rho_{i,m}^- \sum_{\substack{j \in J \\ I_{i,j}^\times = 1}} b_{i,j,t}\,, \forall m \in M, t = 0 \tag{7.1C}$$

$$s_{m,t} = s_{m,t-1} + \sum_{\substack{i \in I \\ I_{i,m}^+ = 1}} \rho_{i,m}^+ \sum_{\substack{j \in J \\ I_{i,j}^\times = 1}} b_{i,j,t}^{shift} - \sum_{\substack{i \in I \\ I_{i,m}^- = 1}} \rho_{i,m}^- \sum_{\substack{j \in J \\ I_{i,j}^\times = 1}} b_{i,j,t}\,, \forall m \in M, \forall t \in T\backslash\{0\} \tag{7.1D}$$

$$s_{m,t} \leq s_m^{max}, \forall m \in M, \forall t \in T \tag{7.1E}$$

$$\sum_{t \in T} x_{i,j,t} = n_{i,j}, n_{i,j} \in N_{i,j}\,, \forall (i,j) \in IJ^\times \tag{7.1F}$$

$$\tau_{i,j}^{P,min} x_{i,j,t} \leq \tau_{i,j,t}^P,\ \forall (i,j) \in IJ^\times,\ \forall t \in T \tag{7.1G}$$

$$\underline{\tau}_{i,j} \in \mathrm{T}_{i,j} \left[\begin{array}{c} Y_{\tau_{i,j},i,j} \\ \tau_{i,j,t}^P \leqq \min\left\{\hat{\delta}\tau_{i,j}, \tau_{i,j}^{P,max}\right\}, \forall t \in T \\ x_{i,j,t}^{sum} = \sum_{\substack{t' \in T \\ t-\tau_{i,j}+1 \leq t' \leq t}} x_{i,j,t'}\,, \forall t \in T \\ b_{i,j,t}^{shift} = \begin{cases} b_{i,j,t-\tau_{i,j}}, & t - \tau_{i,j} \geq 0 \\ 0, & t - \tau_{i,j} < 0 \end{cases}, \forall t \in T\backslash\{0\} \end{array} \right], \forall (i,j) \in IJ^\times \tag{7.1H}$$

Eqs. 7.1A-7.1F define the constraints that are typically used in a discrete-time STN formulation, i.e., unit capacity constraints (Eq 7.1A), unit utilization constraints (Eq 7.1B), material balances (Eq 7.1C-7.1D), and maximum storage constraint (Eq 7.1E). These equations consider additional parameters to define the scheduling problem such as the maximum (b_j^{max}) and minimum (b_j^{min}) capacity of unit j; the maximum storage capacity of state m (s_m^{max}); the initial storage in state m (s_m^{init}), the fraction of material processed by task i that is coming from state m ($\rho_{i,m}^-$), and the fraction of material processed by task i that is dispatched to state m ($\rho_{i,m}^+$). Terms $\rho_{i,m}^-$ and $\rho_{i,m}^+$ are crucial in a STN formulation because they specify the fraction of materials flowing to/from the different states to/from the different tasks in the network. Tasks may have multiple input or output connections to/from different states;

hence, the sum of input fractions and the sum of output fractions from a task must be equal to 1 when declaring the STN, i.e., $\sum_{m\in M, I^-_{i,m}=1} \rho^-_{i,m} = 1$ and $\sum_{m\in M, I^+_{i,m}=1} \rho^+_{i,m} = 1$. For example, if a mixing task called Mix executed in unit $Mixer$ has two inputs coming from states $S1$ and $S2$, then fraction $\rho^-_{Mix,S1}$ of $b_{Mix,Mixer,t}$ is the quantity of materials coming from state $S1$ whereas fraction $\rho^-_{Mix,S2}$ of $b_{Mix,Mixer,t}$ represents the quantity of materials coming from state $S2$. Accordingly, the values of $\rho^-_{Mix,S1}$ and $\rho^-_{Mix,S2}$ must be such that $\rho^-_{Mix,S1} + \rho^-_{Mix,S2} = 1$. Eq. 7.1F was proposed by Velez and Maravelias [361] to facilitate the branching process and efficiency of MIP algorithms such as CPLEX when solving purely scheduling problems. This constraint introduces new ordered discrete decisions ($n_{i,j} \in N_{i,j}$) that can be considered in the master problem of the LD-SDA (see section 7.2.3). These new ordered discrete decisions are batching variables $n_{i,j} \in N_{i,j}$, where $N_{i,j} = \{0,1,\dots,\lfloor n_T/\lceil \tau^{P,min}_{i,j}/\hat{\delta}\rceil\rfloor\}$. Note that the upper bound of set $N_{i,j}$ uses $\lceil\cdot\rceil$ and $\lfloor\cdot\rfloor$ to represent the ceil and floor operators. Eqs. 7.1G-7.1H enable the modeling of continuous variable processing times $\tau^P_{i,j,t}$, which are restricted by user-defined lower ($\tau^{P,min}_{i,j}$) and upper bounds ($\tau^{P,max}_{i,j}$). Eq. 7.1G indicates that, when a task i is executed in unit j at time t (i.e., $x_{i,j,t} = 1$), then its processing time $\tau^P_{i,j,t}$ is bounded below by $\tau^{P,min}_{i,j}$. Due to the discrete-time nature of the formulation, it is also important to relate the number of discrete points required by the execution of a task (i.e., $\tau_{i,j}$) with the actual processing time of the task (i.e., $\tau^P_{i,j,t}$), which is achieved through disjunction shown in Eq. 7.1H. This disjunction states that the optimization must find exactly one value of $\tau_{i,j}$ from $\mathrm{T}_{i,j}$ for each pair $(i,j) \in IJ^{\times}$, where $\mathrm{T}_{i,j} = \{\lceil \tau^{P,min}_{i,j}/\hat{\delta}\rceil, \lceil \tau^{P,min}_{i,j}/\hat{\delta}\rceil + 1, \dots, \lceil \tau^{P,max}_{i,j}/\hat{\delta}\rceil\}$. The selected value of $\tau_{i,j}$ is used to define auxiliary variables $x^{sum}_{i,j,t}$ and $b^{shift}_{i,j,t}$, and to bound $\tau^P_{i,j,t}$ from above through $\tau^P_{i,j,t} \leqq \min\{\hat{\delta}\tau_{i,j}, \tau^{P,max}_{i,j}\}$. The proposed formulation gives flexibility depending on the user's needs to choose between a pure discrete-time representation when the equality form of constraint $\tau^P_{i,j,t} \leqq \min\{\hat{\delta}\tau_{i,j}, \tau^{P,max}_{i,j}\}$ in Eq. 7.1H is used (i.e., $\tau^P_{i,j,t} = \min\{\hat{\delta}\tau_{i,j}, \tau^{P,max}_{i,j}\}$), or a mixed-time formulation when the inequality form is considered (i.e., $\tau^P_{i,j,t} \leq \min\{\hat{\delta}\tau_{i,j}, \tau^{P,max}_{i,j}\}$). In the former case, tasks are forced to finish at fixed time points, whereas in the latter tasks are allowed to finish any time within interval $\tau^{P,min}_{i,j} \leq \tau^P_{i,j,t} \leq \min\{\hat{\delta}\tau_{i,j}, \tau^{P,max}_{i,j}\}$. Also, task modes do not need to be defined by the user given that all the equations needed to model variable processing times are defined based on the values of $\tau^{P,min}_{i,j}$ and $\tau^{P,max}_{i,j}$. Note

that tasks with fixed processing times can be modeled by setting $\tau_{i,j}^{P,max} = \tau_{i,j}^{P,min} = \tau_{i,j}^{P,fixed}$. It is assumed in this work that there are dynamic models available for task-unit pairs $(i,j) \in IJ^{\times}$ with variable processing times, while the remaining pairs (i,j) have fixed processing times $\tau_{i,j}^{P,fixed}$. This assumption is stated mathematically in Eqs. 7.2A and 7.2B, where the subset of task-unit pairs with dynamic models is denoted by $IJ^{\times,dyn} \subseteq IJ^{\times}$.

$$0 \leq \tau_{i,j,t}^{P} \leq \tau_{i,j}^{P,max}, \forall (i,j) \in IJ^{\times,dyn}, \forall t \in T \tag{7.2A}$$

$$\tau_{i,j}^{P,max} = \tau_{i,j}^{P,min} = \tau_{i,j}^{P,fixed}, \forall (i,j) \in IJ^{\times} \backslash IJ^{\times,dyn} \tag{7.2B}$$

7.2.2 Mathematical formulation of the SSDO problem

A general SSDO problem is formulated through Eqs. 7.3A-7.3D, based on the discrete-time STN scheduling formulation with variable processing times described above. According to Eq. 7.3A, the problem considers the minimization of an objective function f that depends on scheduling variables ($\boldsymbol{v}_{\tau_{i,j},i,j,m,t}^{sch}$), and variables that capture the transient operation of tasks ($\boldsymbol{v}_{i,j,t}^{dyn}$). The vector of scheduling variables $\boldsymbol{v}_{\tau_{i,j},i,j,m,t}^{sch}$ includes those variables introduced in the previous section, i.e., $Y_{\tau_{i,j},i,j}, n_{i,j}, x_{i,j,t}, x_{i,j,t}^{sum}, s_{m,t}, \tau_{i,j,t}^{P}$, $b_{i,j,t}$, and $b_{i,j,t}^{shift}$. These variables belong to set SCH, which is given by Eqs. 7.1A-7.1H, 7.2A and 7.2B (see Eq. 7.3B). The reader is referred to section 7.2.1 for more details about scheduling variables $\boldsymbol{v}_{\tau_{i,j},i,j,m,t}^{sch}$ and scheduling constraints in SCH. Variables in $\boldsymbol{v}_{i,j,t}^{dyn}$ include the time-dependent vector of manipulated variables or inputs $\boldsymbol{\mu}_{i,j,t}$ (e.g., flow of utility streams); the time-dependent vectors of state variables $\boldsymbol{\chi}_{i,j,t}$ and their derivatives $\dot{\boldsymbol{\chi}}_{i,j,t}$ (e.g., concentrations, temperatures, etc.); the time-dependent vector of algebraic variables $\boldsymbol{\zeta}_{i,j,t}$ (e.g., enthalpies, reaction rates, etc.), and a vector of time-independent continuous variables $\boldsymbol{\varphi}_{i,j,t}$ (e.g., total hold-up, total task duration). According to Eq. 7.3C, these variables belong to the set DYN, which is defined by the differential equations, equality and inequality constraints imposed over task-unit pairs that belong to set $IJ^{\times,dyn}$, e.g., initial conditions, lower and upper bounds imposed over manipulated variables, and differential equations describing the transient evolution of the states of the processing units. The last group of constraints in Eq 7.3D are linking constraints, which are required to couple scheduling variables $\boldsymbol{v}_{\tau_{i,j},i,j,m,t}^{sch}$ with dynamic variables $\boldsymbol{v}_{i,j,t}^{dyn}$. For instance, when a task i is executed in unit j at

time t (i.e., $x_{i,j,t} = 1$), constraints must be enforced to guarantee that the scheduling-related batch size ($b_{i,j,t}$) matches with the dynamics-related total hold-up, and that the scheduling-related processing time ($\tau^P_{i,j,t}$) agrees with the dynamics-related total task duration.

$$\min_{s.t.} f\left(\boldsymbol{v}^{sch}_{\tau_{i,j},i,j,m,t}, \boldsymbol{v}^{dyn}_{i,j,t}\right) \tag{7.3A}$$

$$\boldsymbol{v}^{sch}_{\tau_{i,j},i,j,m,t} = \left[Y_{\tau_{i,j},i,j}, n_{i,j}, x_{i,j,t}, x^{sum}_{i,j,t}, s_{m,t}, \tau^P_{i,j,t}, b_{i,j,t}, b^{shift}_{i,j,t}\right] \in SCH, \quad \forall (i,j) \in IJ^{\times}, \forall t \in T, \forall \tau_{i,j} \in \mathrm{T}_{i,j}, \forall m \in M \tag{7.3B}$$

$$\boldsymbol{v}^{dyn}_{i,j,t} = \left[\boldsymbol{\mu}_{i,j,t}, \boldsymbol{\chi}_{i,j,t}, \dot{\boldsymbol{\chi}}_{i,j,t}, \boldsymbol{\zeta}_{i,j,t}, \boldsymbol{\varphi}_{i,j,t}\right] \in DYN, \forall (i,j) \in IJ^{\times,dyn}, \forall t \in T \tag{7.3C}$$

$$\left[\boldsymbol{v}^{sch}_{\tau_{i,j},i,j,m,t}, \boldsymbol{v}^{dyn}_{i,j,t}\right] \in LINK, \forall (i,j) \in IJ^{\times}, \forall t \in T, \forall \tau_{i,j} \in \mathrm{T}_{i,j}, \forall m \in M \tag{7.3D}$$

In summary, the formulation presented above tackles the different challenges related to integrating continuous processing times into discrete-time formulations for SSDO. The first challenge solved by the proposed formulation is that it avoids the need of manually declaring *task modes*, i.e., introducing new tasks into the formulation, each with a different discrete processing time. Instead, the user only needs to declare maximum ($\tau^{P,max}_{i,j}$) and minimum ($\tau^{P,min}_{i,j}$) processing time bounds, which are used in Eq. 7.1H to automatically generate disjunctions for task-unit pairs, and as search bounds for variable processing times $\tau^P_{i,j,t}$ through Eq. 7.2A. These disjunctions contain disjunct terms that, depending on the user requirements, can be used to either optimize discrete processing times when $\tau^P_{i,j,t} = \min\{\hat{\delta}\tau_{i,j}, \tau^{P,max}_{i,j}\}$, or the upper bound of continuous processing times when $\tau^P_{i,j,t} \leq \min\{\hat{\delta}\tau_{i,j}, \tau^{P,max}_{i,j}\}$. Having explicit variable processing times $\tau^P_{i,j,t}$ in the scheduling formulation also address the implementation challenge of connecting scheduling processing times with the operation times for time- dependent units whose behavior is described by dynamic models. This connection is straightforward by using linking constraints in Eq. 7.3D. For instance, a linking constraint of the form $-\mathrm{M}(1 - x_{i,j,t}) \leq \tau^P_{i,j,t} - \varphi^P_{i,j} \leq \mathrm{M}(1 - x_{i,j,t})$ would guarantee that $\tau^P_{i,j,t} = \varphi^P_{i,j}$ when $x_{i,j,t} = 1$, where $\varphi^P_{i,j}$ is the processing time counterpart in the dynamic model and M is a big-M parameter.

The proposed solution framework also addresses a key challenge, i.e., efficient exploration of the search space of the newly introduced variable processing times. To tackle this obstacle, I have developed the LD-SDA, given that the solution of the optimization problem discussed above with traditional MINLP solvers exhibits multiple challenges. A direct MINLP solver would be unable to leverage the advantages of the proposed GDP model, e.g., a formulation that involves less variables and constraints than a conventional Big-M or convex hull reformulation when Boolean variables remain fixed. In addition, the consideration of variable processing times may worsen multiplicity issues, i.e., solutions with identical objective function values. For example, instead of having multiplicity for fixed processing times, multiplicity may now appear for multiple fixed combinations of processing times. Similarly, the infeasible search space of the problem is expected to grow with the consideration of variable processing times. For instance, short processing times may cause infeasibilities in Eq. 7.3C, e.g., a desired product composition could not be achieved without violating a bound imposed over a manipulated variable. Consequently, MINLP solvers may spend multiple iterations evaluating infeasible or degenerate combinations of the discrete decisions that appear in the formulation before closing their optimality gap. These issues have motivated the development of a LD-SDA method proposed in this work to tackle problem in Eqs. 7.3A-7.3D. The main feature of the LD-SDA that distinguishes this method from standard MINLP techniques is that it relies on alternative convexity requirements. Although it is a local algorithm, the proposed LD-SDA relies on integral convexity from discrete convex analysis, rather than relying on the convexity of the Relaxed-MINLP to guarantee global optimality. For instance, the LD-SDA is expected to result in enhanced performance when compared to the Generalized Benders Decomposition (GBD) strategy, which is the state-of-the-art solution strategy employed in the solution of network SSDO problems, e.g., see Chu and You [61,355] and Nie et al.[62,366]. To illustrate this, a simple MINLP problem is discussed in section G.4 of appendix G. This illustrative MINLP example shows that when both LD-SDA and GBD are initialized from the same point, the LD-SDA converges to the global optimum of the problem, while GBD stagnates at the initialization. This behavior is attributed to the fact that this illustrative MINLP problem is integrally convex, however, its Relaxed-MINLP is not. Motivated by its convexity properties, and the computational advantages that the LD-SDA offers over the traditional MINLP solvers, the extension of this algorithm to SSDO problems is discussed next.

7.2.3 Solution of the problem using the LD-SDA: reformulation and decomposition

The LD-SDA addresses the issues faced by MINLP solvers discussed in the previous section when solving Eqs. 7.3A-7.3D by relying on a customized reformulation, decomposition and algorithmic framework. The LD-SDA decomposes the problem into a master problem that optimizes ordered discrete decisions, and a subproblem that optimizes the remaining constraints. The ordered discrete decisions may not appear explicitly in the formulation, thus a reformulation of the problem is usually required[4]. In general, ordered discrete decisions may appear as explicit integer variables, or as groups of binary/Boolean variables defined over ordered sets plus an assignment constraint[15]. In this case, there is a mixture of both, batching variables $n_{i,j}, \forall (i,j) \in IJ^{\times}$ (eq 1F) are explicit ordered discrete decisions, whereas Boolean variables $Y_{\tau_{i,j},i,j}, \forall (i,j) \in IJ^{\times}$ are ordered discrete decisions with respect to index $\tau_{i,j} \in \mathrm{T}_{i,j}$ (Eq. 7.1H), where $\mathrm{T}_{i,j}$ can be regarded as an ordered set. Variables $Y_{\tau_{i,j},i,j}, \forall (i,j) \in IJ^{\times}$ can thus be reformulated in terms of external variables $e_{i,j}, \forall (i,j) \in IJ^{\times}$, i.e.,

$$e_{i,j} \in \mathrm{T}_{i,j} = \left\{ \left\lceil \tau_{i,j}^{P,min} / \hat{\delta} \right\rceil, \left\lceil \tau_{i,j}^{P,min} / \hat{\delta} \right\rceil + 1, \dots, \left\lceil \tau_{i,j}^{P,max} / \hat{\delta} \right\rceil \right\}, \forall (i,j) \in IJ^{\times} \tag{7.4A}$$

$$Y_{\tau_{i,j},i,j} = \begin{cases} True, \tau_{i,j} = e_{i,j} \\ False, \tau_{i,j} \neq e_{i,j} \end{cases}, \forall (i,j) \in IJ^{\times}, \forall \tau_{i,j} \in \mathrm{T}_{i,j} \tag{7.4B}$$

External variable $e_{i,j}$ has a more intuitive meaning that variables $Y_{\tau_{i,j},i,j}, \forall (i,j) \in IJ^{\times}$, since it explicitly optimizes the number of discretization points needed to execute task i in unit j. Note that if external variables $e_{i,j}, \forall \in IJ^{\times}$are fixed at a feasible point that satisfies Eq. 7.4A, then all the values of $Y_{\tau_{i,j},i,j}, \forall (i,j) \in IJ^{\times}, \forall \tau_{i,j} \in \mathrm{T}_{i,j}$ will be given by eq. 7.4B, i.e., $Y_{\tau_{i,j},i,j} = Y_{\tau_{i,j},i,j}\left(e_{i,j}\right), \forall (i,j) \in IJ^{\times}, \forall \tau_{i,j} \in \mathrm{T}_{i,j}$. Thus, the SSDO problem can be solved by optimizing external variables $e_{i,j}$ and fixing dependent variables $Y_{\tau_{i,j},i,j}$ at each iteration. This means that LD-SDA can take advantage of the proposed GDP formulation by solving subproblems in reduced space, which is a feature that cannot be exploited with the traditional MINLP solvers. Another advantage of this reformulation compared to the direct MINLP solution of the problem is that the total number of variables decreases, i.e., the original non-reformulated problem has $\sum_{(i,j) \in IJ^{\times}} |\mathrm{T}_{i,j}|$ Boolean variables, while the reformulated problem has $\sum_{(i,j) \in IJ^{\times}} 1 = |IJ^{\times}|$ external variables.

As suggested in previous works, ordered discrete decisions such as $n_{i,j}$ and $e_{i,j}$ can be optimized efficiently by moving those variables into a master problem, and optimizing them using discrete-steepest descent optimization[4]. In this case, the master-subproblem decomposition of the problem given by Eqs. 7.3A-7.3D, and 7.4A-7.4B results in the master problem defined by Eq. 7.5A and the subproblem defined by Eq. 7.5B. The master problem (Eq. 7.5A) aims to optimize ordered discrete decisions $n_{i,j}$ and $e_{i,j}$. These variables are restricted in the master problem within their corresponding bounds, i.e., $n_{i,j} \in N_{i,j}$ and $e_{i,j} \in \mathrm{T}_{i,j}$. Note that $Y_{\tau_{i,j},i,j}$ can be calculated as a function of $e_{i,j}$ (see Eq. 7.4B), hence $Y_{\tau_{i,j},i,j} = Y_{\tau_{i,j},i,j}(e_{i,j})$ is also included in the master problem. The subproblem (Eq. 7.5B) solves the original problem in Eqs. 7.3A-7.3D with variables $n_{i,j}$ and $Y_{\tau_{i,j},i,j}$ fixed at the values specified by the master problem. The solution of a subproblem returns the objective function value $f^*\left(n_{i,j}, Y_{\tau_{i,j},i,j}(e_{i,j})\right) = f\left(\boldsymbol{v}^{sch*}_{\tau_{i,j},i,j,k,t}, \boldsymbol{v}^{dyn*}_{i,j,t}\right)$, where $\boldsymbol{v}^{sch*}_{\tau_{i,j},i,j,k,t}, \boldsymbol{v}^{dyn*}_{i,j,t}$ is the optimal point returned by subproblem 7.5B, i.e., the objective function considered by the master problem in Eq. 7.5A (f^*) corresponds to the evaluation of objective function f at the optimal solution returned by subproblem 7.5B, for a fixed combination of $n_{i,j}$ and $Y_{\tau_{i,j},i,j}(e_{i,j})$. As a convention, $f^* = +\infty$ for combinations of $n_{i,j}$ and $e_{i,j}$ declared as infeasible either by the master problem or the subproblem.

$$\min_{s.t.} f^*\left(n_{i,j}, Y_{\tau_{i,j},i,j}(e_{i,j})\right) \tag{7.5A}$$
$$n_{i,j} \in N_{i,j} = \left\{0,1,\dots,\left\lfloor n_T / \left\lceil \tau^{P,min}_{i,j}/\hat{\delta} \right\rceil \right\rfloor\right\}, \forall (i,j) \in IJ^{\times}$$
$$e_{i,j} \in \mathrm{T}_{i,j} = \left\{\left\lceil \tau^{P,min}_{i,j}/\hat{\delta} \right\rceil, \left\lceil \tau^{P,min}_{i,j}/\hat{\delta} \right\rceil + 1, \dots, \left\lceil \tau^{P,max}_{i,j}/\hat{\delta} \right\rceil\right\}, \forall (i,j) \in IJ^{\times}$$
$$Y_{\tau_{i,j},i,j} = \begin{cases} True, \tau_{i,j} = e_{i,j} \\ False, \tau_{i,j} \neq e_{i,j} \end{cases}, \forall (i,j) \in IJ^{\times}, \forall \tau_{i,j} \in \mathrm{T}_{i,j}$$

$$f^*\left(n_{i,j}, Y_{\tau_{i,j},i,j}(e_{i,j})\right) = \min_{s.t.} f\left(\boldsymbol{v}^{sch}_{\tau_{i,j},i,j,k,t}, \boldsymbol{v}^{dyn}_{i,j,t}\right) \tag{7.5B}$$
$$\boldsymbol{v}^{sch}_{\tau_{i,j},i,j,k,t} \in SCH, \forall (i,j) \in IJ^{\times}, \forall t \in T, \forall \tau_{i,j} \in \mathrm{T}_{i,j}, \forall k \in K$$
$$\boldsymbol{v}^{dyn}_{i,j,t} \in DYN, \forall (i,j) \in IJ^{\times,dyn}, \forall t \in T$$
$$\left[\boldsymbol{v}^{sch}_{\tau_{i,j},i,j,k,t}, \boldsymbol{v}^{dyn}_{i,j,t}\right] \in LINK, \forall (i,j) \in IJ^{\times}, \forall t \in T, \forall \tau_{i,j} \in \mathrm{T}_{i,j}, \forall k \in K$$

7.2.4 Solution of the problem using the LD-SDA: Algorithm steps

The LD-SDA takes advantage of the ordered structure of variables $n_{i,j}$ and $e_{i,j}$, and also benefits from the fact that a simpler optimization problem is obtained when these ordered variables remain fixed. The LD-SDA explores the search space of $n_{i,j}$ and $e_{i,j}$ in the master problem (Eq. 7.5A) and relies on a general-purpose solver for the subproblems, which in this case are MINLPs given that all Boolean variables remain fixed. A flowchart summarizing the LD-SDA steps required to solve the SSDO problem is shown in Figure 7.1. Note that vectors $\boldsymbol{n}^k$ and $\boldsymbol{e}^k$ contain the values of variables $n_{i,j}$, $\forall (i,j) \in IJ^{\times}$ and $e_{i,j}$, $\forall (i,j) \in IJ^{\times}$ at the current iteration, respectively. The LD-SDA begins with a user-defined combination of ordered variables $\boldsymbol{n}^k$ and $\boldsymbol{e}^k$. The algorithm then needs to update these ordered variables, which in this case is performed through neighborhood exploration. A neighborhood verification is performed in step 1, which evaluates the sensitivity of the problem when varying $\boldsymbol{n}^k$ and $\boldsymbol{e}^k$ within a unit of their current value, i.e., in a similar way as gradients provide neighborhood information for continuous variables. This neighborhood verification is performed by solving the subproblem in Eq. 7.5B at every point within $N_2(\boldsymbol{n}^k, \boldsymbol{e}^k) = \left\{[\boldsymbol{n}, \boldsymbol{e}] \in \mathbb{Z}^2 : \left\|[\boldsymbol{n} - \boldsymbol{n}^k, \boldsymbol{e} - \boldsymbol{e}^k]\right\|_2 \leq 1\right\}$, and storing the f^* values obtained. Step 2 then verifies if $f^*(\boldsymbol{n}^k, \boldsymbol{e}^k)$ takes the minimum value within $N_2(\boldsymbol{n}^k, \boldsymbol{e}^k)$, based on a user-defined tolerance ϵ. That is, if $f^*(\boldsymbol{n}^k, \boldsymbol{e}^k) - f^*(\boldsymbol{n}, \boldsymbol{e}) \leq \epsilon \; \forall (\boldsymbol{n}, \boldsymbol{e}) \in N_2(\boldsymbol{n}^k, \boldsymbol{e}^k)$, then there is no direction of significant improvement with respect to tolerance ϵ and the algorithm stops. In this case, the algorithm returns a locally optimal solution $(\boldsymbol{n}^k, \boldsymbol{e}^k)$ with respect to neighborhood $N_2(\boldsymbol{n}^k, \boldsymbol{e}^k)$. Otherwise, the steepest descent direction $\boldsymbol{\delta} = [\boldsymbol{n}^o - \boldsymbol{n}^k, \boldsymbol{e}^o - \boldsymbol{e}^k]$ is calculated in step 3, where $[\boldsymbol{n}^o, \boldsymbol{e}^o]$ is the vector of ordered discrete decisions that minimizes f^* within $N_2(\boldsymbol{n}^k, \boldsymbol{e}^k)$. Step 4 then solves subproblems (Eq. 7.5B) by sequentially fixing ordered discrete decisions $(\boldsymbol{n}, \boldsymbol{e})$ along the steepest descent direction $\boldsymbol{\delta}$, until there is no improvement in f^*. Vector $[\boldsymbol{n}^k, \boldsymbol{e}^k]$ is then updated using the combination of ordered discrete decisions that minimized f^* along direction $\boldsymbol{\delta}$, and a new iteration *k* is performed by returning to Step 1.

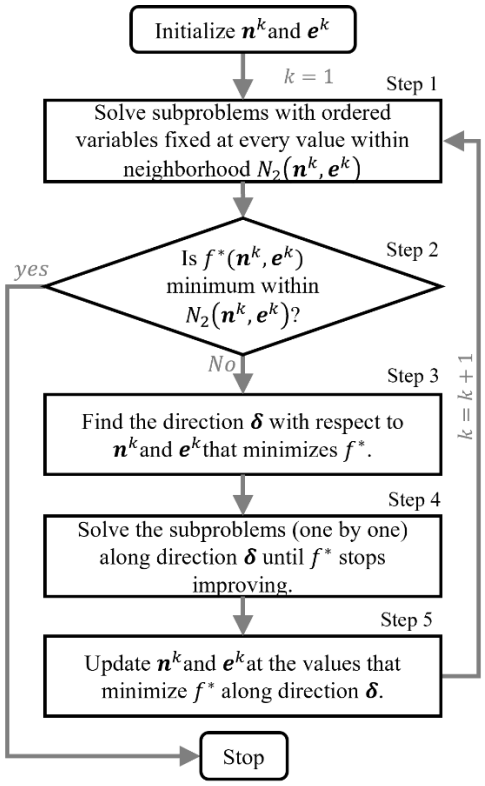


Figure 7.1. The LD-SDA applied to the SSDO formulation in eqs 7.5A and 7.5B.

The scalability of the ordered discrete decisions within the proposed solution framework may be impacted by different factors. The number of batching variables $n_{i,j}$ increases by one every time a new task, a new unit, or a new task-unit pair is added to the formulation. Similarly, the number of external variables $e_{i,j}$increases by one every time a new dynamic task, unit, or task-unit pair with variable processing time is included to the problem. In addition, external variables $e_{i,j}$ help to reduce the combinatorial complexity of the problem by avoiding the explicit exploration of Boolean variables related to variable processing times, as discussed in section 7.2.3. The number of dynamic models in the problem depends on the assumptions imposed over the recipe data generated by the solution of the SSDO problem. In general, this recipe data is given by the optimal values of the time-independent continuous variables $\boldsymbol{\varphi}_{i,j,t}$ (e.g., task durations) and control actions $\boldsymbol{\mu}_{i,j,t}$ needed to run a task i at unit j starting at time t. Alternatively, it can be assumed that recipe data does not change with time for a given unit, i.e., $\boldsymbol{\varphi}_{i,j,t} = \boldsymbol{\varphi}_{i,j}$ and $\boldsymbol{\mu}_{i,j,t} = \boldsymbol{\mu}_{i,j}$. In the former case, the total number of dynamic models in the formulation will be equal to the number of dynamic task-unit pairs multiplied by the number of

scheduling time slots, i.e., $\left|IJ^{\times,dyn}\right| * (n_T + 1)$. In the latter case, the total number of dynamic models will match with the size of set $IJ^{\times,dyn}$. The neighborhood selected for exploration is another factor that affects scalability. I have selected the N_2 neighborhood in the proposed framework, which considers the individual effect of changing the value of an ordered discrete decision in the objective function, i.e., in a similar way partial derivatives and Lagrange multipliers are used to evaluate the sensitivity of variables or constraints on a function. This neighborhood has been selected to avoid scalability issues, as it only grows linearly in size as newly ordered discrete variables are added to the formulation. Nonetheless, alternative neighborhoods such as the integrally convex neighborhood N_∞ are also available, whose size grows exponentially as new external variables appear in the formulation[15]. Exploring within N_∞ neighborhoods is expected to yield better solutions at the expense of longer computational costs, due to the exploration of alternative neighbors, e.g., simultaneously changing the value of two ordered discrete variables.

The solution of the problem in Eqs. 7.5A and 7.5B using the LD-SDA combined with a local MINLP solver for the subproblems offers advantages with respect to the conventional solution of the problem in Eqs. 7.3A-7.3D, i.e., using an appropriate reformulation of disjunctions, e.g., Big-M, followed by the direct execution of a local MINLP solver. First, the search region of variables $n_{i,j}$ and $Y_{\tau_{i,j},i,j}$ may be large and contain multiple infeasibilities or multiplicities, i.e., combinations of variables that lead to the same objective function value. The LD-SDA avoids combinations of variables $n_{i,j}$ and $Y_{\tau_{i,j},i,j}$ that lead to infeasibilities, because it solves the problem while staying within the feasible region or ordered discrete decisions (i.e., $N_{i,j}$ and $\mathrm{T}_{i,j}$), and avoiding search directions that lead to infeasibilities when detected by the subproblem. Similarly, search directions that lead to multiplicities are avoided since these directions do not result in an improvement in the objective function. Second, the local optimality of variables $n_{i,j}$ and $Y_{\tau_{i,j},i,j}(e_{i,j})$ in the LD-SDA is dictated by a local neighborhood verification, which is a feature that is not considered by traditional MINLP strategies. This may help the LD-SDA to avoid stopping prematurely at suboptimal solutions that do not satisfy the neighborhood optimality criterion, but may satisfy commonly used stopping conditions, e.g., crossover between an upper bound and a lower bound estimate. Note that only local optimality can be guaranteed with the LD-SDA, where the optimality of ordered discrete decision is dictated by the neighborhood verification in the master problem, while the optimality of the remaining variables is dictated by the subproblem solver. The LD-SDA can also be combined with a global MINLP solver for the subproblems; however, this feature is

not studied in this work and left for future studies. It is also important to acknowledge that the LD-SDA requires a feasible combination of ordered discrete decisions as the initialization stage. In theory, this requirement is not needed by standard MINLP solvers; however, they commonly require customized initializations to facilitate convergence in practice. To circumvent this issue, feasible initializations are generated using sequential scheduling and dynamic optimization strategies in this work. The first strategy considered is a sequential initialization approach (SIA). In this strategy, batch sizes are heuristically fixed first. This is followed by a minimization of the processing times of the batch processes involved through NLP optimization, then, the optimization of the plant schedule through MIP optimization is performed. The second strategy is an iterative initialization approach (IIA). In this case, the strategy iteratively updates external variables $\boldsymbol{e}$ and solves NLP dynamic optimization problems and MIP scheduling problems until feasibility of the NLP and MIP problems are attained. More details about these initialization strategies are provided in section G.1 and Table G.1 of appendix G.

7.3 Computational Experiments

Two case studies adapted from the literature are used to illustrate the proposed modeling and LD-SDA solution framework. The first case study considers the STN originally proposed by Kondili et al.[360], and incorporates one of the semi-batch kinetic reaction models considered by Srinivasan et al.[367] in one of the reaction tasks involved in the process. The second case study adapts the STN and batch reaction dynamic models proposed by Chu and You[61] to the proposed discrete-time SSDO formulation. The economic objective function defined through Eqs. 7.6A-7.6F is considered for both case studies. This objective function (Eq. 7.6A) involves the minimization of the sum of operation costs ($c^{fix} + c^{var_1} + c^{var_2} + c^{mat}$) minus the revenue ($rev$), i.e., the objective function in Eq. 7.6A is equivalent to maximizing profit. The fixed cost term (c^{fix}) is calculated using Eq. 7.6B, which adds the fixed cost $c_{i,j}^{fix}$ of executing task i in unit j for every $i \in I, j \in J$ and $t \in T$. Variable costs c^{var_1} and c^{var_2} in Eqs. 7.6C and 7.6D distinguish between variable costs for task-unit pairs without and with the dynamic models available, respectively. Cost c^{var_1} in Eq. 7.6C assumes that variable costs are proportional to the batch size, with coefficients $c_{i,j}^{var_1}, \forall (i,j) \in IJ^{\times} \backslash IJ^{\times dyn}$. Cost c^{var_2} related to the transient operation of tasks assumes that there are heating and cooling utilities available, whose flows $\mu_{i,j,t}^{hot}$ and $\mu_{i,j,t}^{cold}$ can be manipulated. The integral of flows $\mu_{i,j,t}^{hot}$ and $\mu_{i,j,t}^{cold}$ over dimensionless time ψ in Eq. 7.6D are multiplied by coefficients c^{hot} (10 $\$/m^3$) and c^{cold} (1 $\$/m^3$) to obtain their costs. c^{mat} in Eq.

7.6E represents the cost of raw materials $m \in M^{in}$, where M^{in} is a subset of states M and c_m^{mat} is the cost of purchasing material in state m. The total amount of raw materials consumed is given by the difference between inventory of state m at the beginning and the end of the scheduling horizon. The revenue of the process is calculated using Eq. 7.6F, which assumes that the inventories of products $m \in M^{out} \in M$ are sold at the end of the scheduling horizon. The cost of these products is given by $c_m^{out}, \forall m \in M^{out}$.

$$f = c^{fix} + c^{var_1} + c^{var_2} + c^{mat} - rev \tag{7.6A}$$

$$c^{fix} = \sum_{i \in I} \sum_{j \in J} \sum_{t \in T} c_{i,j}^{fix} x_{i,j,t} \tag{7.6B}$$

$$c^{var_1} = \sum_{(i,j) \in IJ^{\times} \setminus IJ^{\times dyn}} \sum_{t \in T} c_{i,j}^{var_1} b_{i,j,t} \tag{7.6C}$$

$$c^{var_2} = \sum_{(i,j) \in IJ^{\times dyn}} \sum_{t \in T} x_{i,j,t} \tau_{i,j,t}^{P} \left(c^{hot} \int_0^1 \mu_{i,j,t}^{hot} \, d\psi + c^{cold} \int_0^1 \mu_{i,j,t}^{cold} \, d\psi \right) \tag{7.6D}$$

$$c^{mat} = \sum_{m \in M^{in}} c_m^{mat} \left(s_m^{init} - s_{m,n_T} \right) \tag{7.6E}$$

$$rev = \sum_{m \in M^{out}} c_m^{out} s_{m,n_T} \tag{7.6F}$$

The computational experiments performed in this work were implemented in Pyomo in a PC Intel® Core™ i7-9700K CPU @ 3.60 GHz with 64 GB of RAM. Solvers DICOPT, CPLEX and CONOPT4 were accessed through GAMS and used as MINLP, MIP and NLP solvers respectively. All the computational experiments that made use of MIP or MINLP solvers were conducted with a relative optimality gap set to 0, using the default numerical precision employed by the DICOPT and CPLEX solvers. The LD-SDA stopping tolerance was also set to 0. Also, a time limit of 86400 seconds (1 day) was imposed for all computational experiments. Disjunctions were modeled using Pyomo.GDP[13], and reformulated using the Big-M method when needed. The differential equations of both problems were implemented and discretized using Pyomo.DAE[368] with the Lagrange-Radau discretization, using 3 collocation points and 60 and 30 finite elements for case studies 1 and 2, respectively.

7.3.1 Case study 1

The STN considered for this case study is illustrated in Figure 7.2 and consists of five tasks $I = \{T1, T2, \dots, T5\}$, nine states $K = \{S1, S2, \dots, S9\}$, and four units $J = \{U1, U2, U3, U4\}$. Figure 7.2 also shows the task to units mapping that defines set $IJ^{\times} = \{(T1, U1), (T2, U2), (T3, U2), \dots\}$, and includes the percentages that allow to calculate parameters $\rho^{+}_{i,m}$ and $\rho^{-}_{i,m}$, e.g., $\rho^{-}_{T5,S7} = 1$, $\rho^{+}_{T5,S6} = 0.1$, $\rho^{+}_{T5,S9} = 0.9$. The task-unit pairs that consider a transient (dynamic) operation are given by $IJ^{\times dyn} = \{(T2, U2), (T2, U3)\}$. States representing raw materials and products are $M^{in} = \{S1, S2, S3\}$ and $M^{out} = \{S8, S9\}$, respectively. Tables G.2, G.3 and G.4 containing all the scheduling parameters are available in section G.2 of appendix G.

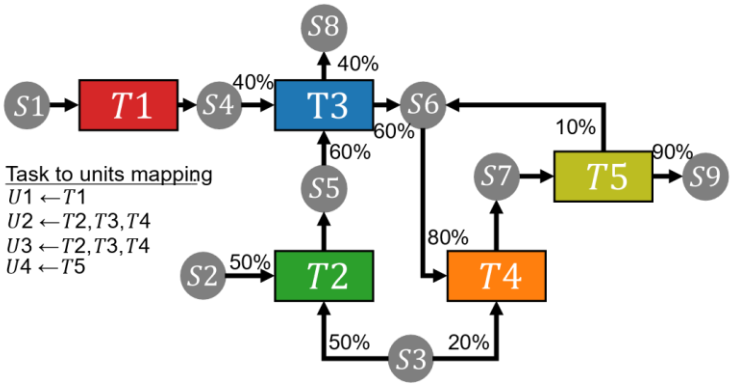


Figure 7.2. STN for case study 1. The execution of task $T2$ in $U2$ and $U3$ is described by a semi-batch reactor model.

A mixed-time formulation is considered for this case study, i.e., variable processing times are allowed to vary continuously in the discrete-time scheduling grid, i.e., $\tau^{P}_{i,j,t} \leq \min\{\hat{\delta}\tau_{i,j}, \tau^{P,max}_{i,j}\}$ in Eq. 7.1H. Units $U2$ and $U3$ are reactors with different capacities that operate in semi-batch mode when executing task $T2$. The exothermic-irreversible reactions involved in $T2$ are $A + B \rightarrow C$ and $C \rightarrow D$, where C is the desired product, and reactant B is assumed to be dosed through the operation of the semi-batch process. Reactant A is assumed to be coming from state $S2$, whereas reactant B is assumed to be coming from state $S3$. The differential-algebraic equations that define dynamic feasible region DYN are shown in Eq. 7.7A to eq 7.7H. Tables G.5 and G.6 provide a description of the different variables and parameters that appear in these equations (see appendix G). The notation ψ, which represents the time-scale of the differential equations, is presented in dimensionless form (i.e., $0 \leq \psi \leq 1$) to enable the

modeling of variable reaction times $\varphi^P_{i,j,t}$. These reaction times are assumed to be equal to the variable processing times $\tau^P_{i,j,t}$, i.e., the contribution of start-up and shut-down times to the processing time are neglected in this case study. Also, volumes of the different species involved are assumed to be additive.

Reaction rate of A in $A + B \rightarrow C$ (7.7A)

$$\zeta^A_{i,j,t} = -k_1 \exp\left(\frac{-Ea_1}{R\chi^{TR}_{i,j,t}}\right)\chi^A_{i,j,t}\chi^B_{i,j,t}, \forall (i,j) \in IJ^{\times dyn}, \forall t \in T$$

Reaction rate of C in $C \rightarrow D$ (7.7B)

$$\zeta^C_{i,j,t} = -k_2 \exp\left(\frac{-Ea_2}{R\chi^{TR}_{i,j,t}}\right)\chi^C_{i,j,t}, \forall (i,j) \in IJ^{\times dyn}, \forall t \in T$$

Concentration balance of A (7.7C)

$$\dot{\chi}^A_{i,j,t} = \varphi^P_{i,j,t}\left(\zeta^A_{i,j,t} - \frac{\mu^B_{i,j,t}}{\chi^{Vol}_{i,j,t}}\chi^A_{i,j,t}\right),\ \chi^A_{i,j,t}(\psi = 0) = \varphi^{A0}_{i,j,t}, \forall (i,j) \in IJ^{\times dyn}, \forall t \in T$$

Concentration balance of B (7.7D)

$$\dot{\chi}^B_{i,j,t} = \varphi^P_{i,j,t}\left(\zeta^A_{i,j,t} + \frac{\mu^B_{i,j,t}}{\chi^{Vol}_{i,j,t}}\left(C^B_{IN} - \chi^B_{i,j,t}\right)\right), \chi^B_{i,j,t}(\psi = 0) = \varphi^{B0}_{i,j,t},$$

$$\forall (i,j) \in IJ^{\times dyn}, \forall t \in T$$

Concentration balance of C (7.7E)

$$\dot{\chi}^C_{i,j,t} = \varphi^P_{i,j,t}\left(-\zeta^A_{i,j,t} + \zeta^C_{i,j,t} - \frac{\mu^B_{i,j,t}}{\chi^{Vol}_{i,j,t}}\chi^C_{i,j,t}\right),\ \chi^C_{i,j,t}(\psi = 0) = 0, \forall (i,j) \in IJ^{\times dyn}, \forall t \in T$$

Change in reactive mixture volume (7.7F)

$$\dot{\chi}^{Vol}_{i,j,t} = \varphi^P_{i,j,t}\left(\mu^B_{i,j,t}\right),\ \chi^{Vol}_{i,j,t}(\psi = 0) = V_0, \forall (i,j) \in IJ^{\times dyn}, \forall t \in T$$

Energy balance (reactor temperature) (7.7G)

$$\dot{\chi}^{TR}_{i,j,t} = \frac{\varphi^P_{i,j,t}}{\varphi^A_{i,j,t} + \varphi^B_{i,j,t}}\left(\frac{Ua\left(\chi^{TJ}_{i,j,t} - \chi^{TR}_{i,j,t}\right)}{V_0 C_P} - \frac{C^B_{IN}\mu^B_{i,j,t}\left(\chi^{TR}_{i,j,t} - T_{B,IN}\right)}{V_0} + \frac{\chi^{Vol}_{i,j,t}}{V_0 C_P}\left(\Delta H_1 \zeta^A_{i,j,t} + \Delta H_2 \zeta^C_{i,j,t}\right)\right),$$

$$\chi^{TR}_{i,j,t}(\psi = 0) = T_{R0}, \forall (i,j) \in IJ^{\times dyn}, \forall t \in T$$

Energy balance (jacket temperature) (7.7H)

$$\dot{\chi}_{i,j,t}^{TJ} = \varphi_{i,j,t}^{P}\left(\frac{\mu_{i,j,t}^{hot}\left(T_H-\chi_{i,j,t}^{TJ}\right)+\mu_{i,j,t}^{cold}\left(T_C-\chi_{i,j,t}^{TJ}\right)}{VJ_j}+\frac{Ua_j\left(\chi_{i,j,t}^{TR}-\chi_{i,j,t}^{TJ}\right)}{VJ_j\rho_J C_{PJ}}\right), \chi_{i,j,t}^{TJ}(\psi=0)=T_{J0},$$

$$\forall(i,j)\in IJ^{\times dyn}, \forall t\in T$$

The linking constraints required to couple scheduling and dynamic decisions denoted by feasible region $LINK$ in Eq. 7.3D are shown in Eqs. 7.8A-7.8E.

$$-\mathrm{M}_{1,i,j,t}^{min}\left(1-x_{i,j,t}\right)\le b_{i,j,t}-\chi_{i,j,t}^{Vol}(\psi=1)\le \mathrm{M}_{1,i,j,t}^{max}\left(1-x_{i,j,t}\right), \forall(i,j)\in IJ^{\times dyn}, \forall t\in T \quad (7.8A)$$

$$-\mathrm{M}_{2,i,j,t}^{min}\left(1-x_{i,j,t}\right)\le \rho_{T2,\mathrm{S2}}^{-}\chi_{i,j,t}^{Vol}(\psi=1)-\frac{\varphi_{i,j,t}^{A0}}{C_{IN}^{A}}V_0\le \mathrm{M}_{2,i,j,t}^{max}\left(1-x_{i,j,t}\right), \quad (7.8B)$$

$$\forall(i,j)\in IJ^{\times dyn}, \forall t\in T$$

$$-\mathrm{M}_{3,i,j,t}^{min}\left(1-x_{i,j,t}\right)\le \left(\rho_{T2,\mathrm{S3}}^{-}-1\right)\chi_{i,j,t}^{Vol}(\psi=1)-\left(\frac{\varphi_{i,j,t}^{B0}}{C_{IN}^{B}}-1\right)V_0\le \mathrm{M}_{3,i,j,t}^{max}\left(1-x_{i,j,t}\right), \quad (7.8C)$$

$$\forall(i,j)\in IJ^{\times dyn}, \forall t\in T$$

$$-\mathrm{M}_{4,i,j,t}^{min}\left(1-x_{i,j,t}\right)\le \chi_{i,j,t}^{C}(\psi=1)-C_{DES}^{C}\le \mathrm{M}_{4,i,j,t}^{max}\left(1-x_{i,j,t}\right), \quad (7.8D)$$

$$\forall(i,j)\in IJ^{\times dyn}, \forall t\in T$$

$$-\mathrm{M}_{5,i,j,t}^{min}\left(1-x_{i,j,t}\right)\le \tau_{i,j,t}^{P}-\varphi_{i,j,t}^{P}\le \mathrm{M}_{5,i,j,t}^{max}\left(1-x_{i,j,t}\right), \forall(i,j)\in IJ^{\times dyn}, \forall t\in T \quad (7.8E)$$

where the $\mathrm{M}_{con,i,j,t}^{min}$and $\mathrm{M}_{con,i,j,t}^{max}$ in Eqs. 7.8A-7.8E represent the big-M constants of constraint con. Note that values for $\mathrm{M}_{con,i,j,t}^{min}$ and $\mathrm{M}_{con,i,j,t}^{max}$ were found for every constraint, based on the bounds imposed over each variable. Eqs. 7.8A-7.8D aim to link scheduling and dynamic decisions only when $x_{i,j,t}=1$, indicating that task i is executed in unit j at time t. When $x_{i,j,t}=1$, Eq. 7.8A ensures that the scheduling batch size $b_{i,j,t}$ is equal to $\chi_{i,j,t}^{Vol}(\psi=1)$, which is the counterpart of $b_{i,j,t}$ in the dynamic model. Eqs. 7.8B and 7.8C link the concentrations of A and B at states $S2$ and $S3$ with the initial amounts of A and B in the reactor, e.g., Eq. 7.8B coupled with Eq. 7.8A ensure that the amount of A

coming from state $S2$ ($\rho^{-}_{T2,S2} b_{i,j,t} C^{A}_{IN}$) is equal to total amount of A initially fed to the reactor ($\varphi^{A0}_{i,j,t} V_0$) when $x_{i,j,t} = 1$. Eq. 7.8D guarantees that the desired concentration of C in the reactor product is attained, while Eq. 7.8E ensures that reaction time $\varphi^{P}_{i,j,t}$ is equal to processing time $\tau^{P}_{i,j,t}$ when $x_{i,j,t} = 1$. This case study considers a scheduling horizon η of 5 hours, and a discretized time set with $n_T = 15$. Table G.7 in appendix G shows the specific details about the reformulation and ordered discrete decisions of the problem. The SIA and IIA initialization strategies described in section 7.2.4 were used to generate feasible initializations. In this case, both the SIA and IIA returned feasible solutions. The LD-SDA coupled with the subproblem solver DICOPT was executed using the SIA and IIA feasible solutions. The direct solution of the problem using a Big-M reformulation of disjunctions and solver DICOPT was also tested for comparison purposes. Notation LD-SDA(SIA) and LD-SDA(IIA) is used to denote the LD-SDA solution obtained from the SIA and IIA initializations, respectively. The same notation is also used for solver DICOPT, i.e., DICOPT(SIA) and DICOPT(IIA). Table 7.1 shows the objective function and the computational time required by the SIA, the IIA, LD-SDA(SIA), LD-SDA(IIA), DICOPT(SIA) and DICOPT(IIA). Likewise, Figure 7.3 shows the Gantt charts corresponding to the SIA, IIA, LD-SDA(SIA) and LD-SDA(IIA) solutions.

Table 7.1. Comparison of initialization and solution strategies (case study 1)

	SIA initialization			IIA initialization		
	SIA	LD-SDA (SIA)	DICOPT (SIA)	IIA	LD-SDA (IIA)	DICOPT (IIA)
Profit objective $(-f)$ [$]	1479.94	1501.41	1501.41	1509.78	1511.26	1510.89
Fixed costs (c^{fix}) [$]	175.00	175.00	175.00	185.00	185.00	185.00
Variable costs (c^{var_1}) [$]	579.17	579.17	579.17	472.50	472.50	472.50
Dynamic costs (c^{var_2}) [$]	32.56	11.09	11.09	7.72	6.24	6.60
Material costs (c^{mat}) [$]	1083.33	1083.33	1083.33	975.00	975.00	975.00
Revenue (rev) [$]	3350.00	3350.00	3350.00	3150.00	3150.00	3150.00
CPU time [s]	161	2891	86400	456	5102	86400
Iterations	-	1	106	2	2	13

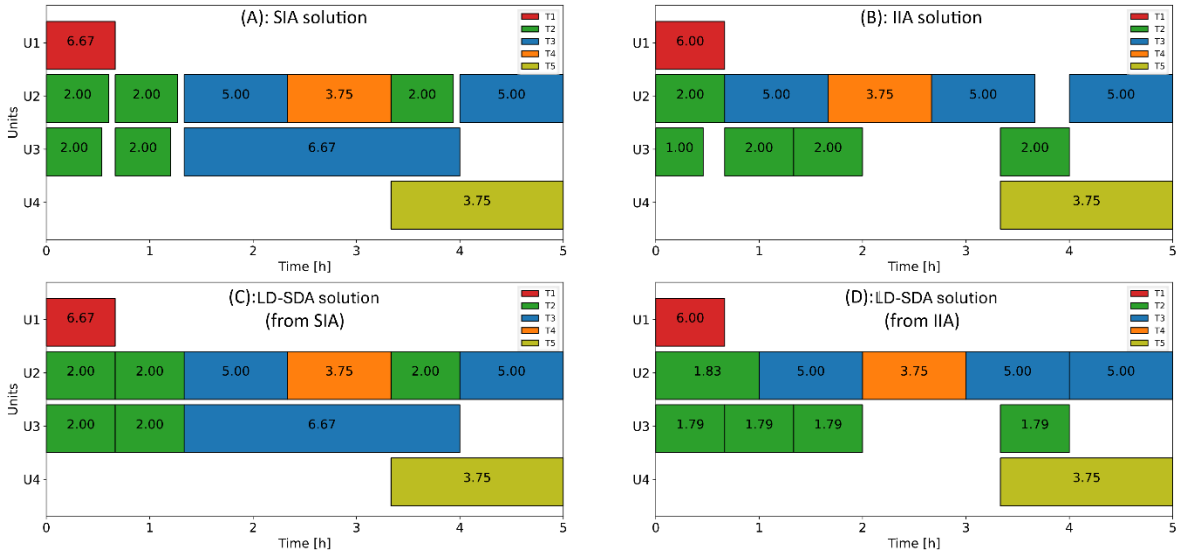


Figure 7.3. Gantt charts illustrating the SIA (A), IIA (B) and LD-SDA solutions (C and D) (case study 1). Colored blocks represent tasks. Batch sizes $b_{i,j,t}$ in m^3 appear as numbers inside colored blocks.

As shown in Table 7.1, the SIA initialization returned a profit of 1479.94$. This solution was used to initialize LD-SDA(SIA), which returned a solution with a profit of 1501.41$. The improvement in this case is attributed to a reduction in costs related to the operation of the semi-batch reactors. For instance, Figure 7.4 compares the control actions in the manipulated variables obtained from the SIA and the LD-SDA(SIA), which correspond to the second execution of $T2$ in $U3$ depicted in Figures 7.3A and 7.3C, i.e., the second green block of $T2$ in $U3$ from left to right in both figures. In the SIA solution, the flowrates of hot and cold utilities reached their upper bound in Figure 7.4A and remain at this value during half of the semi-batch task execution, to keep the temperature below its upper limit of 323.15 K (Figure 7.4E). This is in response to the exothermic reaction taking place and the flowrate of B in Figure 7.4C, which is dosed to the reactor during the first half of the operation. The LD-SDA(SIA) solution reduces costs by avoiding the use of the more costly hot heating fluid and using a different dosage strategy for component B, thus keeping the flowrate of the cold utility as low as possible (see Figures 7.4B and 7.4D). According to Figure 7.4F, the LD-SDA(SIA) solution was also able to satisfy the upper bound temperature constraint of 323.15 K, despite the reduction in operating costs generated by the changes in the manipulated variables.

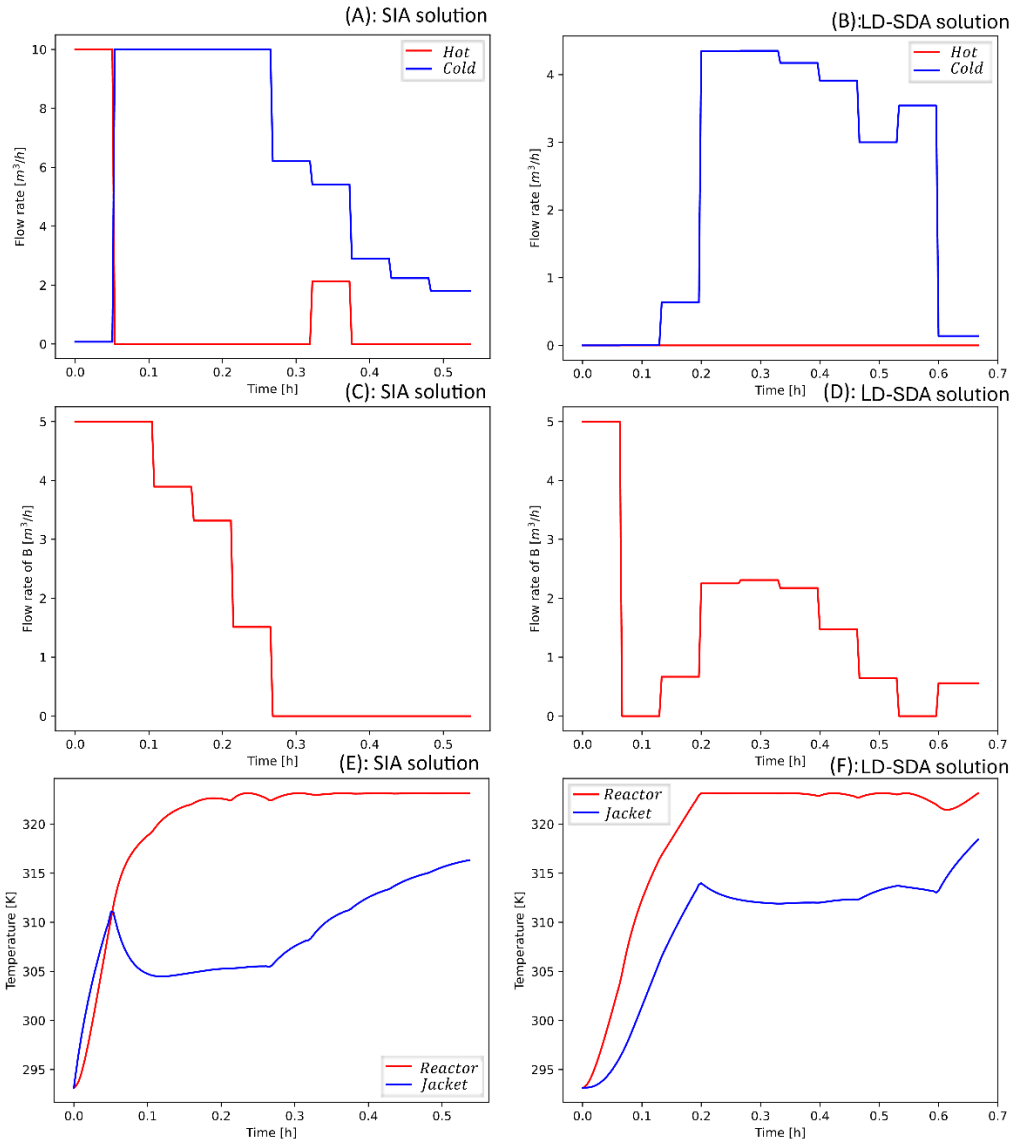


Figure 7.4. Manipulated variables and temperature profiles for the second execution of task $T2$ in unit $U3$. Subplots A, C and E: SIA solution. Subplots B, D and F: LD-SDA(SIA) solution.

The LD-SDA offers computational advantages, when compared to the direct solution of the problem using DICOPT. In the SIA case, solver DICOPT(SIA) was able to attain the solution found by the LD-SDA(SIA); however, DICOPT(SIA) reached the time limit of 86400 seconds, while the LD-SDA(SIA) stopped after 2891 seconds. This is because DICOPT(SIA) was unable to identify any feasible solution, other than that identified by the NLP subproblem solved in its first iteration. Therefore, DICOPT(SIA)

was unable to close its relative optimality gap beyond 7.7%. In contrast, the LD-SDA(SIA) was not only able to identify the solution returned by DICOPT(SIA), but also found other 14 feasible solutions within its neighborhood. Given that none of the neighbors identified by LD-SDA(SIA) improved the objective function, the LD-SDA(SIA) found that the ordered discrete decisions $n_{i,j}$ and $e_{i,j}$ identified by the SIA initialization are, in this case, locally optimal. Note that DICOPT was also executed using its default configuration, which starts by solving a relaxed-MINLP, instead of starting from a feasible point. Nonetheless, this strategy was unable to identify a feasible solution to the problem after reaching the maximum time limit.

The results obtained by the LD-SDA(IIA) in Table 7.1 are similar to those obtained by the LD-SDA(SIA), except that in this case the LD-SDA(IIA) was able to find a better combination of ordered discrete decisions after two iterations. DICOPT(IIA) was also unable to identify a feasible solution better than that identified after solving its first NLP subproblem with discrete variables fixed within the time limit imposed. The marginal improvement of 0.1% achieved by the LD-SDA(IIA) with respect to its initializations is attributed to the identification of a more economically attractive combination of external variables $e_{i,j}$, which allowed to identify better reaction times for reaction task $T2$ in unit $U2$ and $U3$. The optimal solution of $e_{i,j}$ returned by the IIA is $e_{T2,U2} = 2$ and $e_{T2,U3} = 2$, while the optimal solution found by the LD-SDA(IIA) is $e_{T2,U2} = 3$ and $e_{T2,U3} = 2$. As illustrated in Figures 7.3B and 7.3D, the LD-SDA(IIA) allowed to increase the processing time of the semi-batch reaction task $T2$ in both reactors $U2$ and $U3$, and redistributed the batch sizes used for these reactors, yielding a more profitable operation.

This case study shows the computational advantages of using an algorithm that implements a stopping criterion based on the neighborhood information of ordered discrete decisions. In addition, this shows how the proposed decomposition allows DICOPT to solve more manageable subproblems with complicating variables fixed, with respect to the direct solution of the problem using DICOPT. As shown in Table 7.1, the objective function of this problem is mainly affected by c^{fix}, c^{var_1}, c^{mat} and r, which are orders of magnitude larger than c^{var_2}. This is because process dynamics were only considered for two of the reaction tasks involved in the process. Nevertheless, the LD-SDA was still able to improve the economic objective function of the SSDO problem by improving the costs associated with the dynamic operation of the semi-batch reactors involved, and their corresponding processing times. A more complex case study involving a larger number of systems with a transient

operation resulting in higher dynamic operation costs, and therefore more interactions between scheduling and control decisions, is presented next.

7.3.2 Case study 2

The STN for this case study is show in Figure 7.5, which shows the different tasks I, units J, states M, task-unit pairs $IJ^{\times}$, and fractions $\rho_{i,m}^{+}$ and $\rho_{i,m}^{-}$ involved in this network. In this case, process dynamics are considered for the execution of reactions $R1$, $R2$ and $R3$, which can be performed in a large (Rl) or a small (Rs) reactor , i.e., $IJ^{\times dyn} = \{R1, R2, R3\} \times \{Rl, Rs\}$. Tables G.8-G.10 in appendix G (section G.3) provide the different scheduling and cost-related parameters for this problem. An additional scheduling constraint that is included in set SCH is shown in Eq. 7.9, which indicates that a minimum demand constraint must be satisfied at the end of the scheduling horizon for products $P1$ and $P2$. Also, a pure discrete-time representation is selected in this case, i.e., $\tau_{i,j,t}^{P} = \min\{\hat{\delta}\tau_{i,j}, \tau_{i,j}^{P,max}\}$ in Eq. 7.1H.

$$s_{m,n_T} \geq s_{m,n_T}^{demand}, \forall m \in \{P1, P2\}, s_{P1,n_T}^{demand} = 0.5\ m^3, s_{P2,n_T}^{demand} = 1\ m^3 \tag{7.9}$$

As shown in Figure 7.5, tasks $R1$, $R2$ and $R3$ represent batch operations. Task $R1$ represents reaction $B + C \rightarrow 2D$, with component B coming from state $M2$, and component C coming from state $M3$; task $R2$ represents reaction $A + D \rightarrow 2E$, with component A coming from state $I1$ and component D coming from state $I2$, and task $R3$ represents reaction $C + E \rightarrow F$, with component E coming from state $I3$. The differential-algebraic equations in DYN used to model these reactions are shown in Eqs. 7.10A-7.10F, where sets $Q_i, \forall i \in \{R1, R2, R3\}$ are introduced to denote the chemical species involved in each reaction, i.e., $Q_{R1} = \{B, C, D\}$, $Q_{R2} = \{A, D, E\}$ and $Q_{R3} = \{C, E, F\}$. $\boldsymbol{\chi}_{i,j}$ and $\boldsymbol{\nu}_i$ represent the concentrations and stochiometric coefficients of reaction executed in task i, respectively, e.g., $\boldsymbol{\chi}_{R1,j} = \left[\chi_{R1,j}^{q}\right]_{q \in Q_{R1}} = \left[\chi_{R1,j}^{B}, \chi_{R1,j}^{C}, \chi_{R1,j}^{D}\right]$ and $\boldsymbol{\nu}_{R1} = \left[\nu_{R1}^{q}\right]_{q \in Q_{R1}} = [\nu_{R1}^{B}, \nu_{R1}^{C}, \nu_{R1}^{D}] = [-1, -1, 2]$, where notation $[\cdot]_{q \in Q_i}$ is used as short notation for these vectors. Tables G.11 and G.12 in section G.3 of appendix G provide a description of the variables and parameters that appear in these equations. In addition to the assumptions considered for the first case study, this case study assumes that the recipe data defined by the control actions, batch sizes and processing time of a reaction remains unchanged every time a task is executed in a reactor, e.g., the recipe data of $R1$ may be different for Rl and Rs, but it remains unchanged every time a new batch of task $R1$ is processed. This helps to simplify the problem

by reducing the number of dynamic models needed. As a result, subscript t is dropped from Eq. 7.10A to Eq. 7.10F.

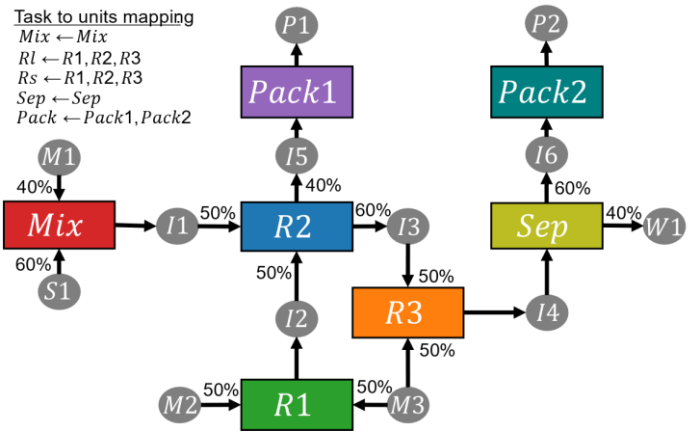


Figure 7.5. STN for case study 2. The execution of tasks $R1$, $R2$ and $R3$ in Rl and Rs is described by a semi-batch reactor model.

Reaction rates (7.10A)

$$\zeta_{i,j} = -k_i \exp\left(\frac{-\widetilde{Ea}_i}{\chi_{i,j}^{TR}}\right) \prod_{\substack{q \in Q_i \\ \nu_i^q < 0}} \left(\chi_{i,j}^{q}\right)^{-\nu_i^q}, \forall (i,j) \in IJ^{\times dyn}$$

Concentration balances (7.10B)

$$\dot{\boldsymbol{\chi}}_{i,j} = -\varphi_{i,j}^{P} \zeta_{i,j} \boldsymbol{\nu}_i, \boldsymbol{\chi}_{i,j}(\psi = 0) = \boldsymbol{C}_{i,0}, \forall (i,j) \in IJ^{\times dyn}$$

Energy balance (reactor temperature) (7.10C)

$$\dot{\chi}_{i,j}^{TR} = \varphi_{i,j}^{P} \left(\frac{Ua\left(\chi_{i,j}^{TJ} - \chi_{i,j}^{TR}\right)}{\varphi_{i,j}^{Vol} C_{P,i} \rho_R} + \frac{\Delta H_i \zeta_{i,j}}{C_{P,i} \rho_R} \right), \chi_{i,j}^{TR}(\psi = 0) = T_{R0}, \forall (i,j) \in IJ^{\times dyn}$$

Energy balance (jacket temperature) (7.10D)

$$\dot{\chi}_{i,j}^{TJ} = \varphi_{i,j}^{P} \left(\frac{\mu_{i,j}^{hot}\left(T_H - \chi_{i,j}^{TJ}\right) + \mu_{i,j}^{cold}\left(T_C - \chi_{i,j}^{TJ}\right)}{VJ_j} + \frac{Ua_j\left(\chi_{i,j}^{TR} - \chi_{i,j}^{TJ}\right)}{VJ_j \rho_J C_{PJ}} \right), \chi_{i,j}^{TJ}(\psi = 0) = T_{J0},$$

$$\forall (i,j) \in IJ^{\times dyn}$$

Maximum temperature constraint at the end (reactor cooling) (7.10E)

$$\chi_{i,j}^{TR}(\psi = 1) \leq T_R^{max}, \forall (i,j) \in IJ^{\times dyn}$$

Product concentration requirement (7.10F)

$$\boldsymbol{\chi}_{i,j}(\psi = 1) = \boldsymbol{C}_{i,DES}, \forall (i,j) \in IJ^{\times dyn}$$

The linking constraints that define $LINK$ in this case study ensure that if $x_{i,j,t} = 1$, then the reactive mixture volume $\varphi_{i,j}^{Vol}$ and the reaction time $\varphi_{i,j}^{P}$ agree with batch sizes $b_{i,j,t}$ and processing time $\tau_{i,j,t}^{P}$ for every task-unit pair $(i,j) \in IJ^{\times dyn}$ and every $t \in T$ (see Eqs 7.11A and 7.11B). Big-M terms are used to decide when equalities $b_{i,j,t} = \varphi_{i,j}^{Vol}$ and $\tau_{i,j,t}^{P} = \varphi_{i,j}^{P}$ in eqs. 7.11A and 7.11B are enforced. Note that these constraints also ensure that the same batch sizes and processing times are used every time reaction i is executed in reactor j. This case study considers a scheduling horizon η of 14 hours and a discretized time set with $n_T = 28$, resulting in a step size of $\hat{\delta} = 0.5$. Table G.13 in appendix G shows specific details about the reformulation and ordered discrete decisions of this case study.

$$-\mathrm{M}_{1,i,j}^{min}\left(1 - x_{i,j,t}\right) \leq b_{i,j,t} - \varphi_{i,j}^{Vol} \leq \mathrm{M}_{1,i,j}^{max}\left(1 - x_{i,j,t}\right), \forall (i,j) \in IJ^{\times dyn}, \forall t \in T \quad (7.11A)$$

$$-\mathrm{M}_{2,i,j}^{min}\left(1 - x_{i,j,t}\right) \leq \tau_{i,j,t}^{P} - \varphi_{i,j}^{P} \leq \mathrm{M}_{2,i,j}^{max}\left(1 - x_{i,j,t}\right), \forall (i,j) \in IJ^{\times dyn}, \forall t \in T \quad (7.11B)$$

The computational experiments performed for this case study are similar to those conducted for the first case study. The objective function and computational performance obtained using the LD-SDA and DICOPT from the SIA and IIA solutions are shown in Table 7.2. The Gantt charts of the SIA, IIA and LD-SDA solutions are shown in Figure 7.6. In this case, both the LD-SDA(SIA) and the LD-SDA(IIA) converged to the same solution, thus only three schedules are reported in Figure 7.6. Similarly, DICOPT(SIA) and DICOPT(IIA) converged to the same solution. Thus, I only refer to the LD-SDA and the DICOPT solutions through this section.

In contrast to the first case study, the dynamic costs c^{var_2} play a more important role in this case study. All the tasks performed in reactors Rl and Rs effect c^{var_2}, thus the optimal economic operation of these tasks becomes critical. Consequently, the SSDO solution returned by the LD-SDA substantially improved the SIA solution showing a 185% increase in profits (744.6$). For example, Figures 7.7A and 7.7B illustrate the differences between the concentration and cooling/heating utilities between the SIA solution and the LD-SDA solution for reaction $R2$ ($A + D \rightarrow 2E$) in reactor Rs. The SIA solution results in a shorter processing time (3 hours) at the expense of higher costs associated with higher heating and cooling requirements to accelerate the reaction rate at the beginning of the operation and to cool down the reactor at the end of the batch, respectively. On the other hand, the LD-SDA solution

increases the reaction time by half an hour to keep heating and cooling flows as low as possible, while still satisfying the product quality requirement at the end of the batch.

Table 7.2. Comparison of initialization and solution strategies (case study 2)

	SIA initialization			IIA initialization		
	SIA	LD-SDA (SIA)	DICOPT (SIA)	IIA	LD-SDA (IIA)	DICOPT (IIA)
Profit objective ($-f$) [$]	403.42	1148.09	1036.98	1001.18	1148.09	1036.98
Fixed costs (c^{fix}) [$]	470.00	470.00	470.00	470.00	470.00	470.00
Variable costs (c^{var_1}) [$]	391.67	381.94	381.94	391.67	381.94	381.94
Dynamic costs (c^{var_2}) [$]	859.92	212.46	323.57	362.15	212.46	323.57
Material costs (c^{mat}) [$]	1875.00	1631.94	1631.94	1775.00	1631.94	1631.94
Revenue (rev) [$]	4000.00	3844.44	3844.44	4000.00	3844.44	3844.44
CPU time [s]	54	4579	10757	73	4568	8827
Iterations	-	3	3	5	3	2

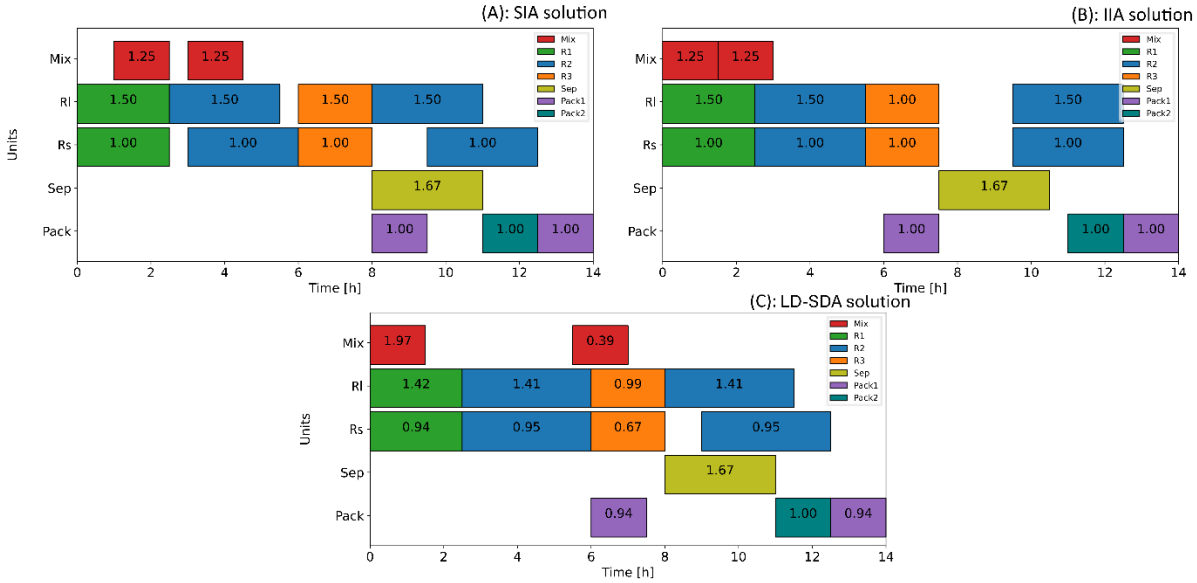


Figure 7.6. Gantt charts illustrating the SIA (A), IIA (B) and LD-SDA solutions (C) (case study 2). Colored blocks represent tasks. Batch sizes $b_{i,j,t}$ in m^3 appear as numbers inside colored blocks.

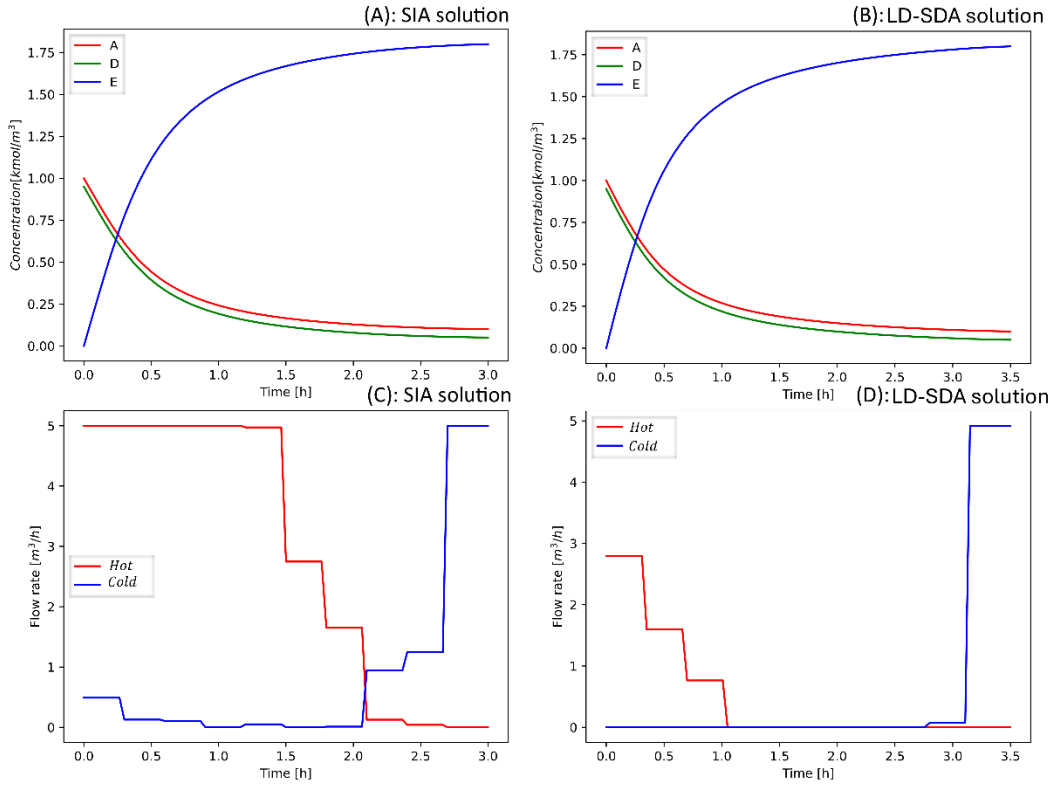


Figure 7.7. Concentration and manipulated variables profiles profiles when executing task $R2$ in unit Rs. A and C: SIA solution. B and D: LD-SDA solution.

Another difference in the solution obtained with respect to the results obtained for the first case study is that reductions in dynamic costs c^{var_2} are not the only factor improving the profit of the LD-SDA solution, when compared to the SIA and IIA solutions. The SIA solution reports the largest revenue of 4000$, which is explained by the fact that the reactors are operating at their maximum capacity (see Figure 7.6A). To maximize production, the material and variable cost of other tasks is also pushed towards their maximum in the SIA solution, i.e., $c^{mat} = 1875\$$ and $c^{var_1} = 391.67\$$. The outcome is high revenue at the expense of a relatively costly operation. The IIA solution shows a similar behavior, except that it recognizes that one of the reaction tasks (i.e., $R3$ in Rl) does not need to operate at its maximum capacity to achieve the same revenue of 4000$, thus reducing material costs (see Figures 7.6A and 7.6B). In contrast, the LD-SDA solution illustrates that the plant does need to operate at its maximum capacity to maximize profits. This results in a locally optimal solution showing a tradeoff between revenue and costs, where reactors are not set to operate at their maximum capacity (see Figures

7.6C). As an example, variable processing times show a tradeoff between profit and costs. When analyzing the minimization of c^{var_2} for each of the reaction tasks involved and ignoring scheduling constraints, it was found that a value of $c^{var_2} = 0$ can be attained if reaction time is long enough to avoid the need of heating or cooling utilities to promote the reactions. Alternatively, processing times should be as short as possible from the scheduling perspective, to maximize the number of times each reaction is performed within the scheduling horizon and increase profits. Feasibility considerations also play an important role, given that the time needed for a reaction task must be within a finite scheduling horizon, but also needs to be long enough to satisfy product quality requirements. These interactions are recognized by the LD-SDA but ignored by the SIA or the IIA, which are considered sequential strategies.

When comparing the performance of the LD-SDA and DICOPT, it was found that the solution reported by DICOPT with a profit of 1036.98$ corresponds to the solution obtained from the first subproblem solved by the LD-SDA. This suggests that DICOPT was unable to identify better combinations of ordered discrete decisions, other than those used to initialize its search. The solver DICOPT has two stopping criteria available: 1) stop when the gap between its MIP master problem and the NLP subproblem closes, and 2) stop when the NLP subproblem solution stops improving[369]. Both stopping criteria were tested in this case, but none of them resulted in a better objective function value. This shows that the superiority of the LD-SDA in comparison to a conventional MINLP solver like DICOPT extends beyond just superstructure design problems. The benefits provided by the LD-SDA encompass other types of problems, including the SSDO problems examined in this study.

The attractive features and innovation introduced by the LD-SDA is attributed to its theoretical discrete convex analysis foundations. While a general-purpose MINLP solver such as DICOPT relies on the convexity of a relaxed-MINLP to guarantee global optimality, i.e., discrete variables are allowed to vary continuously within their bounds, the theory of discrete convex analysis generalizes this notion and acknowledges that the MINLP relaxation is not the only continuous relaxation of the original problem. Alternative relaxations such as the local convex extension are also available, and this relaxation may be convex even if the relaxed-MINLP relaxation is not convex. The local convex extension can be geometrically interpreted as an interpolation of the subproblem objective function f^* within the search space of ordered discrete decisions $n_{i,j}$ and $e_{i,j}$, e.g., in the one dimensional case, the local convex extension is obtained by connecting the subproblem objective function for fixed values of the ordered variable using lines[6]. If the N_∞ neighborhood is used, the search strategy and stopping

criterion used by the LD-SDA guarantee global optimality when the local convex extension is convex, independently of the convexity properties of the relaxed-MINLP[4]. Consequently, the LD-SDA strategy may lead to better quality solutions than DICOPT, as demonstrated through this work.

7.3.3 Modelling and computational performance analysis

Overall, the proposed scheduling model augmented with dynamic optimization models (i.e., Eq. 7.3A-7.3D) includes linking constraints that allow the scheduling and control layers to interact with each other, e.g., see Eqs. 7.11A and 7.11B, which link processing times and batch sizes between the scheduling and dynamic optimization formulations. Accordingly, the proposed LD-SDA solution framework simultaneously optimizes scheduling and dynamic optimization variables. From an optimization point of view, this strategy avoids discrepancies between the scheduling and dynamic optimization models by guaranteeing feasibility and local optimality. From a practical point of view, the LD-SDA solution returns the optimal operation recipe data for the scheduling and dynamic operation of the batch plant, under the assumption that the dynamic models are an accurate representation of the actual system. Note that it is not possible to know how close to global optimality the solutions obtained are given that a standard global solver such as BARON would fail even to identify feasible solutions for the case studies discussed. Furthermore, the discrete-convex properties needed to verify global optimality (e.g., integral convexity) of the case studies discussed would require a complete enumeration of external variables and the solution of their corresponding subproblems. This would result in ~10^7 and ~10^{15} MINLP subproblems for case studies 1 and 2, respectively. Hence, it is an overly computationally challenging and demanding task to assess the global optimality of the solutions obtained with the discrete convex analysis tools available, given that there is no global solver based on this theory available in the literature.

The case studies considered in this work used first-principles dynamic optimization models, including both the mass and energy balances of the reactors involved. Data-driven models, linearization or model reduction techniques to simplify the dynamic optimization part of the problem were not considered in this work; however, the general formulation shown in Eqs. 7.3A-7.3D is flexible to incorporate other modelling strategies depending on the user's needs. This would allow to simplify and facilitate the solution of subproblems; nonetheless, these approximated modeling techniques pose additional challenges. For instance, the direct linearization of process dynamics around an operating point would not result in an accurate representation of nonlinear batch process dynamics, e.g., linearize

around the initial state of a batch reactor with a concentration of target products close to 0% may not accurately represent the final state of the batch with a target product concentration closer to 100%. Thus, further research would be needed to incorporate alternative linearization schemes for batch processes within the proposed framework, e.g., see the adaptative model predictive control approach with successive linearization proposed by Mendis et al.[370].

Based on the results presented, the complexity of the SSDO problems studied does not come from the specific process dynamic nor the scheduling formulations, but from their integration. The sequential solution of scheduling and dynamic optimization problems corresponds to the SIA column in Tables 7.1 and 7.2. For example, the sequential optimization of scheduling and dynamic optimization for case study 1 required 161 seconds only, while the solution of the SSDO problem using the LD-SDA(SIA) required 2891 seconds, i.e., a relative difference of 94% with respect to the LD-SDA(SIA) time. Similarly, the computational time of the LD-SDA(SIA) was two orders of magnitude larger than the SIA computational time. These differences are attributed to a variety of factors. For example, the need of including disjunctions to enable the modeling of variable processing times in a discrete-time SSDO formulation; the tradeoffs added to the problem when simultaneously considering scheduling and control decisions, e.g., the need of longer processing times in case study 2 to benefit dynamic costs at the expense of a lower revenue; and the need of finding optimal batch sizes and processing times, whose interactions are often neglected in purely scheduling problems that do not incorporate process dynamics.

Note that the average solution time needed to solve both SSDO problems with the LD-SDA (including the initialization time) was 95% and 53% shorter than the time needed by MINLP solver DICOPT for case studies 1 and 2, respectively. Despite this performance improvement, additional research is required to further decrease the computational costs, particularly if implementing the LD-SDA in larger industrial-scale applications, which often incorporate more units in the batch plant than those considered in this work. For instance, case studies 1 and 2 in this work include 4 and 5 units, respectively. In contrast, the generic industrial-scale batch production process proposed by Blömer and Günther[371] includes 9 units, while the industrial scale sugarhouse process studied by Durand and Bandoni[372] includes 28 units. As future work, alternative decomposition strategies can be explored and assessed when addressing these larger-scale problems, e.g., evaluating the effect of moving/removing variables from the master problem, studying the effect of alternative neighbors or variable size neighbors for the exploration, evaluating the performance of nested decomposition strategies, etc.

Varying the length of the scheduling horizon may also result in more challenging problems involving more discrete scheduling variables and larger search spaces for ordered discrete decisions. Scheduling problems typically consider scheduling horizons of hours or days; however, medium and long-term planning models sometimes need to be simultaneously optimized with process scheduling, which results in much larger horizons of days-years[363]. The solution of scheduling problems with medium and long-term scheduling horizons has been previously studied using rolling horizon approaches (e.g., Verderame and Floudas[373,374]). As part of the future work, those strategies could be coupled with the LD-SDA to enable medium and long-term simultaneous planning, scheduling and dynamic optimization.

Emphasis was placed on the importance of variable processing times, demonstrating that they exhibit the most substantial impact on profit increase for both case studies. Note that processing times were only considered for reaction tasks in this study; however, these can be incorporated into other relevant unit operations, e.g., filtration and separation. For instance, variable processing times play a key role in the operation of industrial-size reactive-semibatch distillation operations, as demonstrated in the work by Li et al.[375] where an endothermic transesterification of two esters and two alcohols takes place in the reboiler. That study showed the first experimental study of batch distillation processes optimization, demonstrating that simultaneously optimizing processing times with key operating variables such as reflux ratio profiles and feed flow rate resulted in up to 30% operation time savings and reductions in major costs, e.g., heat consumption and working time of the operators. Based on this, the consideration of standard and intensified separation operations within the proposed SSDO solution framework would result in substantial cost savings attributed to shorter operational times.

Chapter 8

Multicut logic-based Benders decomposition for discrete-time scheduling and dynamic optimization of network batch plants

The main objective of this chapter is to extend the LD-BD strategy to solve SSDO problems using a discrete-time STN model coupled with dynamic models. From the solution algorithm perspective, the solution strategy that has been predominantly used to solve problems like the herein considered SSDO problem is GBD. Chu and You used GBD to solve SSDO problems with state task network (STN) structures[61], and then extended their framework to address SSDO problems under uncertainty[355]. Nie et al. applied GBD to solve SSDO problems with resource task network (RTN) structures[62], and studied an application in block copolymerization[366]. More recently, Ji and Gu introduced a hybrid deterministic-stochastic GBD strategy for the SSDO of state equipment networks (SEN) where the master problem was solved using a genetic algorithm[60], and then showed an application of SSDO for the no-wait batch process[356]. Mitrai and Daoutidis extended the GBD framework to consider integrated planning, scheduling, and control problems of single-stage continuous processes[48]. Thus, the application of a LBBD technique such as LD-BD for the SSDO problem herein considered has not been discussed in the existing literature.

LD-BD was proposed in chapter 4 for superstructure optimization, but its underlying principles can be generalized and extended to SSDO. However, there are some challenges that must be addressed to enable the application of LD-BD in SSDO. First, the superstructure optimization problems that have been successfully solved with LD-BD thus far resulted in NLP subproblems; however, subproblems in SSDO are MINLPs because there are some discrete scheduling variables that do not follow an ordered discrete structure and cannot be part of the master problem, e.g., sequencing variables in discrete-time STNs. Second, the size of the search spaces and the number of ordered discrete decisions in multi-task multi-unit SSDO problems is large, compared to the superstructure-based design of a single processing unit. The first challenge, i.e., solution of MINLP subproblems, is addressed in this chapter through the implementation of a pruning strategy that saves computational time by avoiding the solution of MINLP subproblems that are not expected to improve the objective function. Similar pruning approaches have been implemented in the past and have shown to significantly reduce the CPU time required to solve simultaneous design and control problems[20]; nevertheless, these strategies have not been studied in the context of SSDO. The second challenge is addressed by generating multiple cuts instead of a single cut

per iteration with the purpose of providing more information to the master problem at each iteration. Multicut generation strategies have been successfully applied in the past to increase the convergence speed of GBD, but these have not been considered yet within the LD-BD algorithm[48]. Based on the above, this work proposes a Multicut LD-BD (MLD-BD) local optimization method for SSDO problems that incorporates distinctive features aimed at finding local solutions and reducing the computational costs.

This chapter is organized as follows. Section 8.1 summarizes the problem under consideration. Section 8.2 explains the proposed MLD-BD algorithm. In Section 8.3, the performance of the proposed MLD-BD strategy is tested using two case studies. A comparison of the proposed framework with respect to sequential scheduling and dynamic optimization solutions and GBD strategies is also shown in this section.

8.1 Problem statement

This work considers a problem in the form of Eq. (8.1A), which is a common formulation in top-down SSDO studies for systems with network structures, e.g., Chu and You[61], Nie et al.[62] and Ji and Gu[356]. The goal of problem (8.1A) is to enable the interaction between scheduling variables in vector $\boldsymbol{v}^{sch}$ and dynamic variables in vector $\boldsymbol{v}^{dyn}$ to simultaneously identify schedules and open-loop control actions (often found from dynamic optimization) that can contribute to the optimization of an economic objective function f, e.g., cost minimization. Scheduling variables $\boldsymbol{v}^{sch}$ include the assignment, sequencing and timing of the execution of tasks within the scheduling horizon, and the amount of material needed when a task is executed. Dynamic variables are those used to predict the transient performance of an operation, e.g., state and control action profiles. Constraints are categorized as scheduling constraints (set SCH), dynamic constraints (set DYN) and linking constraints (set $LINK$). Scheduling constraints are those that only involve scheduling variables $\boldsymbol{v}^{sch}$ only, e.g., a processing unit can only execute one task at a time, precedence constraints between tasks, etc. Dynamic constraints are those that only involve dynamic variables $\boldsymbol{v}^{dyn}$ only, e.g., differential equations used to represent the transient behavior of a process. Linking constraints are the most general group, as they involve both variables $\boldsymbol{v}^{sch}$ and $\boldsymbol{v}^{dyn}$, e.g., the processing time of a task considered by the scheduling model must match with the execution time considered in the differential equations that represent this same task. It is assumed that the objective function in problem (8.1A) is bounded, and that problem (8.1A) is feasible.

$$\min_{s.t.} f\left(\boldsymbol{v}^{sch}, \boldsymbol{v}^{dyn}\right) \tag{8.1A}$$
$$\boldsymbol{v}^{sch} \in SCH, \boldsymbol{v}^{dyn} \in DYN, [\boldsymbol{v}^{sch}, \boldsymbol{v}^{dyn}] \in LINK$$

$$\boldsymbol{v}^{sch} = [\boldsymbol{y}, \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}] \tag{8.1B}$$

Although the proposed MLD-BD method is particularly appealing for the discrete-time SSDO formulation discussed in section 8.1.1, the MLD-BD proposed in this work aims to solve problems with the more general formulation shown in problem (8.1A), by relying on the variable partitioning of scheduling variables shown in equation (8.1B). Without loss of generality, the vector of scheduling variables can be partitioned as $\boldsymbol{v}^{sch} = [\boldsymbol{y}, \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}]$, where $\boldsymbol{y}$ represents the scheduling Boolean/binary variables that poses ordered structures (e.g., indexed binary variables defined over a discrete-time grid that are enforced to take the value of 1 to indicate the end of the scheduling horizon and 0 elsewhere); $\boldsymbol{y}^{sch}$ are the remaining Boolean/binary variables of the problem without ordered properties (e.g., decide between turning on/off a processing unit); $\boldsymbol{n}$ represents the integer variables in the scheduling formulation (e.g., a batching variable that indicates the number of times a task need to be executed), and $\boldsymbol{x}^{sch}$ are the continuous scheduling variables (e.g., volume of material processed in a task). A formal definition of ordered Boolean variables in $\boldsymbol{y}$ that allows to distinguish between $\boldsymbol{y}$ and $\boldsymbol{y}^{sch}$ is provided in section H.1 of appendix H.

8.1.1 The discrete-time STN model with variable processing times for SSDO

This study considers the discrete-time STN model with dynamic and linking constraints introduced in chapter 7 and is summarized in this section for completeness. This model enables the modeling of variable processing times to test the performance of the MLD-BD methodology. Note that the discrete-time STN representation has been chosen over a continuous-time representation because SSDO solutions with discrete-time formulations have not been widely studied in the literature. To the authors' knowledge, this is attributed to the combinatorial complexity added to discrete-time scheduling formulations when variable processing times are introduced. This complexity is compounded by the nonlinear relationships between these processing times and the remaining dynamic variables of the formulation. One way to overcome the aforementioned issues is to take advantage of the ordered discrete decisions (i.e., integer variables $\boldsymbol{n}$ and Boolean variables with ordered structures $\boldsymbol{y}$) that inherently appear in Eqs. (8.2A)-(8.2M), as discussed below. These variables, also referred to as

complicating variables, are optimized by MLD-BD in a master problem. While integer variables $\boldsymbol{n}$ can be readily decomposed and optimized in the MLD-BD master problem, ordered Boolean variables $\boldsymbol{y}$ need to be reformulated (see section 8.1.2).

I introduced some basic notation first: the main sets in the formulation are the set of discretized scheduling time points T with index t, the set of states M with index m that includes the different input, intermediate and final product storage facilities, and the set of task-unit pairs $IJ^{\times}$ with indices (i,j) containing the different combinations of tasks i that can be performed in units j, e.g., a reaction task can be performed in 3 available reactors. Set $IJ^{\times dyn}$ is a subset of $IJ^{\times}$ with the pairs (i,j) with dynamic models in the formulation. Other sets in the formulation include those with the available discrete processing times $\tau_{i,j}$ needed for the execution of pair (i,j), and denoted by $\mathrm{T}_{i,j}, \forall (i,j) \in IJ^{\times}$.

The ordered Boolean variables $\boldsymbol{y}$ of this problem are $Y_{\tau_{i,j},i,j}, \forall (i,j) \in IJ^{\times}, \forall \tau_{i,j} \in \mathrm{T}_{i,j}$ that take the value of $True$ when processing time $\tau_{i,j}$ is selected for the execution of task i in unit j. The integer variables $\boldsymbol{n}$ of the problem are batching-assignment variables $n_{i,j}, \forall (i,j) \in IJ^{\times}$ that represent the total number of times task i is executed in unit j within the scheduling horizon. As discussed by Velez and Maravelias[361], batching variables can vary from 0 to $\left\lfloor n_T / \left\lceil \tau_{i,j}^{P,min} / \hat{\delta} \right\rceil \right\rfloor$, where $n_T + 1$ is the number of time-slots in the scheduling time-grid and $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ represent the ceil and floor operators. The formulation also includes additional scheduling binary variables $x_{i,j,t}, \forall (i,j) \in IJ^{\times}, \forall t \in T$ and $x_{i,j,t}^{sum}, \forall (i,j) \in IJ^{\times}, \forall t \in T$ that appear in $\boldsymbol{y}^{sch}$. When $x_{i,j,t}$ takes the value of 1, it means that the execution of task i in unit j starts at time t. Auxiliary variables $x_{i,j,t}^{sum}$ are included to model variable processing times, and they are a function of independent binary variables $x_{i,j,t}$. Similarly, continuous scheduling variables $\boldsymbol{x}^{sch}$ include independent variables $b_{i,j,t}, \forall (i,j) \in IJ^{\times}, \forall t \in T$ and auxiliary variables $b_{i,j,t}^{shift}, \forall (i,j) \in IJ^{\times}, \forall t \in T \backslash \{0\}$. Continuous variable $b_{i,j,t}$ denotes the amount of material that is processed when executing task i in unit j starting at time t, while $b_{i,j,t}^{shift}$ is a variable introduced with the sole purpose of enable the modeling of variable processing times. Other variables that are included in $\boldsymbol{x}^{sch}$ are the inventory of state m at time t ($s_{m,t}, \forall m \in M, \forall t \in T$), and the continuous processing time of task i in unit j when executed at time t ($\tau_{i,j,t}^{P}, \forall (i,j) \in IJ^{\times}, \forall t \in T$). The vector of dynamic variables $\boldsymbol{v}^{dyn}$ includes a vector of manipulated variables or inputs $\boldsymbol{\mu}_{i,j,t}$ (e.g., flow of utility streams); the time-dependent vectors of state variables $\boldsymbol{\chi}_{i,j,t}$ (e.g., concentrations, temperatures, etc.);

the time-dependent vector of algebraic variables $\boldsymbol{\zeta}_{i,j,t}$ (e.g., enthalpies, reaction rates, etc.), and a vector of time-independent continuous variables $\boldsymbol{\varphi}_{i,j,t}$ (e.g., total hold-up $\varphi_{i,j,t}^{Vol}$, total task duration $\varphi_{i,j,t}^{P}$, etc). The constraints used in this work to formulate the SSDO problem are shown in Eqs. (8.2A)-(8.2M).

$$b_j^{min} x_{i,j,t} \le b_{i,j,t} \le b_j^{max} x_{i,j,t}, \ \forall (i,j) \in IJ^{\times}, \forall t \in T \tag{8.2A}$$

$$\sum_{\substack{i \in I \\ I_{i,j}^{\times}=1}} x_{i,j,t}^{sum} \le 1, \ \forall j \in J, \forall t \in T \tag{8.2B}$$

$$s_{m,0} = s_m^{init} - \sum_{\substack{i \in I \\ I_{i,m}^{-}=1}} \rho_{i,m}^{-} \sum_{\substack{j \in J \\ I_{i,j}^{\times}=1}} b_{i,j,0}, \forall m \in M \tag{8.2C}$$

$$s_{m,t} = s_{m,t-1} + \sum_{\substack{i \in I \\ I_{i,m}^{+}=1}} \rho_{i,m}^{+} \sum_{\substack{j \in J \\ I_{i,j}^{\times}=1}} b_{i,j,t}^{shift} - \sum_{\substack{i \in I \\ I_{i,m}^{-}=1}} \rho_{i,m}^{-} \sum_{\substack{j \in J \\ I_{i,j}^{\times}=1}} b_{i,j,t}, \tag{8.2D}$$

$$\forall m \in M, \forall t \in T \backslash \{0\}$$

$$s_{m,t} \le s_m^{max}, \forall m \in M, \forall t \in T \tag{8.2E}$$

$$\sum_{t \in T} x_{i,j,t} = n_{i,j}, \forall (i,j) \in IJ^{\times} \tag{8.2F}$$

$$\tau_{i,j}^{P,min} x_{i,j,t} \le \tau_{i,j,t}^{P}, \ \forall (i,j) \in IJ^{\times}, \forall t \in T \tag{8.2G}$$

$$\underline{\tau}_{i,j} \in \mathrm{T}_{i,j} \left[\begin{array}{c} Y_{\tau_{i,j},i,j} \\ \tau_{i,j,t}^{P} \leqq \min\{\hat{\delta}\tau_{i,j}, \tau_{i,j}^{P,max}\}, \quad \forall t \in T \\ x_{i,j,t}^{sum} = \sum_{\substack{t' \in T \\ t-\tau_{i,j}+1 \le t' \le t}} x_{i,j,t'}, \forall t \in T \\ b_{i,j,t}^{shift} = \begin{cases} b_{i,j,t-\tau_{i,j}}, & t - \tau_{i,j} \ge 0 \\ 0, & t - \tau_{i,j} < 0 \end{cases}, \forall t \in T \backslash \{0\} \end{array} \right], \forall (i,j) \in IJ^{\times}, \tag{8.2H}$$

$$\mathrm{T}_{i,j} = \left\{ \left\lceil \tau_{i,j}^{P,min}/\hat{\delta} \right\rceil, \left\lceil \tau_{i,j}^{P,min}/\hat{\delta} \right\rceil + 1, \ldots, \left\lceil \tau_{i,j}^{P,max}/\hat{\delta} \right\rceil \right\}, \forall (i,j) \in IJ^{\times}$$

$$\dot{\boldsymbol{\chi}}_{i,j,t} = \varphi_{i,j,t}^{P} \widehat{\boldsymbol{h}}_{i,j,t}\left(\boldsymbol{\chi}_{i,j,t}, \boldsymbol{\zeta}_{i,j,t}, \boldsymbol{\mu}_{i,j,t}, \boldsymbol{\varphi}_{i,j,t}\right), \forall (i,j) \in IJ^{\times dyn}, \forall t \in T \tag{8.2I}$$

$$\boldsymbol{\chi}_{i,j,t}(\psi = 0) = \boldsymbol{\chi}^{0}_{i,j,t}, \forall (i,j) \in IJ^{\times dyn}, \forall t \in T \tag{8.2J}$$

$$\boldsymbol{h}_{i,j,t}\left(\boldsymbol{\chi}_{i,j,t}, \boldsymbol{\zeta}_{i,j,t}, \boldsymbol{\mu}_{i,j,t}, \boldsymbol{\varphi}_{i,j,t}\right) \leq \mathbf{0}, \forall (i,j) \in IJ^{\times dyn}, \forall t \in T \tag{8.2K}$$

$$-\mathrm{M}\left(1 - x_{i,j,t}\right) \leq \tau^{P}_{i,j,t} - \varphi^{P}_{i,j,t} \leq \mathrm{M}\left(1 - x_{i,j,t}\right), \forall (i,j) \in IJ^{\times dyn}, \forall t \in T \tag{8.2L}$$

$$-\mathrm{M}\left(1 - x_{i,j,t}\right) \leq b_{i,j,t} - \varphi^{Vol}_{i,j,t} \leq \mathrm{M}\left(1 - x_{i,j,t}\right), \forall (i,j) \in IJ^{\times dyn}, \forall t \in T \tag{8.2M}$$

The constraints include those scheduling constraints in SCH that are typically included in network batch scheduling problems, shown in Eqs. (8.2A)-(8.2F). Unit capacity constraints (8.2A) enforce upper and lower bounds over $b_{i,j,t}$ when $x_{i,j,t} = 1$. Unit utilization constraints (8.2B) guarantee that units can only be executing one task at a time. Material balances in Eqs. (8.2C) and (8.2D) enable the flow of material between the different units and states (e.g., storage tanks) in the process. Maximum storage constraints (8.2E) define the capacity of the different states. Batching constraints (8.2F) guarantee that batching-assignment variables $n_{i,j}$ define the number of times tasks are executed in the available units, while variables $x_{i,j,t}$ stablish the sequencing and timing of these tasks within the scheduling horizon. Furthermore, additional constraints (8.2G) and (8.2H) are included in SCH to enable variable processing times, i.e., if $Y_{\tau_{i,j},i,j} = True$, then the continuous processing time $\tau^{P}_{i,j,t}$ of a task-unit pair (i,j) is allowed to vary between lower and upper bounds $\tau^{P,min}_{i,j}$ and $\min\{\hat{\delta}\tau_{i,j}, \tau^{P,max}_{i,j}\}$, where δ is the length of scheduling time slots. Dynamic constraints that define DYN include the differential Eqs. (8.2I), initial conditions (8.2J), and algebraic equations/inequalities (8.2K) used to model the tasks involved, e.g., differential equations that model a semi-batch reactor or a batch distillation column. Linking constraints in $LINK$ are added to guarantee that those variables in $\boldsymbol{v}^{dyn}$ agree with their counterparts in $\boldsymbol{v}^{sch}$. These are specific to each case study, but they generally include constraints to guarantee that the processing times and batch sizes used in $\boldsymbol{v}^{sch}$ (i.e., $\tau^{P}_{i,j,t}$ and $b_{i,j,t}$) and $\boldsymbol{v}^{dyn}$ (i.e., $\varphi^{P}_{i,j,t}$ and $\varphi^{Vol}_{i,j,t}$) are equal, through constraints (8.2L) and (8.2M). Note that a reformulation of ordered Boolean variables $Y_{\tau_{i,j},i,j}, \forall (i,j) \in IJ^{\times}, \forall \tau_{i,j} \in \mathrm{T}_{i,j}$ is still needed prior solving the problem with MLD-BD. This reformulation is explained in the following section.

8.1.2 Problem reformulation

The key idea behind MLD-BD is to improve the search strategy for systems that involve ordered discrete decisions $[\boldsymbol{n}, \boldsymbol{y}]$, by considering them as complicating variables. To do that, $[\boldsymbol{n}, \boldsymbol{y}]$ need to be reformulated first as $[\boldsymbol{n}, \boldsymbol{e}]$, where $\boldsymbol{e}$ is a vector of integer variables referred to as external variables. This reformulation is needed since an exhaustive enumeration of Boolean variables in $\boldsymbol{y}$ would be performed if they are passed directly to MLD-BD thus resulting in overly expensive calculations. Instead, the MLD-BD framework can efficiently optimize integer variables $\boldsymbol{e}$, by considering them as complicating variables. Based on the different reformulations presented through this thesis, a framework that generalizes reformulation $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{e})$ and their requirements for a general class of chemical engineering applications that exhibit naturally ordered discrete sets is presented in section H.1 of appendix H. In summary, the key requirements that must be satisfied by vector $\boldsymbol{y}$ in a general MINLP/GDP optimization problem to perform the reformulation are: i) $\boldsymbol{y}$ can be partitioned into one or more groups of binary/Boolean variables defined over ordered sets (R1); ii) these ordered sets must contain ordinal, discrete interval, or discrete ratio data, as defined by Kaptein et al.[376] (R2); and iii) partitioning constraints (e.g., exactly one of the Boolean variables in each group is $True$) must be enforced over each group of binary/Boolean variables defined over ordered sets (R3). If these conditions are met, the resulting reformulation results in a vector of reformulated complicating integer variables $[\boldsymbol{n}, \boldsymbol{e}]$ that can be efficiently explored by combining discrete convex analysis and LBBD principles (see section 8.2). This means that MLD-BD systematically explores combinations of variables $[\boldsymbol{n}, \boldsymbol{e}]$, while variables in $\boldsymbol{y}$ are expressed as a function of $\boldsymbol{e}$ (i.e., $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{e})$), instead of explicitly searching for the optimal solution of $\boldsymbol{y}$. Upper and lower bounds imposed over variables $\boldsymbol{n}$ and $\boldsymbol{e}$ are denoted by search regions N and E, respectively. For the particular class of problems addressed in this work, i.e., the SSDO formulation in section 8.1.1, vector $\boldsymbol{y}$ contains Boolean variables $Y_{\tau_{i,j},i,j}, \forall (i,j) \in IJ^{\times}, \forall \tau_{i,j} \in \mathrm{T}_{i,j}$ that are defined over indices $\tau_{i,j} \in \mathrm{T}_{i,j}$ that satisfy the reformulation requirements R1-R3, as summarized below:

R1: Sets $\mathrm{T}_{i,j}, \forall (i,j) \in IJ^{\times}$ are non-empty finite subsets of $\mathbb{Z}$, where the elements of $\mathrm{T}_{i,j} = \left\{\left\lceil \tau_{i,j}^{P,min}/\hat{\delta} \right\rceil, \left\lceil \tau_{i,j}^{P,min}/\hat{\delta} \right\rceil + 1, \ldots, \left\lceil \tau_{i,j}^{P,max}/\hat{\delta} \right\rceil\right\}$ are consecutive numbers in ascending order.

R2: Sets $\mathrm{T}_{i,j}, \forall (i,j) \in IJ^{\times}$ contain ordinal data, which in this case corresponds to integer numbers representing the number of discrete points in the scheduling time-grid needed to execute task unit pairs $(i,j) \in IJ^{\times}$.

R3: As shown in equation (8.2H), constraints of the form "exactly 1 variable from $Y_{\tau_{i,j},i,j}, \forall \tau_{i,j} \in \mathrm{T}_{i,j}$ is $True$" are defined for every pair $(i,j) \in IJ^{\times}$.

Therefore, reformulation $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{e})$ can be applied to the Boolean variables that appear in equation (8.2H), resulting in the vector of external variables and reformulation, respectively, i.e.,

$$\boldsymbol{e} \coloneqq e_{i,j} \in \mathrm{T}_{i,j} = \left\{\left\lceil \tau_{i,j}^{P,min}/\hat{\delta} \right\rceil, \left\lceil \tau_{i,j}^{P,min}/\hat{\delta} \right\rceil + 1, \dots, \left\lceil \tau_{i,j}^{P,max}/\hat{\delta} \right\rceil\right\}, \forall (i,j) \in IJ^{\times} \tag{8.3A}$$

$$\boldsymbol{y}(\boldsymbol{e}) \coloneqq Y_{\tau_{i,j},i,j} = \begin{cases} True, \tau_{i,j} = e_{i,j} \\ False, \tau_{i,j} \neq e_{i,j} \end{cases}, \forall (i,j) \in IJ^{\times}, \forall \tau_{i,j} \in \mathrm{T}_{i,j} \tag{8.3B}$$

Eq. (8.3A) defines external variables $\boldsymbol{e}$ for the SSDO problem, which in this case are integer variables $e_{i,j}, \forall (i,j) \in IJ^{\times}$ that vary between lower bounds $\left\lceil \tau_{i,j}^{P,min}/\hat{\delta} \right\rceil$ and upper bounds $\left\lceil \tau_{i,j}^{P,max}/\hat{\delta} \right\rceil$, i.e., take a pair $(i,j) \in IJ^{\times}$, then external variable $e_{i,j}$ indicates the number of time points in the STN time grid needed for the execution of task i in unit j. Eq. (8.3B) calculates $Y_{\tau_{i,j},i,j}, \forall (i,j) \in IJ^{\times}, \forall \tau_{i,j} \in \mathrm{T}_{i,j}$ as a function of $e_{i,j}$. For instance, take a pair $(i,j) \in IJ^{\times}$ and assume $e_{i,j}$ has been given a value. Then, only Boolean variable $Y_{\tau_{i,j},i,j}$ with subindex $\tau_{i,j} = e_{i,j}$ is given the value of $True$, while the remaining Boolean variables $Y_{\tau_{i,j},i,j}$ with subindex $\tau_{i,j} \neq e_{i,j}$ are set to $False$. As a result, for every pair $(i,j) \in IJ^{\times}$, this reformulation allows to reformulate $|\mathrm{T}_{i,j}|$ Boolean variables as a single integer variable $e_{i,j}$. Based on the above, the reformulated version of the optimization problem introduced in Eqs. (8.1A) and (8.1B) is as follows:

$$\begin{aligned} &\min_{s.t.} f\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}\right) \\ &\boldsymbol{v}^{sch} = [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}] \in SCH, \boldsymbol{v}^{dyn} \in DYN, [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}] \in LINK \\ &\boldsymbol{e} \in E, \boldsymbol{n} \in N \end{aligned} \tag{8.4}$$

Note that a reformulation of the binary variables $x_{i,j,t}, \forall (i,j) \in IJ^{\times}, \forall t \in T$ that appear in $\boldsymbol{y}^{sch}$ is not trivial given that constraints that satisfy requirement R3 such as $\sum_{t \in T} x_{i,j,t} = 1$ do not appear in the STN formulation. This means that it is not known *a priori* how many variables from $x_{i,j,t} \forall t \in T$ will take the value of 1 in the optimal solution; thus, the number of external variables needed to reformulate

variables in $\boldsymbol{y}^{sch}$ is unknown. A more detailed explanation of why $\boldsymbol{y}^{sch}$ cannot be reformulated is provided at the end of section H.1 of appendix H.

8.2 Algorithm framework

Optimization problem (8.4) is solved following the master-subproblem decomposition introduced in section 8.2.1. The master problem is subject to multiple benders cuts generated at each iteration, following the guidelines described in section 8.2.2. On the other hand, subproblems can be either solved directly, or following the subproblem solution techniques proposed in section 8.2.3. Moreover, section 8.2.4 incorporates these ideas into the proposed MLD-BD algorithm, which aims to overcome some of the limitations of current GBD strategies when dealing with SSDO problems. For instance, despite the fact that MLD-BD is a local exploration algorithm, it may find better solutions than GBD for some applications, as illustrated through examples H.2 and H.3 in section H.2 of appendix H. These examples illustrate how MLD-BD may be better than GBD at finding high quality local solutions (and vice versa) depending on the convexity characteristics of the problem under consideration. Furthermore, GBD may converge to an infeasible solution when feasibility cuts are not incorporated, even when the problem is initialized form a feasible solution. The feasibility issues faced by GBD for SSDO problems are studied in detail in Appendix B. In contrast, the proposed MLD-BD framework is able to move away from infeasibilities once a first feasible solution is identified and still guarantee convergence to a feasible solution in a finite number of iterations (see proof in section H.3 of appendix H).

8.2.1 Master-subproblem decomposition

Based on LBBD theory, problem (8.4) with complicating variables $[\boldsymbol{n}, \boldsymbol{e}]$ can be decomposed into a master problem (8.5B) and subproblems (8.5A). The latter results from fixing complicating variables $[\boldsymbol{n}, \boldsymbol{e}]$ in problem (8.4). The fixed values of complicating variables $[\boldsymbol{n}, \boldsymbol{e}]$ evaluated by the subproblem are iteratively generated by master problem (8.5B). The solution of subproblem (8.5A) with complicating variables fixed returns objective function $f_0^*(\boldsymbol{n}, \boldsymbol{e})$, which is assumed to take a value of $+\infty$ when the subproblem is infeasible for a given combination in $\boldsymbol{n}$ and $\boldsymbol{e}$. Variables $\boldsymbol{y}$ also remain fixed in the subproblem because they can be calculate based on the values of $\boldsymbol{e}$. Accordingly, the optimization of vector $[\boldsymbol{n}, \boldsymbol{e}]$ is delegated to the MIP master problem (8.5B), which iteratively generates an updated trial vector of complicating variables $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ that is tested by subproblem (8.5A),

where k is used to denote the current iteration and $\hat{k}$ is an auxiliar index to denote current and/or previous iterations.

$$f_0^*(\boldsymbol{n},\boldsymbol{e}) = \min_{\boldsymbol{x}^{sch},\boldsymbol{y}^{sch},\boldsymbol{v}^{dyn}} f\left(\boldsymbol{y}(\boldsymbol{e}),\boldsymbol{n},\boldsymbol{x}^{sch},\boldsymbol{y}^{sch},\boldsymbol{v}^{dyn}\right) \tag{8.5A}$$

$$s.t.\ [\boldsymbol{y}(\boldsymbol{e}),\boldsymbol{n},\boldsymbol{x}^{sch},\boldsymbol{y}^{sch}] \in SCH, \boldsymbol{v}^{dyn} \in DYN,$$

$$[\boldsymbol{y}(\boldsymbol{e}),\boldsymbol{n},\boldsymbol{x}^{sch},\boldsymbol{y}^{sch},\boldsymbol{v}^{dyn}] \in LINK$$

$$[z^{k+1},\boldsymbol{n}^{k+1},\boldsymbol{e}^{k+1}] \in \arg\min_{z,\boldsymbol{n},\boldsymbol{e}} z \tag{8.5B}$$

$$s.t.\ z \geq \boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k}}}(\boldsymbol{n},\boldsymbol{e}), \forall\, \hat{k} \in \{1,2,\dots,k\}, \boldsymbol{e} \in E, \boldsymbol{n} \in N, z \in \mathbb{R}$$

$$\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k}}}(\boldsymbol{n},\boldsymbol{e}) = \left[\beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},1}}(\boldsymbol{n},\boldsymbol{e}), \beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},2}}(\boldsymbol{n},\boldsymbol{e}), \dots, \beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},n_{\hat{k}}}}(\boldsymbol{n},\boldsymbol{e})\right]$$

Note that problem (8.4) can be equivalently expressed in terms of $f_0^*(\boldsymbol{n},\boldsymbol{e})$ as shown in problem (8.6A), which is a key equivalence in the derivation of master problem (8.5B). Problem (8.6A) is in turn equivalent to problem (8.6B), i.e., problem (8.6B) is the epigraph form of problem (8.6A). Constraints in problem (8.6B) require *a-priori* knowledge of the subproblem's objective value $f_0^*(\boldsymbol{n},\boldsymbol{e})$ for every feasible value of $[\boldsymbol{n},\boldsymbol{e}]$; thus, problem (8.6B) is not useful in practice. Consequently, the master problem (8.5B) approximates constraints $z \geq f_0^*(\boldsymbol{n},\boldsymbol{e})$ in problem (8.6B) using a vector of Benders cuts $\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k}}}(\boldsymbol{n},\boldsymbol{e})$ obtained after solving subproblem (8.5A) at trial value $[\boldsymbol{n},\boldsymbol{e}]^{\hat{k}} = [\boldsymbol{n}^{\hat{k}},\boldsymbol{e}^{\hat{k}}]$. Note that Benders cuts must be designed to suit our problem structure, i.e., cuts for problems involving ordered discrete decisions (see section 8.2.2). In general, Benders cuts are generated based on information from the subproblems evaluated in previous iterations $\hat{k} = 1,2,\dots,k$. Note that, if Benders cuts in equation (8.5B) are equivalently expressed as shown in equation (8.6C), then the right-hand-side of equation (8.6C) can be described as an estimator of $f_0^*(\boldsymbol{n},\boldsymbol{e})$ at iteration k. The accuracy of the estimation provided by Benders cuts improves as the iterations progress and new subproblems are solved. In the remainder of this chapter, I refer to the objective function returned by subproblems as $f^*(\boldsymbol{n},\boldsymbol{e})$ instead of $f_0^*(\boldsymbol{n},\boldsymbol{e})$ to acknowledge that the solution of subproblems may involve other steps when $f(\boldsymbol{n},\boldsymbol{e})$ has special characteristics. For example, this study covers the case when the solution of subproblems may incorporate a pruning strategy, resulting in expressions for $f^*(\boldsymbol{n},\boldsymbol{e})$ different than equation (8.5A) where $f^*(\boldsymbol{n},\boldsymbol{e}) = f_0^*(\boldsymbol{n},\boldsymbol{e})$. A detailed discussion on this point is given in section 8.2.3. The multicut strategy is discussed next.

$$\min\{f_0^*(\boldsymbol{n},\boldsymbol{e}): \boldsymbol{e}\in E, \boldsymbol{n}\in N\} \tag{8.6A}$$

$$\min\{z: z\geq f_0^*(\boldsymbol{n},\boldsymbol{e}), \boldsymbol{e}\in E, \boldsymbol{n}\in N, z\in\mathbb{R}\} \tag{8.6B}$$

$$z\geq \max_{\hat{k}\in\{1,2,\ldots,k\},\theta\in\{1,2,\ldots,n_{\hat{k}}\}} \beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},\theta}}(\boldsymbol{n},\boldsymbol{e}) \tag{8.6C}$$

8.2.2 Generation of multiple benders cuts per iteration

Traditional LBBD strategies only generate one new Benders cut $\beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k}}}(\boldsymbol{n},\boldsymbol{e})$ per iteration. However, the solution strategy proposed in this work generates a vector with multiple Benders cuts $\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k}}}(\boldsymbol{n},\boldsymbol{e})$ per iteration, by evaluating $f^*(\boldsymbol{n},\boldsymbol{e})$ within local neighborhood of each trial vector $[\boldsymbol{n}^k,\boldsymbol{e}^k]$. The local neighborhood considered for the generation of cuts is $N_2(\boldsymbol{n}^k,\boldsymbol{e}^k)=\left\{[\boldsymbol{n},\boldsymbol{e}]\in\mathbb{Z}^{n_n+n_e}: \left\|[\boldsymbol{n}-\boldsymbol{n}^k,\boldsymbol{e}-\boldsymbol{e}^k]\right\|_2\leq 1\right\}$. Evaluating $f^*(\boldsymbol{n},\boldsymbol{e})$ for every point $N_2(\boldsymbol{n}^k,\boldsymbol{e}^k)$ allows to evaluate the sensitivity of $f^*(\boldsymbol{n},\boldsymbol{e})$ by increasing or decreasing the value of each variable in $[\boldsymbol{n}^k,\boldsymbol{e}^k]$ by one unit. Given that cuts are generated based on local neighborhood information, only local optimality can be guaranteed by the proposed MLD-BD approach. Also, the generation of Benders cuts using local neighborhood information of $[\boldsymbol{n}^k,\boldsymbol{e}^k]$ requires cuts generated in previous iterations to be updated, due to the non-convex characteristics of function $f^*(\boldsymbol{n},\boldsymbol{e})$ when handling general MINLP or GDP problems. If Benders cuts are not updated, they will provide a poor approximation to $f^*(\boldsymbol{n},\boldsymbol{e})$ that may guide the framework towards sub-optimal solutions[15]. Hence, the cut refinement strategy proposed in chapter 4 for superstructure optimization is adapted to consider the problem in this work. The details of the multicut generation and refinement strategy proposed in this work are described next.

One characteristic of cuts in LBBD is that these are designed to be linear to guarantee that the master problem (8.5B) remains as an MIP. Thus, Benders cuts used in this work are defined in equation (8.7), which indicates that at iteration k a new vector of linear Benders cuts $\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^k}(\boldsymbol{n},\boldsymbol{e})$ is generated, and previously generate vectors $\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^1}(\boldsymbol{n},\boldsymbol{e})$, $\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^2}(\boldsymbol{n},\boldsymbol{e})$,…, $\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^{k-1}}(\boldsymbol{n},\boldsymbol{e})$ are updated. Each individual Benders cut is expressed as a weighted linear combination of $\boldsymbol{n}$ and $\boldsymbol{e}$ plus an offset term. Hence, $\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k}}}(\boldsymbol{n},\boldsymbol{e})$ is expressed as shown in Eq. (8.7), where each column of matrices $\boldsymbol{P}^{\hat{k}}$ and $\boldsymbol{Q}^{\hat{k}}$ contain the weights used in the linear combination of row vectors $\boldsymbol{n}$ and $\boldsymbol{e}$ that define an individual cut, whereas vector $\boldsymbol{\alpha}^{\hat{k}}$ contains the offset value for each cut. For the remainder of this section, assume that vectors

(lowercase bold notation) are row vectors, and superscript T is used to transpose them. The equations used to compute $\boldsymbol{P}^{\hat{k}}$, $\boldsymbol{Q}^{\hat{k}}$ and $\boldsymbol{\alpha}^{\hat{k}}$ are given below.

$$\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k}}}(\boldsymbol{n},\boldsymbol{e}) = \boldsymbol{n}\boldsymbol{P}^{\hat{k}} + \boldsymbol{e}\boldsymbol{Q}^{\hat{k}} + \boldsymbol{\alpha}^{\hat{k}}\ ,\forall\ \hat{k} \in \{1,2,\dots,k\} \tag{8.7}$$

Benders cuts are obtained in this work through local neighborhood verification, where multiple subproblems are solved per iteration, as opposed to traditional LBBD strategies that solve a single subproblem per iteration. Notation $\left[\boldsymbol{n}^{k,\theta},\boldsymbol{e}^{k,\theta}\right]$ is used to differentiate between central point $[\boldsymbol{n}^{k},\boldsymbol{e}^{k}] = [\boldsymbol{n}^{k,1},\boldsymbol{e}^{k,1}]$ with $\theta = 1$, and a neighboring point $\left[\boldsymbol{n}^{k,\theta},\boldsymbol{e}^{k,\theta}\right]$ with $\theta \in \{2,3,\dots,|N_2|\}$ evaluated at iteration k. Hence, the idea of the proposed approach is to add as many new cuts as subproblems solved per iteration. As described in section 8.2.1, the estimation of $f^*(\boldsymbol{n},\boldsymbol{e})$ provided by Benders cuts improves as more subproblems are considered in the generation of multiple cuts. Hence, the size of $\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k}}}(\boldsymbol{n},\boldsymbol{e})$ is equal to $n_{\hat{k}} = \left|\widetilde{D}^{\hat{k}}\right|$, where $\left|\widetilde{D}^{\hat{k}}\right|$ stands for the number of subproblems solved at iteration $\hat{k}$, i.e., $\widetilde{D}^{\hat{k}}$ is the set with fixed values of complicating variables $\left\{[\boldsymbol{n}^{\hat{k},1},\boldsymbol{e}^{\hat{k},1}],\dots,[\boldsymbol{n}^{\hat{k},\left|\widetilde{D}^{\hat{k}}\right|},\boldsymbol{e}^{\hat{k},\left|\widetilde{D}^{\hat{k}}\right|}]\right\}$ used to evaluate $f^*(\boldsymbol{n},\boldsymbol{e})$ at iteration $\hat{k}$, and $\left|\widetilde{D}^{\hat{k}}\right|$ is the size of this set. I also define set D^k, which contains all combinations of $[\boldsymbol{n},\boldsymbol{e}]$ used to evaluate $f^*(\boldsymbol{n},\boldsymbol{e})$ prior and up to iteration k. Sets D^k and $\widetilde{D}^{\hat{k}}$ are mathematically defined in Eqs. (8.8A)-(8.8C). Eq. (8.8A) initializes D^0 as an empty set, which is used to define D^k recursively in Eq. (8.8B). According to Eq. (8.8B), if the subproblem associated to central point $[\boldsymbol{n}^{k},\boldsymbol{e}^{k}]$ is feasible ($f^*\left(\boldsymbol{n}^{k},\boldsymbol{e}^{k}\right) < +\infty$), then $f^*(\boldsymbol{n},\boldsymbol{e})$ is evaluated within $N_2\left(\boldsymbol{n}^{k},\boldsymbol{e}^{k}\right)$ and all the points in $N_2\left(\boldsymbol{n}^{k},\boldsymbol{e}^{k}\right)$ are added to D^k; otherwise, $f^*\left(\boldsymbol{n}^{k},\boldsymbol{e}^{k}\right) = +\infty$ indicates infeasibility and $f^*(\boldsymbol{n},\boldsymbol{e})$ is not evaluated within $N_2\left(\boldsymbol{n}^{k},\boldsymbol{e}^{k}\right)$. Eq. (8.8C) defines $\widetilde{D}^{\hat{k}}$ as $D^{\hat{k}}\backslash D^{\hat{k}-1}$, i.e., new fixed complicating variables excluding complicating variables from previous iterations.

$$D^0 = \emptyset \tag{8.8A}$$

$$D^k = \begin{cases} D^{k-1} \cup N_2\left(\boldsymbol{n}^{k},\boldsymbol{e}^{k}\right), & f^*\left(\boldsymbol{n}^{k},\boldsymbol{e}^{k}\right) < +\infty \\ D^{k-1} \cup \{[\boldsymbol{n}^{k},\boldsymbol{e}^{k}]\}, & f^*\left(\boldsymbol{n}^{k},\boldsymbol{e}^{k}\right) = +\infty \end{cases} \tag{8.8B}$$

$$\widetilde{D}^{\hat{k}} = D^{\hat{k}}\backslash D^{\hat{k}-1} = \left\{[\boldsymbol{n}^{\hat{k},1},\boldsymbol{e}^{\hat{k},1}],\dots,[\boldsymbol{n}^{\hat{k},\left|\widetilde{D}^{\hat{k}}\right|},\boldsymbol{e}^{\hat{k},\left|\widetilde{D}^{\hat{k}}\right|}]\right\} \tag{8.8C}$$

Given that vectors of Benders cuts have size $\left|\widetilde{D}^{\hat{k}}\right|$, matrices $\boldsymbol{P}^{\hat{k}}$, $\boldsymbol{Q}^{\hat{k}}$ and vector $\boldsymbol{\alpha}^{\hat{k}}$ in Eq. (8.7) have $\left|\widetilde{D}^{\hat{k}}\right|$ columns $\left(\boldsymbol{p}^{\hat{k},\theta}\right)^{T}$, $\left(\boldsymbol{q}^{\hat{k},\theta}\right)^{T}$ and entries $\alpha^{\hat{k},\theta}, \forall\theta \in \left\{1,\dots,\widetilde{D}^{\hat{k}}\right\}$, respectively (see Eq. 8.9A). Each combination of $\boldsymbol{p}^{\hat{k},\theta}$, $\boldsymbol{q}^{\hat{k},\theta}$ and $\alpha^{\hat{k},\theta}$ define the required parameters to obtain an individual Benders cut of a central or neighboring point θ at iteration $\hat{k}$, i.e., Benders cut $\beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},\theta}}(\boldsymbol{n},\boldsymbol{e})$. Each individual set of parameters for Benders cut $\beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},\theta}}(\boldsymbol{n},\boldsymbol{e})$ can be obtained using Eq. (8.9B), which has the form of a linear programming (LP) problem, where notation $\langle\cdot,\cdot\rangle$ represents the dot product. This cut generation strategy has its roots in discrete convex analysis, which uses LPs of this type to verify the integrally convex properties of functions with integer arguments. Hence the generation of cuts through this auxiliary LP is the main element that distinguishes the proposed MLD-BD algorithm from traditional MINLP techniques such as GBD. In this work, I use this LP to return a Benders cut $\beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},\theta}}(\boldsymbol{n},\boldsymbol{e})$ that closely satisfies the global optimality requirements imposed over Benders cuts by Hooker et al.[10], i.e., 1) each newly added cut $\beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},\theta}}(\boldsymbol{n},\boldsymbol{e})$ should underestimate $f^{*}(\boldsymbol{n},\boldsymbol{e}), \forall\boldsymbol{e} \in E, \forall\boldsymbol{n} \in N$, and 2) $f^{*}\left(\boldsymbol{n}^{\hat{k},\theta},\boldsymbol{e}^{\hat{k},\theta}\right)$ should agree with the value of the cut generated at $\left[\boldsymbol{n}^{\hat{k},\theta},\boldsymbol{e}^{\hat{k},\theta}\right]$, when evaluated at $\left[\boldsymbol{n}^{\hat{k},\theta},\boldsymbol{e}^{\hat{k},\theta}\right]$, i.e., $\beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},\theta}}\left(\boldsymbol{n}^{\hat{k},\theta},\boldsymbol{e}^{\hat{k},\theta}\right) = f^{*}\left(\boldsymbol{n}^{\hat{k},\theta},\boldsymbol{e}^{\hat{k},\theta}\right)$. The first requirement is approximately satisfied by the constraints in Eq. (8.9B), which ensure that newly added cuts underestimate $f^{*}(\boldsymbol{n},\boldsymbol{e})$ within $D^{k} \subseteq E \times N$. The maximization of objective function in Eq. (8.9B) tries to satisfy the second requirement by pushing $\beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},\theta}}\left(\boldsymbol{n}^{\hat{k},\theta},\boldsymbol{e}^{\hat{k},\theta}\right)$ towards $f^{*}\left(\boldsymbol{n}^{\hat{k},\theta},\boldsymbol{e}^{\hat{k},\theta}\right)$; however, only inequality $\beta_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k},\theta}}\left(\boldsymbol{n}^{\hat{k},\theta},\boldsymbol{e}^{\hat{k},\theta}\right) \leq f^{*}\left(\boldsymbol{n}^{\hat{k},\theta},\boldsymbol{e}^{\hat{k},\theta}\right)$ can be satisfied when solving general non-convex optimization problems. As a result, global optimality cannot be ensured for f^{*} when using cuts described by Eq. (8.9B). The steps required to generate multiple Benders cuts using the present approach are shown in Table 8.1. This multicut strategy is expected to improve convergence speed, as it aims to provide a more exact estimation of f^{*} and it does not require to solve additional subproblems (generally MINLP) when compared to the standard LD-BD methodology introduced in chapter 4. Instead, it only requires solving additional LPs (Eq. 8.9B).

Table 8.1. Steps to obtain multiple benders cuts at iteration k, and refine previously generated cuts.

Step 1. Feasibility verification
If $f^*(\boldsymbol{n}^k, \boldsymbol{e}^k) < +\infty$, obtain f^* for every $[\boldsymbol{n}, \boldsymbol{e}]$ within $N_2(\boldsymbol{n}^k, \boldsymbol{e}^k)$.

Step 2. Updating D^k and $\widetilde{D}^k$
If $k = 1$, let $D^0 = \emptyset$ (see Eq. (8.8A)).

If $f^*(\boldsymbol{n}^k, \boldsymbol{e}^k) < +\infty$, add every element of $N_2(\boldsymbol{n}^k, \boldsymbol{e}^k)$ to D^k and store $f^*(\boldsymbol{n}, \boldsymbol{e}), \forall [\boldsymbol{n}, \boldsymbol{e}] \in N_2(\boldsymbol{n}^k, \boldsymbol{e}^k)$. Else, add $[\boldsymbol{n}^k, \boldsymbol{e}^k]$ to D^k and store $f^*(\boldsymbol{n}^k, \boldsymbol{e}^k)$ (see Eq. (8.8B)).

Define $\widetilde{D}^k$ as $\widetilde{D}^k = D^k \backslash D^{k-1}$ (see Eq. (8.8C)).

Step 3. Refinement of Benders cuts
Solve the optimization problem in Eq. (8.9B), for every $\hat{k}$ in $\{1,2,\dots,k\}$ and every θ in $\{1,2,\dots,|\widetilde{D}^{\hat{k}}|\}$. Then use the optimal solution found to update $\boldsymbol{\beta}_{[\boldsymbol{n},\boldsymbol{e}]^{\hat{k}}}(\boldsymbol{n}, \boldsymbol{e}), \forall\, \hat{k} \in \{1,2,\dots,k\}$ using Eqs. (8.7) and (8.9A).

$$\boldsymbol{P}^{\hat{k}} = \left[\left(\boldsymbol{p}^{\hat{k},1}\right)^T, \dots, \left(\boldsymbol{p}^{\hat{k},|\widetilde{D}^{\hat{k}}|}\right)^T\right], \boldsymbol{Q}^{\hat{k}} = \left[\left(\boldsymbol{q}^{\hat{k},1}\right)^T, \dots, \left(\boldsymbol{q}^{\hat{k},|\widetilde{D}^{\hat{k}}|}\right)^T\right], \tag{8.9A}$$
$$\boldsymbol{\alpha}^{\hat{k}} = \left[\alpha^{\hat{k},1}, \dots, \alpha^{\hat{k},|\widetilde{D}^{\hat{k}}|}\right], \forall\, \hat{k} \in \{1,2,\dots,k\}$$

$$\left[\boldsymbol{p}^{\hat{k},\theta}, \boldsymbol{q}^{\hat{k},\theta}, \alpha^{\hat{k},\theta}\right] \in \arg \max_{\boldsymbol{p}\in\mathbb{R}^{n_n}, \boldsymbol{q}\in\mathbb{R}^{n_e}, \alpha\in\mathbb{R}} \langle \boldsymbol{p}, \boldsymbol{n}^{\hat{k},\theta}\rangle + \langle \boldsymbol{q}, \boldsymbol{e}^{\hat{k},\theta}\rangle + \alpha \quad , \tag{8.9B}$$
$$s.t.\ \langle \boldsymbol{p}, \boldsymbol{n}\rangle + \langle \boldsymbol{q}, \boldsymbol{e}\rangle + \alpha \le f^*(\boldsymbol{n}, \boldsymbol{e}), \forall [\boldsymbol{n}, \boldsymbol{e}] \in D^k$$
$$\forall\, \hat{k} \in \{1,2,\dots,k\}, \forall \theta \in \{1,2,\dots,|\widetilde{D}^{\hat{k}}|\}$$

8.2.3 Proposed strategies for the solution of subproblems

8.2.3.1 Pruning strategy

This work explores alternative strategies for the solution of subproblems, which are needed to avoid the time-consuming solution of every MINLP subproblem that emerges when using the MLD-BD strategy to solve SSDO problems. The strategies herein introduced provide different approaches to obtain subproblem objective function $f^*(\boldsymbol{n}, \boldsymbol{e})$ other than the naïve solution of MINLP subproblem (8.5A), which yields the optimal objective function $f^*(\boldsymbol{n}, \boldsymbol{e}) = f_0^*(\boldsymbol{n}, \boldsymbol{e})$. In this chapter, I introduce three strategies to obtain $f^*(\boldsymbol{n}, \boldsymbol{e})$ when considering an optimization problem whose objective function has the form shown in Eqs. (8.10A) and (8.10B). Eq. (8.10A) formulates the objective function as the

sum of f_1 and f_2. Objective functions with this form are often found in SSDO formulations, e.g., see Chu and You[61], Nie et al.[62] and Ji and Gu[356]. While f_1 considers scheduling-related terms $\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}$ and $\boldsymbol{y}^{sch}$, f_2 accounts for the interactions of scheduling and control decisions by considering the effect of dynamic variables ($\boldsymbol{v}^{dyn}$) along with $\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}$ and $\boldsymbol{y}^{sch}$ in the objective function. For instance, f_1 may include the fixed processing costs of tasks, the purchase costs of raw materials, and the revenue generated by selling products, while f_2 may include utility costs (e.g., heating and cooling costs) needed to maintain the dynamic operation near optimal conditions.

$$f\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}\right) = f_1\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}\right) + f_2\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}\right) \quad (8.10A)$$

$$f_2\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}\right) \geq 0 \quad (8.10B)$$

A requirement that must be satisfied by Eq. (8.10A) to use the strategies discussed through this section is shown in Eq. (10B) and indicates that term f_2 must always be positive. This requirement is satisfied by the batch network SSDO problems considered in this work, where f_2 only considers costs that are known to be positive, e.g., cooling cost, heating costs, purchase costs of a chemical, etc. f_1 is guaranteed to be a lower bound of the original objective function (8.9A) (i.e., $f_1 \leq f$) under the assumption shown in equation (8.10B), as demonstrated in chapter 6. This property is exploited in this work to formulate a pruning strategy that allows to reduce the number of MINLP subproblems that must be solved at each iteration of the MLD-BD algorithm, with the objective of reducing computational time. To accomplish this goal, consider optimization problem (8.11A), which is a simplified version of the original subproblem (8.5A), except that it only minimizes the scheduling objective function (i.e., f_1) and neglects transient costs, i.e. f_2. Since f_1 is not affected by dynamic variables in $\boldsymbol{v}^{dyn}$, dynamic constraints $\boldsymbol{v}^{dyn} \in DYN$ were removed from (8.11A). Thus, the key benefit of minimizing f_1 is that nonlinear terms in DYN are not required in the formulation, thus making the simplified subproblem (8.11A) a MIP problem instead of a MINLP. In addition, the optimal objective function value $f_1^*(\boldsymbol{n}, \boldsymbol{e})$ of optimization problem (8.11A) is expected to be a lower bound of $f^*(\boldsymbol{n}, \boldsymbol{e})$ for fixed values of $\boldsymbol{n}$ and $\boldsymbol{e}$, given that $f_2 > 0$, i.e., $f_1^*(\boldsymbol{n}, \boldsymbol{e})$ would be the optimal objective function of subproblem (8.5A) under the additional assumption that operating/transient costs f_2 are negligible.

$$f_1^*(\boldsymbol{n},\boldsymbol{e}) = \min_{\boldsymbol{x}^{sch},\boldsymbol{y}^{sch}} f_1\left(\boldsymbol{y}(\boldsymbol{e}),\boldsymbol{n},\boldsymbol{x}^{sch},\boldsymbol{y}^{sch}\right) \tag{8.11A}$$

$$s.t.\,[\boldsymbol{y}(\boldsymbol{e}),\boldsymbol{n},\boldsymbol{x}^{sch},\boldsymbol{y}^{sch}] \in SCH, [\boldsymbol{y}(\boldsymbol{e}),\boldsymbol{n},\boldsymbol{x}^{sch},\boldsymbol{y}^{sch},\boldsymbol{v}^{dyn}] \in LINK$$

$$f^*(\boldsymbol{n},\boldsymbol{e}) = \begin{cases} f_1^*(\boldsymbol{n},\boldsymbol{e}), f_1^*(\boldsymbol{n},\boldsymbol{e}) \geq f^{best} \\ f_0^*(\boldsymbol{n},\boldsymbol{e}), f_1^*(\boldsymbol{n},\boldsymbol{e}) < f^{best} \end{cases} \tag{8.11B}$$

Based on the above, the proposed subproblem solution and pruning strategy to obtain $f^*(\boldsymbol{n},\boldsymbol{e})$ is shown in Eq. (8.11B). This equation uses f^{best} as input, which corresponds to the objective function of the incumbent solution, i.e., the current best value of f^*. Note that there is no incumbent before step 1 of the MLD-BD search, hence f^{best} is initialized at $+\infty$. Eq. (8.11B) uses the current value of f^{best} and proceeds as follows: the MIP simplified subproblem (8.11A) is solved first and its optimal objective function $f_1^*(\boldsymbol{n},\boldsymbol{e})$ is compared to f^{best}. If $f_1^*(\boldsymbol{n},\boldsymbol{e}) \geq f^{best}$, then $f^*(\boldsymbol{n},\boldsymbol{e})$ is set equal to $f_1^*(\boldsymbol{n},\boldsymbol{e})$ and there is no need to solve the original MINLP subproblem in (8.5A), whose objective value is expected to be worse than the incumbent f^{best}, i.e., $f_0^*(\boldsymbol{n},\boldsymbol{e}) \geq f_1^*(\boldsymbol{n},\boldsymbol{e}) \geq f^{best}$ given the property in equation (8.10B). Otherwise, the original MINLP subproblem is solved, and its optimal objective function value $f_0^*(\boldsymbol{n},\boldsymbol{e})$ is assigned to $f^*(\boldsymbol{n},\boldsymbol{e})$. The main advantage of this approach is that the rigorous MINLP solution of subproblem (8.5A) is avoided (i.e., pruned) during the MLD-BD algorithm anytime $f_1^*(\boldsymbol{n},\boldsymbol{e}) \geq f^{best}$, thus reducing the computational costs.

8.2.3.2 Approximated solution of subproblems

In addition to the pruning approach discussed above, the MLD-BD strategy can also be coupled with heuristics to further simplify the solution of subproblems and reduce the computational costs. Hence, this work proposes the pruning strategy with heuristic solution of subproblems, as shown in Eqs. (8.12A)-(8.12C). The steps required to solve subproblems using this strategy are similar to those used for the pruning strategy in Eqs. (8.11A) and (8.11B). First, the approximate MIP scheduling subproblem in Eq. (8.12A) is solved as in subproblem (8.11A), except that the information that needs to be stored from this solution includes the optimal discrete scheduling variables $\boldsymbol{y}^{sch*}$, in addition to the optimal objective value $f_1^*(\boldsymbol{n},\boldsymbol{e})$. As shown in Eq. (8.12C), if $f_1^*(\boldsymbol{n},\boldsymbol{e}) \geq f^{best}$, then $f^*(\boldsymbol{n},\boldsymbol{e})$ is set equal to $f_1^*(\boldsymbol{n},\boldsymbol{e})$, as in Eq. (8.11B). Otherwise, instead of solving the original MINLP subproblem, the solution of a heuristic NLP subproblem with integer scheduling variables $\boldsymbol{y}^{sch*}$ fixed that yields optimal objective $f_0^*(\boldsymbol{n},\boldsymbol{e})$ is considered in this work (see Eq. (8.12B)). The idea behind this heuristic is that

although binary scheduling decisions in ($\boldsymbol{y}^{sch}$) may indirectly impact the objective function in heuristic subproblem (8.12B) through its constraints, the variables that typically have a direct effect on the objective function are the continuous dynamic variables ($\boldsymbol{v}^{dyn}$), the continuous scheduling variables ($\boldsymbol{x}^{sch}$), the integer variables in ($\boldsymbol{n}$), and the binary/Boolean variables that have been reformulated with external variables ($\boldsymbol{y}(\boldsymbol{e})$). Consequently, the proposed heuristic neglects potential interactions between $\boldsymbol{y}^{sch}$ and $\boldsymbol{v}^{dyn}$, but it still considers the scheduling-control interactions between $\boldsymbol{n}$, $\boldsymbol{y}(\boldsymbol{e})$ and $\boldsymbol{v}^{dyn}$ in the master problem (8.5B), and the interactions between $\boldsymbol{x}^{sch}$ and $\boldsymbol{v}^{dyn}$ in the heuristic subproblem (8.12B). For example, an economic objective function is commonly used in literature when considering the SSDO of network batch processes. This objective function generally does not depend directly on sequencing and timing decisions in $\boldsymbol{y}^{sch}$, but it typically includes fixed scheduling costs that may depend on the frequency a task is executed (considered in $\boldsymbol{n}$), inventories (considered in $\boldsymbol{x}^{sch}$), control actions (considered in $\boldsymbol{v}^{dyn}$), and variables coupling scheduling and control actions such as batch sizes and processing times (considered in both $\boldsymbol{v}^{dyn}$ and $\boldsymbol{x}^{sch}$)[61,62,355,366].

$$[\boldsymbol{x}^{sch*}, \boldsymbol{y}^{sch*}] \in \underset{\boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}}{\arg\min}\, f_1\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}\right) \tag{8.12A}$$
$$s.t. [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}] \in SCH, [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}] \in LINK$$

$$f_{\tilde{0}}^{*}(\boldsymbol{n}, \boldsymbol{e}) = \min_{\boldsymbol{x}^{sch}, \boldsymbol{v}^{dyn}} f\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}\right) \tag{8.12B}$$
$$s.t. [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}] \in SCH, \boldsymbol{v}^{dyn} \in DYN,$$
$$[\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}] \in LINK,$$
$$\boldsymbol{y}^{sch} = \boldsymbol{y}^{sch*}$$

$$f^{*}(\boldsymbol{n}, \boldsymbol{e}) = \begin{cases} f_1^{*}(\boldsymbol{n}, \boldsymbol{e}), f_1^{*}(\boldsymbol{n}, \boldsymbol{e}) \geq f^{best} \\ f_{\tilde{0}}^{*}(\boldsymbol{n}, \boldsymbol{e}), f_1^{*}(\boldsymbol{n}, \boldsymbol{e}) < f^{best} \end{cases}, \text{ where } f_1^{*}(\boldsymbol{n}, \boldsymbol{e}) = f_1\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch*}, \boldsymbol{y}^{sch*}\right) \tag{8.12C}$$

The pruning strategy (Eqs. (8.11A) and (8.11B)) avoids the solution of unnecessary MINLP subproblems, and the heuristic strategy (Eqs. (8.12A)-(8.12C)) intends to facilitate the solution of the remaining MINLP subproblem by expressing them as a sequence of an MIPs and an NLPs; nevertheless, a challenge that may still persist is the combinatorial complexity of each MIP that may lead to long computational times. To partially address this issue, the proposed MLD-BD framework with heuristic subproblems can be coupled with a simple cut off technique to further speed-up the

solution of subproblems (see Eqs. (8.13A)-(8.13C)). The key difference between Eqs. (8.13A)-(8.13C) and Eqs. (8.12A)-(8.12C) is the addition of cut off constraint $f_1 \leq f^{best}$ in the MIP subproblem (8.13A). This cut off constraint provides information of the current incumbent solution f^{best} to the MIP subproblem, and indicates that any solution whose objective function is worse than f^{best} can be discarded. This feature is particularly useful when using branch-and-bound-based solvers such as CPLEX or branch-and-cut-based solvers such as Gurobi, as it allows to avoid the exploration of nodes that are not expected to return objective functions better than f^{best}. Therefore, this strategy is expected to lead to a faster solution of MIP subproblems, as it allows to stop their execution prematurely.

$$[\tilde{\boldsymbol{x}}^{sch}, \tilde{\boldsymbol{y}}^{sch}] \in \underset{\boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}}{\arg\min}\, f_1\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}\right) \tag{8.13A}$$
$$s.t. [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}] \in SCH, [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}] \in LINK, f_1 \leq f^{best}$$

$$f_{\tilde{0}}^{*}(\boldsymbol{n}, \boldsymbol{e}) = \min_{\boldsymbol{x}^{sch}, \boldsymbol{v}^{dyn}} f\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}\right) \tag{8.13B}$$
$$s.t. [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}] \in SCH, \boldsymbol{v}^{dyn} \in DYN,$$
$$[\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}] \in LINK,$$
$$\boldsymbol{y}^{sch} = \tilde{\boldsymbol{y}}^{sch}$$

$$f^{*}(\boldsymbol{n}, \boldsymbol{e}) = \begin{cases} f_{\tilde{1}}^{*}(\boldsymbol{n}, \boldsymbol{e}), f_{\tilde{1}}^{*}(\boldsymbol{n}, \boldsymbol{e}) \geq f^{best} \\ f_{\tilde{0}}^{*}(\boldsymbol{n}, \boldsymbol{e}), f_{\tilde{1}}^{*}(\boldsymbol{n}, \boldsymbol{e}) < f^{best} \end{cases}, \text{where } f_{\tilde{1}}^{*}(\boldsymbol{n}, \boldsymbol{e}) = f_1\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \tilde{\boldsymbol{x}}^{sch}, \tilde{\boldsymbol{y}}^{sch}\right) \tag{8.13C}$$

Note that the relative magnitude of terms f_1 and f_2 depends on the case study under consideration. Accordingly, the greater the ratio f_1/f_2, the tighter the pruning strategy proposed in section 8.2.3.1. For example, f_1 is expected to be larger than f_2 in a case where feedstock material costs and revenue from products dominate the economic performance of the process. Similarly, the magnitude of f_2 plays a more significant role as the dynamics of more unit operations are considered, meaning that more operating costs are considered as dynamic costs in f_2, rather than simpler linear variable costs in f_1. Note that further work is still needed to improve the MLD-BD performance in cases where f_2 dominates over f_1. For example, the pruning criterion can be further tightened with the help of the approximated solution of the subproblems discussed in this section. Following GBD principles, linearized versions of dynamic optimization problems can be constructed by using the Lagrange multipliers of linking constraints from previously solved heuristic NLP subproblems. This would result in a linear model of

dynamic considerations that could be used to tighten the pruning criteria; the evaluation of the viability of this idea is left as future work.

8.2.4 The MLD-BD algorithm for SSDO

The MLD-BD algorithm that considers the master-subproblem decomposition discussed in section 8.2.1, the multicut generation strategy proposed in section 8.2.2, and the strategies to solve subproblems introduced in section 8.2.3 is shown in Figure 8.1. The MLD-BD algorithm extends LD-BD to problems involving both integer variables ($\boldsymbol{n}$) and external variables ($\boldsymbol{e}$), which are common in SSDO formulations, as discussed in section 8.2.1. Another difference between LD-BD and MLD-BD is that the latter takes advantage of solving multiple subproblems per iteration, which allows to generate multiple cuts per iteration following the guidelines in section 8.2.2. It also offers flexibility between alternative subproblem solution strategies, as discussed in section 8.2.3. Moreover, MLD-BD incorporates a new stopping criterion that directly assesses the local optimality characteristic of a solution.

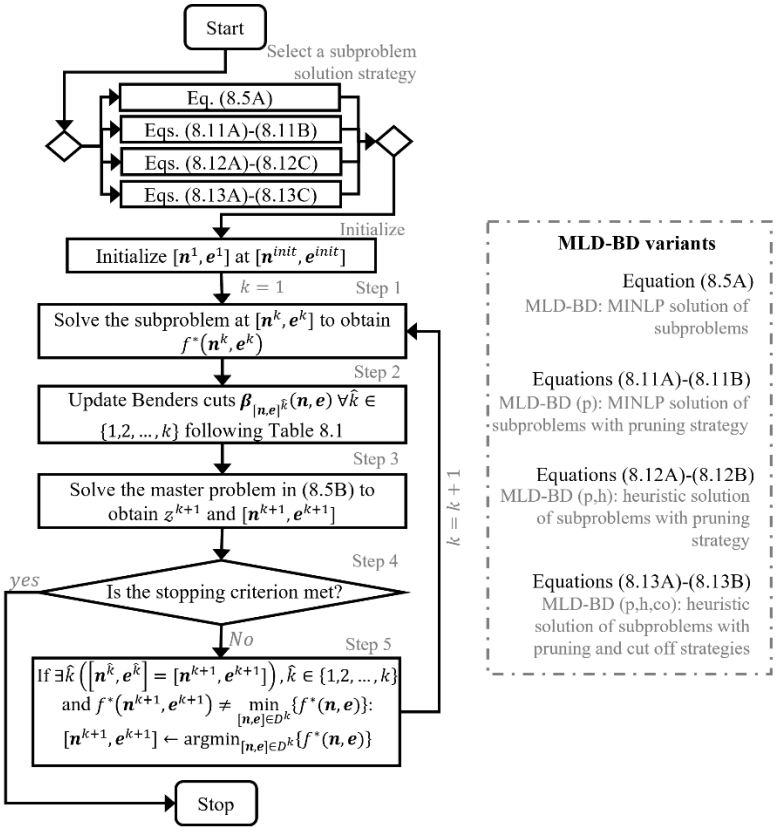


Figure 8.1. The multicut Logic-Based Discrete Benders (MLD-BD) decomposition algorithm.

The MLD-BD algorithm shown in Figure 8.1 can be applied to SSDO problems in the form of Eq. (8.4), when decomposed in a subproblem and MIP master-problem. As shown in Figure 8.1, a subproblem solution strategy must be selected first. Subproblems can be solved directly using Eq. (8.5A) by setting $f^*(\boldsymbol{n},\boldsymbol{e}) = f_0^*(\boldsymbol{n},\boldsymbol{e})$; however, the special structure of objective functions found in SSDO can be exploited to use any of the alternative strategies to obtain $f^*(\boldsymbol{n},\boldsymbol{e})$, as discussed in section 8.2.3. The algorithm then continues with the initialization of complicating variables at a feasible combination $[\boldsymbol{n}^{init},\boldsymbol{e}^{init}]$. If $[\boldsymbol{n}^{init},\boldsymbol{e}^{init}]$ is infeasible, LBBD theory provides guidelines to find a feasible combination of $[\boldsymbol{n}^{init},\boldsymbol{e}^{init}]$ prior to the execution of the algorithm in Figure 8.1, e.g., a LBBD-type feasibility algorithm for problems in the form of Eq. (8.4) is provided in section H.3 of appendix H. Then, iterations start with the solution of the subproblem associated to the current combination of complicating variables $[\boldsymbol{n}^k,\boldsymbol{e}^k]$ (Step 1 in Figure 8.1). Step 2 uses the value of $f^*(\boldsymbol{n}^k,\boldsymbol{e}^k)$ obtained in step 1 to generate multiple Benders cuts and update all cuts previously generated, following the steps described in Table 8.1. All these cuts are used as constraints in the master problem equation (8.5B), which is solved to specify a new trial value of complicating variables $[\boldsymbol{n}^{k+1},\boldsymbol{e}^{k+1}]$, and a new lower bound estimate z^{k+1} (Step 3 in Figure 8.1). The next step evaluates if the stopping criterion is satisfied: If $f^*(\boldsymbol{n}^k,\boldsymbol{e}^k)$ is the best objective function obtained by a subproblem thus far, i.e., $f^*(\boldsymbol{n}^k,\boldsymbol{e}^k) = \min_{[\boldsymbol{n},\boldsymbol{e}]\in D^k}\{f^*(\boldsymbol{n},\boldsymbol{e})\}$, then stop, an optimal local solution has been found. By the definition of D^k in Eq. (8.8B), this criterion guarantees that $f^*(\boldsymbol{n}^k,\boldsymbol{e}^k) \leq f^*(\boldsymbol{n},\boldsymbol{e}), \forall[\boldsymbol{n},\boldsymbol{e}] \in N_2([\boldsymbol{n}^k,\boldsymbol{e}^k])$, i.e., the solution obtained is locally optimal. This stopping criterion offers advantages when compared to the stopping conditions traditionally imposed in LBBD strategies, e.g., a gap such as $|f^*(\boldsymbol{n}^k,\boldsymbol{e}^k) - z^{k+1}|$ or $|f^{best} - z^{k+1}|$ must close within an ϵ tolerance, where f^{best} is the best subproblem objective function evaluated thus far. The proposed stopping criterion avoids the declaration of a user defined optimality tolerance ϵ and the selection of an appropriate optimality gap, whose local optimality guarantees may depend on the case study. Also, it is common in LBBD strategies that a local optimum is identified by the subproblem before the lower bound estimate z^{k+1} closes the optimality gap, e.g., see Zhu et al.[126]. Thus, a stopping criterion that directly verifies the local optimality of a solution candidate may result in faster convergence. If the stopping criterion in step 4 is not satisfied, then the algorithm continues to step 5, as shown in Figure 8.1. This step verifies if the value of complicating variables $[\boldsymbol{n}^{k+1},\boldsymbol{e}^{k+1}]$ obtained in step 3 coincides with a trial value from previous iterations, i.e., $\exists\hat{k}([\boldsymbol{n}^{\hat{k}},\boldsymbol{e}^{\hat{k}}] = [\boldsymbol{n}^{k+1},\boldsymbol{e}^{k+1}])$ with $\hat{k} \in \{1,2,\dots,k\}$. If $[\boldsymbol{n}^{k+1},\boldsymbol{e}^{k+1}]$ is a trial value from previous iterations and

$f^*(\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1})$ does not return the best-known objective function to the problem, then the algorithm moves away from suboptimal solution $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ by assigning $\underset{[\boldsymbol{n},\boldsymbol{e}]\in D^k}{\operatorname{argmin}}\{f^*(\boldsymbol{n}, \boldsymbol{e})\}$ to $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$. This step guarantees convergence of the MLD-BD algorithm in a finite number of iterations.

8.3 Computational experiments

This section illustrates the application of the proposed MLD-BD framework using two case studies involving different features. The first case study aims to illustrate the computational advantages of the different strategies to solve subproblems discussed in sections 8.2.4, and to show how the MLD-BD algorithm performs against the GBD strategy. The second case study is used to highlight the importance of simultaneously considering scheduling and control decisions through the MLD-BD framework to generate economic schedules that significatively differ from those obtained through sequential scheduling and dynamic optimization methodologies. In particular, the second case study highlights the importance of considering the dynamic model of the separations stage, e.g., a batch distillation column model, when considering SSDO solutions such as MLD-BD for network bath processes, instead of heuristic sequential scheduling and dynamic optimization methodologies. Both case studies consider an economic objective function that minimizes the sum of fixed costs (c^{fix}), material costs (c^{mat}), variable costs for tasks with (c^{var_1}) and without (c^{var_2}) dynamic models, and the negative of the revenue ($-rev$). The computational experiments performed in this work were implemented in Pyomo in a PC Intel® Core™ i7-9700K CPU @ 3.60 GHz with 64 GB of RAM. DICOPT, CPLEX and CONOPT4 were used as MINLP, MIP and NLP solvers, respectively.

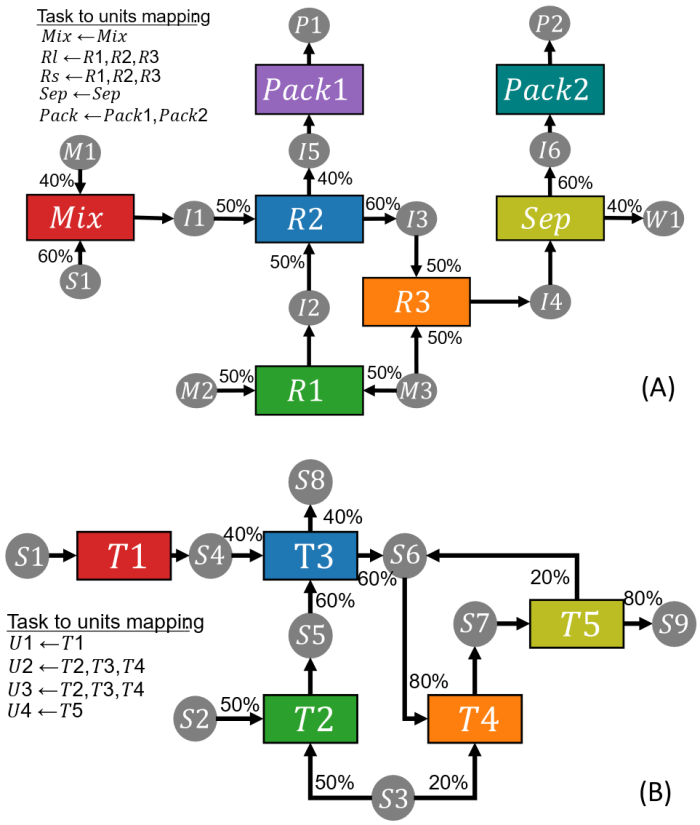


Figure 8.2. A: STN for case study 1. B: STN for case study 2.

8.3.1 Case study 1

This case study optimizes the STN shown in Figure 8.2A originally proposed by Chu and You[61]. The task in the formulation appear in colored blocks, while the states that interconnect these tasks appear as grey circles. Batch reaction tasks $R1$, $R2$ and $R3$ are modeled using differential equations that represent their mass and energy balances to account for the interactions between variable processing time, batch sizes and control actions. These reactions are controlled using heating and cooling utilities that keep the reactor temperatures within feasible operation limits. The remaining tasks, i.e., mixing task Mix, reparation task Sep, and packing tasks $Pack1$ and $Pack2$, are assumed to have fixed processing times and batch sizes that vary between lower and upper bounds. The raw materials of the process are represented through states $M1$, $M2$, $M3$ and solvent $S1$; state $W1$ is a waste product whereas $P1$ and $P2$ are the products that generate revenues. The goal is to maximize the profit of the plant by optimizing the schedule of its batch tasks within a scheduling horizon $\eta = 50400$ seconds, while

simultaneously finding the optimal operating conditions and processing times of reaction tasks $R1$, $R2$ and $R3$ that meet their product quality requirements. Problem statistics including information about the number of variables and constraints are provided in section H.4 of appendix H. Further details about the constraints and variables involved, as well as the problem reformulation, are also provided in section H.4 of appendix H.

The same initialization of variables was considered for every computational experiment performed in this section, using the sequential initialization approach (SIA) proposed by Chu and You[61]. The SIA optimizes control actions first by minimizing the processing time of each task-unit pair $\{R1, R2, R3\} \times \{Rs, Rl\}$ through NLP optimization, followed by a MIP scheduling step. The problem is then solved with different variants of the MLD-BD strategy. For comparison purposes, the problem was also solved using the GBD algorithm. The different MLD-BD and GBD approaches used to solve the problem are as follows:

- **LD-BD**: a version of the MLD-BD method that only adds a single cut per iteration.
- **MLD-BD**: the MLD-BD method with MINLP subproblems solved through Eq. (8.5A).
- **MLD-BD(p)**: the MLD-BD method with the pruning (p) strategy in Eqs. (8.11A) and (8.11B).
- **MLD-BD(p,h)**: the MLD-BD method with the pruning (p) strategy and heuristic (h) solution of subproblems in Eqs. (8.12A)-(8.12C).
- **MLD-BD(p,h,co)**: the MLD-BD method with the pruning (p) strategy, heuristic (h) solution of subproblems and cut off strategy in Eqs. (8.13A)-(8.13C).
- **GBD**: a standard version of GBD that is often used in the solution of SSDO problems (see [62]).
- **GBD(f):** the standard GBD algorithm with feasibility (f) cuts.
- **GBD(f,c):** a standard GBD algorithm that combines feasibility cuts (f) with additional constraints (c) over processing times in the master problem.

The standard GBD strategy, often used in the solution of SSDO problems, decomposes the problem into a MIP master problem that optimizes scheduling variables $\boldsymbol{v}^{sch}$ subject to scheduling and linking constraints, and NLP subproblems that optimize dynamic variables $\boldsymbol{v}^{dyn}$ with scheduling variables fixed[61,62,366]. One key assumption in previous works that implemented the GBD strategy is that NLP subproblems are feasible for all combinations of scheduling variables; nevertheless, general SSDO

problems, such as the one considered in this section, do not satisfy this assumption. Therefore, feasibility (f) cuts were implemented to handle infeasible subproblems, resulting in the GBD (f) variant. I also consider a GBD (f,c) variant that incorporates additional constraints in the master problem to keep variable processing times bounded below by their minimum value, where this minimum processing time is obtained for each reaction $R1$, $R2$ and $R3$ at reactors Rl and Rs through NLP optimization in a pre-processing step. The results obtained for each approach are shown in Table 8.2. Note that a time limit of 86400 seconds for each optimization run was used.

Table 8.2. Performance of MLD-BD and GBD solution strategies (case study 1, $\eta = 50400\ s$).

	SIA (initialization)	LD-BD	MLD-BD	MLD-BD(p)	MLD-BD(p,h)	MLD-BD(p,h, co)	GBD	GBD(f)	GBD(f,c)
Profit objective $(-f)$ [$]	403.42	1109.86	1148.09	1148.09	1148.09	1148.09	890.38	890.38	1129.97
Fixed costs (c^{fix}) [$]	470.00	470	470.00	470.00	470.00	470.00	470.00	470.00	470.00
Variable costs (c^{var_1}) [$]	391.67	381.94	381.94	381.94	381.94	381.94	391.67	391.67	391.67
Dynamic costs (c^{var_2}) [$]	859.92	250.70	212.46	212.46	212.46	212.46	372.95	372.95	233.36
Material costs (c^{mat}) [$]	1875.00	1631.94	1631.94	1631.94	1631.94	1631.94	1875.00	1875.00	1775.00
Revenue (rev) [$]	4000.00	3844.44	3844.44	3844.44	3844.44	3844.44	4000.00	4000.00	4000.00
CPU time [s]	54	86400	6405	981	667	659	150	86400	86400
Iterations	-	198	28	26	26	26	2	310	701

Despite returning a solution after only 54 seconds, SIA only achieved a profit of 403.32$, compared to the 1148.09$ profit returned by the MLD-BD approach, as shown in Table 8.2. MLD-BD required 6405 seconds to converge, thanks to the computational benefits offered by the multicut strategy. For instance, if a single cut per iteration is added instead, the best solution identified by the LD-BD

algorithm still has an objective function of 1109.86$ when reaching the time limit of 86400 seconds, which is 3% worse than the MLD-BD objective function. In addition, the MLD-BD solution of 1148.09$ was consistently found using MLD-BD(p) and MLD-BD(p,h). The MLD-BD(p) strategy shows improvements in terms of convergence speed, reducing the MLD-BD computational time by an order of magnitude. In this case, the pruning strategy avoided the rigorous MINLP solution of 68 subproblems, which corresponds to 60% of the total number of subproblems solved. Among the 68 pruned MINLP subproblems, 30 returned a MIP scheduling solutions with an objective function worse than the incumbent solution, while the remaining 38 subproblems were deemed infeasible based on scheduling information alone; hence they were not solved thus reducing the computational costs. The MLD-BD(p,h) strategy shows further convergence speed improvements, reducing the MLD-BD(p) computational time by 32%. The ordered discrete decisions in both the MLD-BD(p) and MLD-BD(p,h) followed the same convergence path. Note that 74 out of 76 (97%) subproblem objective functions identified by MLD-BD(p,h) are equal to those identified by MLD-BD(p). Based on the above, the heuristic solution of subproblems implemented by the MLD-BD(p,h) strategy is a reasonable assumption for this case study. Note that this heuristic may be invalid in situations where the ordering of tasks has a direct impact on process dynamics, e.g., multiproduct continuous processes where binary variables are used to decide the ordering of products in a wheel fashion[330]. To gain insights on the computational costs of the proposed MLD-BD, MLD-BD(p) and MLD-BD(p,h) strategies compared to the LD-BD, I have analyzed the CPU time requirements in the first iteration of each algorithm (see section H.5 in appendix H). The results show that the main improvements in CPU-time effort can be observed in the solution of subproblems, e.g., the MLD-BD(p) subproblem solution time was reduced by 32% when compared to the MLD-BD subproblem solution time.

As shown in Table 8.2, a naïve execution of the GBD strategy without feasibility cuts results in failure of the algorithm, whose first MIP master problem leads to an infeasible NLP subproblem in its second iteration after 150 seconds. As a result, GBD is only able to identify one solution whose improved objective function of 890.38$ is the result of an economic optimization of the control actions of the SIA solution with scheduling variables fixed. When implementing feasibility cuts, GBD(f) avoided the premature convergence failure; however, it could not find another feasible subproblem before reaching the time limit. Thus, although the feasibility cuts in GBD(f) allowed to close the optimality gap from 223% to 204%, GBD(f) was unable to improve the objective function after 310 iterations. The additional constraints included in the master problem for GBD(f,c) allowed to find new

feasible subproblems, and an incumbent solution with an objective of 1129.97$. These additional constraints imposed over variable processing times allowed to exclude processing time combinations that may be feasible from the scheduling viewpoint but infeasible from the control viewpoint, thus reducing the size of the search space explored by the GBD(f,c) algorithm. Nonetheless, GBD(f,c) reached the time limit, finding a solution with an absolute optimality gap of 21%. Note that none of the MLD-BD runs required the additional constraints implemented in GBD(f,c). Overall, the results in Table 8.2 show that none of the GBD strategies considered for this case study showed a better performance than the MLD-BD algorithm and its variants, e.g., GBD(f,c) reached the time limit to converge to a solution with an objective function that is still 1.6% lower than the best MLD-BD solution. In addition, the direct solution of the monolithic optimization problem using DICOPT with the default configuration and from the same SIA initialization was unable to converge to a feasible solution after reaching the maximum time limit (86400 seconds).

According to Table 8.2, the addition of the cut-off technique in MLD-BD(p,h,co) only allowed to slightly decrease the total computational time slightly by 1.2% with respect to MLD-BD(p,h), due to the short scheduling horizon used, which does not represent a limiting factor in the solution of MIP subproblems. The results in Table 8.3 illustrate the advantages of implementing a cut-off strategy in addition to the pruning and heuristic strategies to facilitate the solution of subproblems when considering a longer scheduling horizon η of 100800 seconds. The statistics of this problem are available in section H.4 of appendix H; this show that the case study with $\eta = 100800$ seconds has almost twice the number of binary variables than the case study with $\eta = 50400$ seconds, thus increasing the complexity of the MIP subproblems. In this case with $\eta = 100800$ seconds, the MLD-BD(p,h) was again initialized from a SIA solution, which in this case required 2178 seconds due to the size increase in the MIP scheduling part of the problem. The MLD-BD(p,h) was allowed to run without imposing a time limit until completion, resulting in a solution with an objective function of 3781.09$, thus showing an economic improvement of 47% with respect to the SIA initialization. However, the computational time increased by three orders of magnitude when compared to case when the MLD-BD(p,h) was solved with $\eta = 50400$ seconds (see Table 8.2). This increase in computational time is attributed to larger MIP subproblems requiring more iterations to close their optimality gap. The implementation of the MLD-BD(p,h,co) strategy helped to address this issue by prematurely stopping MIP subproblems based on the incumbent objective function f^{best}, as explained in section 8.2.3.2. Despite requiring more iterations than the MLD-BD(p,h) strategy, MLD-BD(p,h,co) reduced the

required computational time by two orders of magnitude from 238659 seconds to 6044 seconds, while attaining the same optimal objective of 3781.09$. This result illustrates how the proposed pruning strategy coupled with the approximated solution of subproblems is not only useful to simplify the solution of MINLP subproblems into a single MIP or a sequence of MIPs and NLPs, but also to facilitate the solution of the MIPs involved.

Table 8.3. Performance of MLD-BD solution strategies (case study 1, $\eta = 100{,}800\ s$)

	SIA (initialization)	MLD-BD(p,h)	MLD-BD(p,h,co)
Profit objective $(-f)$ [$]	2570.09	3781.09	3781.09
Fixed costs (c^{fix}) [$]	1510.00	1510.00	1510.00
Variable costs (c^{var_1}) [$]	1520.00	1520.00	1520.00
Dynamic costs (c^{var_2}) [$]	1929.91	718.91	718.91
Material costs (c^{mat}) [$]	4550.00	4550.00	4550.00
Revenue (rev) [$]	12080.00	12080.00	12080.00
CPU time [s]	2178	238659	6044
Iterations	-	45	61

The Gantt chart of the optimal solution found by the MLD-BD(p,h,co) strategy is shown in Figure 8.3A, which is compared against the SIA schedule shown in Figure 8.3B. As shown in Table 8.3, the only cost-related term that changed between the SIA and MLD-BD(p,h,co) solutions is the dynamic cost c^{var_2}, which decreased by 63% when using the latter approach. The reduction in the objective function is thus attributed to a more economic manipulation of the heating and cooling flows in reactions $R1$, $R2$ and $R3$, e.g., some of these control profiles are shown in section H.5 of appendix H. In addition, the SSDO approach not only affects the continuous variables of the SIA solution, but also their corresponding discrete variables. In this case, the processing time needed to execute $R2$ in Rs is larger in the MLD-BD(p,h,co) solution, which causes a change in the order tasks must be executed. For example, consider the areas in Figure 8.3A and Figure 8.3B surrounded by red dashed lines. Reactions $R2$ (blue block) and $R3$ (orange block) must be executed before the separation task Sep (yellow block), which starts at the same time in both Figure 8.3A and Figure 8.3B. However, due to the lengthier task $S2$, the second execution of task $S2$ in Rs was switched with the first execution of $R3$ in Rs to ensure that there is enough material from $R3$ to execute the separation process. Additionally, the execution of task $Pack1$ (purple block) that is a direct product of $R2$ is also delayed because of the new execution order of tasks in Rs. Note that despite the differences in processing times, sequencing of tasks, and

control actions in the reactors, batching variables $\boldsymbol{n}$ that define how many times each task is executed in each reactor did not change between the SIA and MLD-BD(p,h,co) solutions. A case study where the scheduling and dynamic optimization interactions affect batching variables $\boldsymbol{n}$ is presented next.

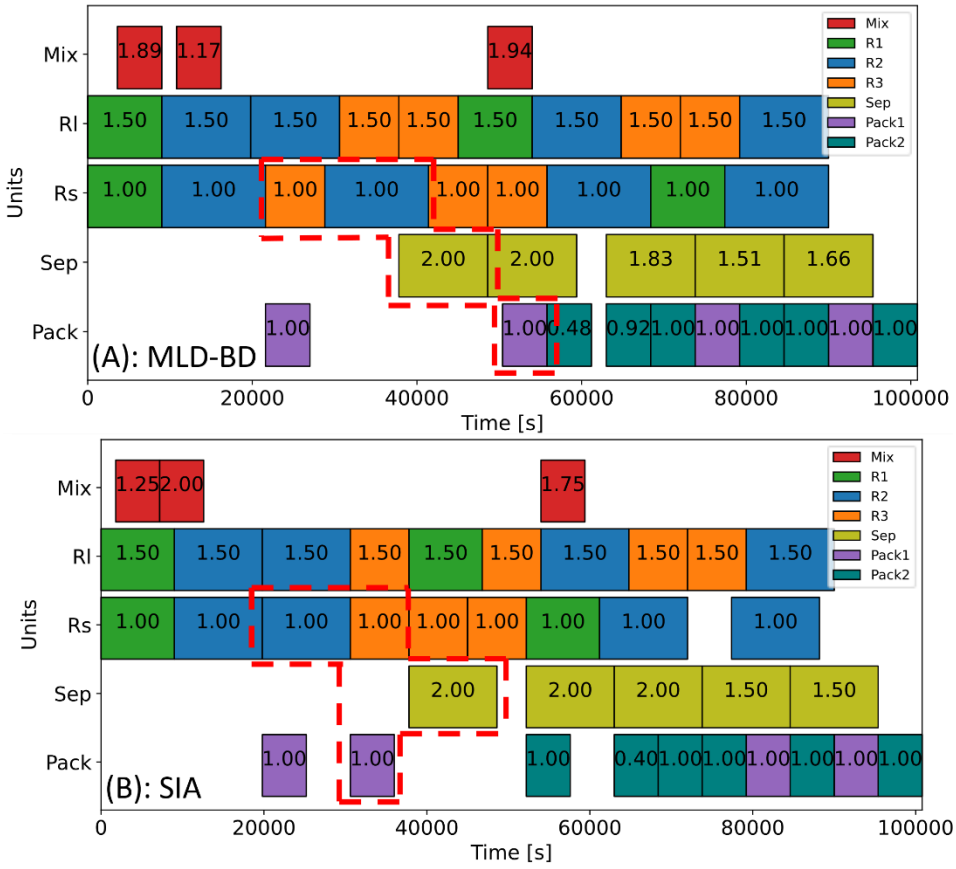


Figure 8.3. MLD-BD(p,h,co) (A) and SIA (B) Gantt charts (case study 1, $\eta = 100800\ s$).

8.3.2 Case study 2

This case study optimizes the Kondili et al.[360] STN shown in Figure 8.2B. In addition to considering a different STN, one key difference with respect to the first case study is the inclusion of dynamic models for both reaction and separation units. Semi-batch reaction task $T2$ is modeled using transient mass and energy balances to account for the interactions between variable processing time, batch sizes and control actions, which include the flowrate of a reactant that is dosed to the reaction, heating and cooling utilities. Differential equations are also used to model the dynamic behavior of the batch separation task ($T5$), which is a feature that is often neglected in network SSDO formulations. This distillation stage is

modeled as a pseudo-binary distillation system with stage-by-stage differential mass balance equations and a vapor-liquid equilibrium relation based on the relative volatility of the mixture. The remaining tasks are assumed to have fixed processing times and batch sizes that vary between lower and upper bounds. The raw materials of the process are represented through states $S1$, $S2$, and $S3$, while $S8$ and $S9$ are the products that generate revenue. A scheduling horizon η of 18000 seconds is considered for this case study. Statistics related to the number of variables and constraints for this problem are available in section H.4 of appendix H. Further details about the problem formulation and its reformulation with external variables are also provided in section H.4 of appendix H.

Based on the results of case study 1, a suitable approach to address this problem is the MLD-BD(p,h) strategy. To highlight the benefits of SSDO, the MLD-BD(p,h) strategy was compared against two heuristic scheduling and dynamic optimization approaches, i.e., SIA, and an iterative initialization approach (IIA). The IIA considers the fact that external variables $\boldsymbol{e}$ define the upper bounds of processing times in the network SSDO formulation considered in this work; hence, they play a major role in the feasibility/infeasibility of a candidate solution, e.g., a lengthy task may not be accommodated within an infeasible scheduling horizon leading to infeasibilities, while short processing times may saturate control actions leading to product quality requirements that may not be satisfied. Hence, the IIA starts with external variables $\boldsymbol{e}$ at their lower bounds and then iteratively increases the value of these variables with increments of one unit. For each trial value of $\boldsymbol{e}$, the IIA sequentially optimizes scheduling decisions through MIP optimization, followed by an NLP dynamic optimization step with scheduling variables fixed. The procedure stops when feasibility is attained for both scheduling and dynamic variables.

Table 8.4 shows information related to the objective function and solution time for each of the solution strategies considered. Compared to the first case study, the longer computational times shown in Table 8.4 are attributed mainly to the addition of the nonlinear dynamic distillation model, which complicates the NLP subproblems of the different solution approaches, e.g., the first two IIA iterations required 4919 seconds, while the same two iterations required 595 seconds if the dynamic distillation model was ignored, i.e., a reduction in one order of magnitude. The MLD-BD(p,h) strategy required 13 iterations, two of which were needed to detect a feasible solution after 6284 seconds using the feasibility strategy discussed in section S3 of the supplementary material. As shown in Table 8.4, the SIA solution shows the lowest profit of 372.73$, followed by the IIA (983.12$) and the MLD-BD (1738.25$). Note that MLD-BD(p,h) can also be initialized from SIA and IIA feasible solutions. On

the one hand, the MLD-BD(p,h) strategy converged to a solution with a profit of 415.18$ after 15031 seconds of execution, when initialized from SIA. This short processing time and marginal improvement in the objective function with respect to the SIA initialization is attributed to the fact that the complicating variables values from the SIA solution where identified as locally optimal by MLD-BD(p,h) strategy. On the other hand, MLD-BD(p,h) converged to the same solution when initialized from an infeasible point and the IIA initialization, after 57080 seconds and 59539 seconds, respectively. The optimization runs using MLD-BD(p,h) (from IIA initialization) and MLD-BD(p,h) (from infeasible) are CPU-time intensive compared to MLD-BD(p,h) (from SIA initialization), because the complicating variables in MLD-BD(p,h) (from IIA initialization) and MLD-BD(p,h) (from infeasible) do not stagnate at the first feasible solution the algorithm detects, e.g., while MLD-BD(p,h) (from SIA initialization) found a local optimum in the first neighborhood of complicating variables it evaluated, MLD-BD(p,h) (from IIA initialization) and MLD-BD(p,h) (from infeasible) explored 9 and 8 neighborhoods of complicating variables before stopping, respectively, thus making a more exhaustive search. Despite the superior performance showed by the IIA initialization with respect to the SIA initialization, the monolithic solution of this problem with DICOPT was unable to identify a feasible solution when starting from the IIA initialization after 259200 seconds of execution.

Table 8.4. Comparison of the MLD-BD (p,h) approach against the SIA and IIA (case study 2)

	SIA	IIA	MLD-BD(p,h)
Profit objective $(-f)$ [$]	372.73	983.12	1738.25
Fixed costs (c^{fix}) [$]	195.00	155.00	185.00
Variable costs (c^{var_1}) [$]	480.00	370.00	460.00
Dynamic costs (c^{var_2}) [$]	52.27	25.21	41.75
Material costs (c^{mat}) [$]	1300.00	866.67	975.00
Revenue (rev) [$]	2400.00	2400.00	3400.00
CPU time [s]	689	15924	57080
Iterations	-	7	13

Gantt charts for this case study are shown in Figure 8.4A, Figure 8.4B and Figure 8.4C for the SIA, IIA and MLD-BD(p,h), respectively. When comparing the SIA and IIA solutions, an excessive material cost in the SIA solution is the key factor that contributes the most to the difference in profit, representing 71% of the total difference in profits between the SIA and IIA objective functions. The excessive material cost in the SIA solution is needed to compensate the absence of reaction task $T4$ and separation task $S5$ to generate product $S9$, which has a revenue that is 100 $\$/m^3$ higher than product $S8$, e.g., the

separation unit $U4$ is being unused and task $T4$ is never executed, as shown in Figure 8.4A. Thus, the SIA schedule compensates the absence of the more expensive product $S9$ in its schedule by producing a larger amount of the least expensive product $S8$, leading to higher material costs with the same revenue than the IIA solution. The absence of separation task in the SIA solution is attributed to the sequential nature of the SIA. Specifically, the first SIA step returned a processing time of 10260 seconds for task $T5$ in unit $U4$. Given this processing time, the second SIA step (i.e., optimal scheduling) determined it to be more cost-effective to omit the execution of this lengthy task, rather than incorporating it into the schedule. As illustrated in Figure 8.4B, the IIA approach predicted a shorter processing time for $T5$ in unit $U4$ of 4644 seconds, making it possible to accommodate tasks $T4$ and $T5$ while returning a higher profit.

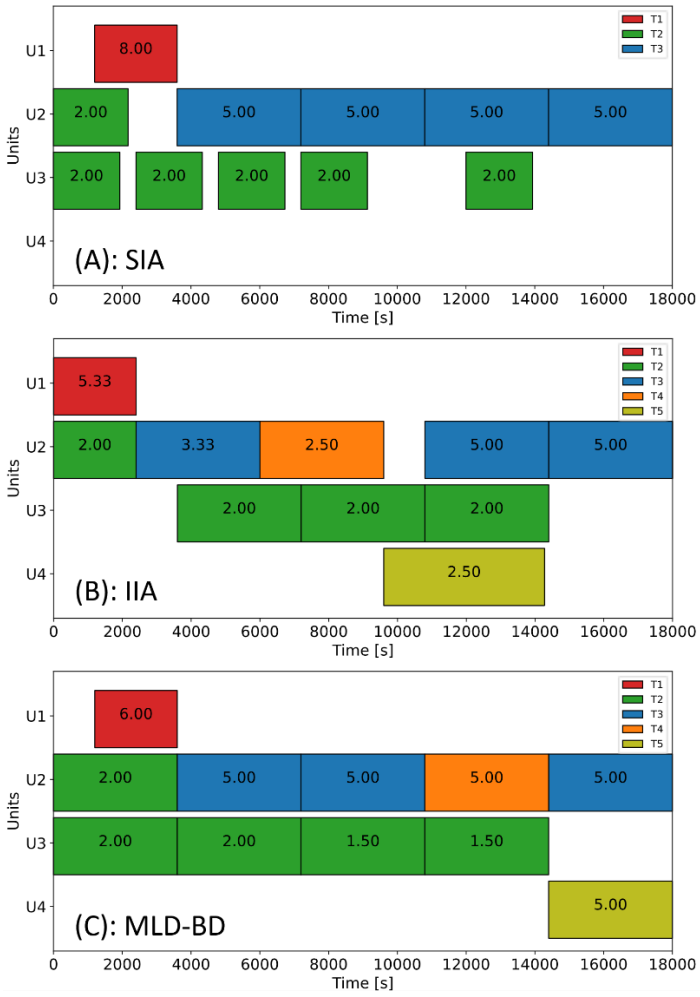


Figure 8.4. SIA (A), IIA (B) and MLD-BD (p,h) (C) Gantt charts (case study 2).

Despite the advantage of using an optimization-based iterative approach, note that the IIA strategy is still a heuristic designed to find a feasible schedule. When comparing the IIA and MLD-BD (p,h) schedules in Figure 8.4B and Figure 8.4C, it is possible to observe that the IIA underestimated the processing time of task $T2$ in unit $U2$, and overestimated the processing time of $T5$ in $U4$. In this case, the slightly shorter processing time of $T5$ in $U4$ of 3600 seconds identified by the MLD-BD(p,h) allowed to accommodate a new batch of reaction $T2$ in unit $U3$, as shown in Figure 8.4C. As a result, MLD-BD(p,h) incurred extra costs of 244.87$ to boost its production with respect to the IIA solution, leading to a 1000$ increase in revenue. For example, the amount of material processed in the distillation column was doubled to increase the production of material in $S9$. Similarly, the dynamic costs of MLD-BD(p,h) almost doubled with respect to the IIA solution. The differences in the schedules and costs between the IIA and the MLD-BD(p,h) solutions are also accompanied with changes in the operation policy of dynamic tasks such as the distillation column (task $T5$). Figure 8.5 shows the transient evolution of the mole fraction of the desired product in the distillate flow and the column's receiver, as well as the optimal control actions for IIA and MLD-BD(p,h). In both cases, the product target of 95% mol/mol of the desired product in the receiver is achieved at the end of the operation (see Figure 8.5A and Figure 8.5B). Nevertheless, the sudden drop in distillate flowrate occurring around 66% (i.e., after 3060 seconds for the IIA in Figure 8.5C) and 90% (i.e., after 3240 seconds for the MLD-BD(p,h) Figure 8.5D) of the total batch completion time suggests that there are differences in the control strategies. On the one hand, the IIA solution focuses on keeping low reflux and boil-up rates to sustain a low energy consumption throughout 4644 seconds of operation. For example, the reflux rate goes to 0 m^3/s, while the boil-up rate is gradually reaching 0 m^3/s after 3060 seconds of operation (see Figure 5C). On the other hand, the distillation column in MLD-BD(p,h) need to process a larger amount of material (i.e., 5 m^3 instead of 2.5 m^3) in a shorter period of time (3600 seconds). Thus, higher reflux and boil-up rates are sustained for a relatively longer period of time in the MLD-BD(p,h) solution, resulting in a 93% increase in the operating cost of the distillation unit (see Figure 8.5D). This illustrates how SSDO solutions obtained with the proposed MLD-BD methodology may result in schedules that, despite suggesting operation changes that may be counterintuitive when analyzed locally, return more profitable schedules and control decisions for the entire plant.

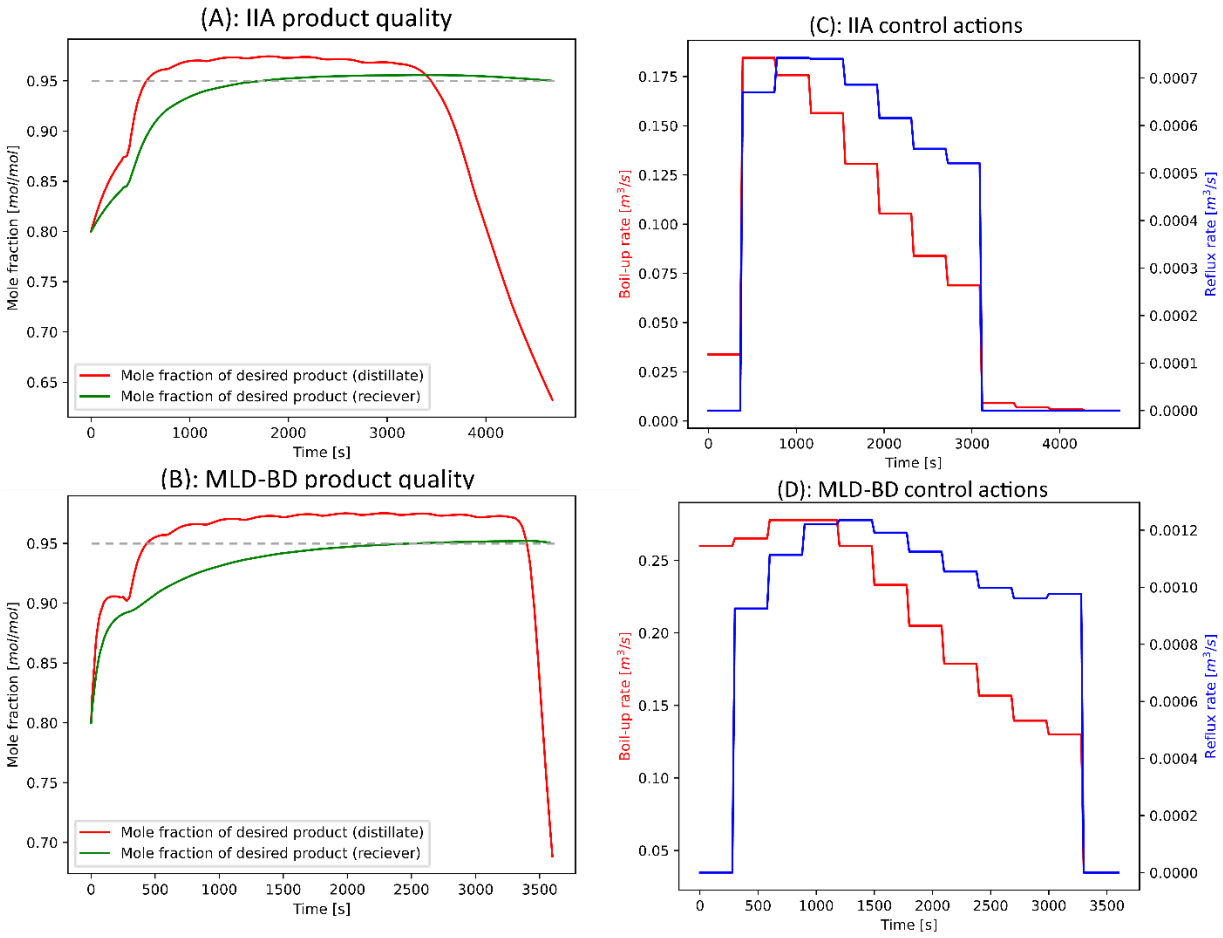


Figure 8.5. Distillate product quality and control actions for IIA (A and C); MLD-BD (B and D).

Chapter 9
Conclusions and Future Work

This thesis explored an emerging research field in the area of chemical engineering optimization: the solution of MINLP problems through algorithms based on DCA, with applications in optimal process design, optimal design and control and optimal scheduling and control. This thesis not only adapted the existing DCA-based strategies such as D-SDA and LD-SDA to consider new applications in decision-making integration, but also proposed algorithmic enhancements and new solution methods that rely on DCA concepts. New algorithms studied in this thesis include the enhanced D-SDA, the LD-BD algorithm, the MLD-BD algorithm, and the DSDA-VB within a hybrid deterministic-stochastic optimization algorithm. The enhanced D-SDA incorporates a pruning strategy within D-SDA that makes it suitable for integrated design and control, which is a feature that had not been studied before. LD-BD brings novelty by extending LBBD to nonlinear GDP optimization, which has not been discussed in the existing literature. MLD-BD builds upon LD-BD and brings novelty by introducing a multicut approach and a pruning strategy that makes this algorithm suitable for integrated scheduling and dynamic optimization. DSDA-VB introduces a variable bounding strategy that makes D-SDA suitable for simulation-based hybrid deterministic-stochastic optimization. To the author's knowledge, this is the first thesis that studies the algorithmic methods mentioned above, making this a relevant contribution to the field of deterministic MINLP optimization and integrated decision-making.

In addition to the algorithmic methods covered in this research, this thesis also brings novelty by studying challenging MINLP problems that have not been widely studied in the literature including the deterministic rate-based optimal design of catalytic distillation columns; the comparison of different logic-based algorithms for the direct solution of GDP superstructure problems; the optimal design of process flowsheets through parallel-hybrid deterministic-stochastic optimization; the deterministic simultaneous design and control of multitask (e.g., multiproduct) catalytic distillation columns, and the simultaneous discrete-time scheduling and dynamic optimization of network batch plants with variable processing times. The key conclusions from the different chapters of this thesis are outlined as follows:

While most process intensification systems are designed using equilibrium-based models, chapter 3 shows that neglecting simultaneous reaction kinetics, mass and energy transfer phenomena taking place between the interface and bulk vapor and liquid phases can have implications at the macroscale, thus affecting the process economics. For instance, process designs identified as optimal using equilibrium-

based models have been shown to be invalid since they cannot achieve the design goals specified for those systems when rate-based models (which in principle resemble a more realistic operation) are considered in the analysis. Hence, rate-based models featuring multiscale and multiphase phenomena are crucial for the optimal synthesis and design of intensified systems such as catalytic distillation columns. Generic MINLP solvers often face challenges when solving those problems due to the large number of combinatorial decisions that need to be considered, combined with the process's complex nonlinear behavior. In contrast, the D-SDA can achieve novel, economically attractive optimal process designs for catalytic distillation columns using rate-based models.

Based on chapter 4, the solutions obtained from LD-BD have proven local optimality characteristics based on DCA, which is a distinctive feature of this method. Three case studies including an illustrative mathematical test problem (case study 1) the design of a distillation column (case study 2) and a reactor network (case study 2) were used to demonstrate the key features of this method. Specifically, LD-BD has enhanced search capabilities that may allow the algorithm to escape from local optima, as observed in case study 1. Another feature of LD-BD is its independence of the initialization of discrete decisions when solving a non-convex distillation design problem. Such behavior was not observed for traditional local logic-based and MINLP solvers (see case study 2). In addition, the warm-start capabilities of the algorithm to increase the chances of converging to the global optimum were tested for the reactor network design problem in case study 3.

The parallel-hybrid approach proposed in chapter 5 combines the advantages of a pure SM with the efficiency of the DSDA-VB, resulting in a practical and computationally efficient algorithm. Two case studies involving intensified distillation units were considered to test the performance of this hybrid algorithm. The results show that the proposed hybrid algorithm quickly identifies a good quality local solution in the early stages of its execution. After that, the hybrid method relies on the exploratory capabilities of the SM method to identify new feasible solutions that serve as starting points to find better local optima. Overall, the parallel hybridization of the DSDA-VB and an SM such as DETL results in an enhanced convergence performance of both algorithms toward improved solutions, with respect to the naïve execution of the SM. In contrast to traditional hybrid approaches, the discrete and continuous variables obtained from the present approach satisfy local optimality.

An efficient and systematic methodology to optimize the total annualized cost of continuous multitask CD columns with dynamic transitions was proposed in chapter 6, based on an enhanced

version of the D-SDA that incorporates a pruning strategy for MIDO problems. As shown in the ETBE case study, if a judicious initialization procedure based on steady state information is applied, then this enhanced D-SDA can converge in a few iterations to an economically optimal solution. To do so, steady state optimizations can be solved first with the conventional D-SDA to trim the feasible region of the problem and discard designs that cannot fulfill the desired dynamic operation requirements. Consequently, the solution of this optimization problem becomes computationally tractable.

A modeling and solution framework for the solution of SSDO problems with variable processing times was introduced in chapter 7. A D-SDA was proposed to solve the resulting problem efficiently, by taking advantage of the ordered nature of batching variables and processing times that emerged in the SSDO formulation. A first case study involving a series of reactions that can be performed in two available semi-batch reactor within the Kondili STN shows that the decomposition performed by the D-SDA alleviates the computational difficulties faced when using the solver DICOPT alone, e.g., difficulties at closing the optimality gap due to the infeasibilities generated when considering variable processing times. A second case study involving three reaction tasks that can be executed in two available reactors shows that the D-SDA achieved more profitable solutions than sequential scheduling and dynamic optimization approaches, thanks to its rigorous exploration of the search space of variable processing times. In both case studies, the exploration performed by the D-SDA returned a locally optimal solution to the problem better than those identified by DICOPT, in shorter computational times.

The MLD-BD technique introduced in chapter 8 incorporates a pruning strategy that avoids the exploration of schedules whose process dynamics may not be economically attractive, a heuristic to simplify the solution of MINLP subproblems as a sequence of MIP and NLP problems, and a cut-off technique designed to speed up the solution of the MIP scheduling problems involved. Two case studies illustrate that the proposed improvements (e.g., multicut and pruning strategy) help to reduce the computational time by orders of magnitude, and that the MLD-BD technique may identify better solutions than GBD in shorter computational times. Also, heuristic strategies considered to address scheduling and dynamic optimization problems not only generate suboptimal control-related decisions, but they also may negatively impact scheduling-related decisions that may lead to inaccurate sequencing and timing of tasks, or suboptimal batching variables and unnecessary shutdowns of unit operations. In particular, the second case study illustrates the advantages of incorporating the dynamic model of distillation units, which are often treated as ideal separators when testing SSDO strategies.

9.1 Recommendations for future work

The insights gained through this thesis have demonstrated an important number of applications for the proposed methodologies; however, several potential improvements and applications can be considered as part of future work. The mayor recommendations derived from this research are as follows:

Developing robust strategies for the optimal design and operation of CD columns under uncertainty using the proposed rate-based model. Also, the proposed rate-based model can be expanded to consider nonideal mixing, intraparticle diffusion limitations, and the transient operation of the system.

Test the performance of LD-BD with larger superstructure design problems involving more discrete decisions, e.g., studying how does the method scale as the problem size increases, and studying the effect of using alternative neighborhoods and reformulations. Future research directions also include developing more rigorous warm-start strategies for the algorithm.

Improve the proposed parallel-hybrid algorithm to handle multi-objective, and multimodal objective functions. The algorithm must also be expanded to consider decisions that typically appear in process synthesis problems, such as deciding between two separation or reaction technologies, e.g., molecular sieve membrane separation against distillation.

The computational effort of D-SDA techniques can be further distributed into several processors that may work independently at different stages. For instance, multiple neighbors can be solved simultaneously if enough processors are available. Furthermore, the optimality verifications used by the enhanced D-SDA could also be adapted to consider multiple objectives related to environmental, safety and control concerns.

Future work also includes extending the proposed SSDO framework to consider disturbances, parameter uncertainty, and the closed-loop scheduling and control of chemical processes, as well as improving the computational performance of the proposed D-SDA strategy when dealing with MINLP subproblems. Future work is also needed to adapt and study the performance of the proposed modeling and solution approach with other problem types, e.g., mixed batch-continuous processes.

Regarding MLD-BD, Further research is needed to provide global optimality guarantees, and to generalize the feasibility LBBD step of the algorithm to consider more general scenarios. Lastly, research is needed to enable the efficient integration of three decision layers (e.g., planning, scheduling and control) using the algorithms introduced through this thesis.

Letters of Copyright Permissions

Chapter 3

Reprinted from publication Liñán DA, Bernal DE, Gómez JM, Ricardez-Sandoval LA. Optimal synthesis and design of catalytic distillation columns: A rate-based modeling approach. *Chemical Engineering Science*. 2021;231:116294. doi: https://doi.org/10.1016/j.ces.2020.116294. Copyright 2020, permission is not required from Elsevier (see more details below).



Chapter 4

JOHN WILEY AND SONS LICENSE
TERMS AND CONDITIONS
Jul 29, 2024

This Agreement between David A. Liñán ("You") and John Wiley and Sons ("John Wiley and Sons") consists of your license details and the terms and conditions provided by John Wiley and Sons and Copyright Clearance Center.

License Number	5838331181103
License date	Jul 29, 2024
Licensed Content Publisher	John Wiley and Sons
Licensed Content Publication	AIChE Journal
Licensed Content Title	A Benders decomposition framework for the optimization of disjunctive superstructures with ordered discrete decisions
Licensed Content Author	Luis A. Ricardez-Sandoval, David A. Liñán
Licensed Content Date	Jan 11, 2023
Licensed Content Volume	69
Licensed Content Issue	5
Licensed Content Pages	16
Type of use	Dissertation/Thesis
Requestor type	Author of this Wiley article
Format	Electronic
Portion	Full article
Will you be translating?	No
Title of new work	Optimal Integrated Decision-Making in Chemical Engineering: MINLP Advances and Applications
Institution name	University of Waterloo
Expected presentation date	Sep 2024
Order reference number	Chapter 4
The Requesting Person / Organization to Appear on the License	David A. Liñán
Requestor Location	Mr. David Liñán 81802-158 king street north Waterloo, ON N2J0E5 Canada Attn: Mr. David Liñán
Publisher Tax ID	EU826007151

Total 0.00 CAD

Terms and Conditions

TERMS AND CONDITIONS

This copyrighted material is owned by or exclusively licensed to John Wiley & Sons, Inc. or one of its group companies (each a"Wiley Company") or handled on behalf of a society with which a Wiley Company has exclusive publishing rights in relation to a particular work (collectively "WILEY"). By clicking "accept" in connection with completing this licensing transaction, you agree that the following terms and conditions apply to this transaction (along with the billing and payment terms and conditions established by the Copyright Clearance Center Inc., ("CCC's Billing and Payment terms and conditions"), at the time that you opened your RightsLink account (these are available at any time at http://myaccount.copyright.com).

Terms and Conditions

- The materials you have requested permission to reproduce or reuse (the "Wiley Materials") are protected by copyright.

- You are hereby granted a personal, non-exclusive, non-sub licensable (on a stand-alone basis), non-transferable, worldwide, limited license to reproduce the Wiley Materials for the purpose specified in the licensing process. This license, **and any CONTENT (PDF or image file) purchased as part of your order,** is for a one-time use only and limited to any maximum distribution number specified in the license. The first instance of republication or reuse granted by this license must be completed within two years of the date of the grant of this license (although copies prepared before the end date may be distributed thereafter). The Wiley Materials shall not be used in any other manner or for any other purpose, beyond what is granted in the license. Permission is granted subject to an appropriate acknowledgement given to the author, title of the material/book/journal and the publisher. You shall also duplicate the copyright notice that appears in the Wiley publication in your use of the Wiley Material. Permission is also granted on the understanding that nowhere in the text is a previously published source acknowledged for all or part of this Wiley Material. Any third party content is expressly excluded from this permission.

- With respect to the Wiley Materials, all rights are reserved. Except as expressly granted by the terms of the license, no part of the Wiley Materials may be copied, modified, adapted (except for minor reformatting required by the new Publication), translated, reproduced, transferred or distributed, in any form or by any means, and no derivative works may be made based on the Wiley Materials without the prior permission of the respective copyright owner.**For STM Signatory Publishers clearing permission under the terms of the STM Permissions Guidelines only, the terms of the license are extended to include subsequent editions and for editions in other languages, provided such editions are for the work as a whole in situ and does not involve the separate exploitation of the permitted figures or extracts,** You may not alter, remove or suppress in any manner any copyright, trademark or other notices displayed by the Wiley Materials.

You may not license, rent, sell, loan, lease, pledge, offer as security, transfer or assign the Wiley Materials on a stand-alone basis, or any of the rights granted to you hereunder to any other person.

- The Wiley Materials and all of the intellectual property rights therein shall at all times remain the exclusive property of John Wiley & Sons Inc, the Wiley Companies, or their respective licensors, and your interest therein is only that of having possession of and the right to reproduce the Wiley Materials pursuant to Section 2 herein during the continuance of this Agreement. You agree that you own no right, title or interest in or to the Wiley Materials or any of the intellectual property rights therein. You shall have no rights hereunder other than the license as provided for above in Section 2. No right, license or interest to any trademark, trade name, service mark or other branding ("Marks") of WILEY or its licensors is granted hereunder, and you agree that you shall not assert any such right, license or interest with respect thereto

- NEITHER WILEY NOR ITS LICENSORS MAKES ANY WARRANTY OR REPRESENTATION OF ANY KIND TO YOU OR ANY THIRD PARTY, EXPRESS, IMPLIED OR STATUTORY, WITH RESPECT TO THE MATERIALS OR THE ACCURACY OF ANY INFORMATION CONTAINED IN THE MATERIALS, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTY OF MERCHANTABILITY, ACCURACY, SATISFACTORY QUALITY, FITNESS FOR A PARTICULAR PURPOSE, USABILITY, INTEGRATION OR NON-INFRINGEMENT AND ALL SUCH WARRANTIES ARE HEREBY EXCLUDED BY WILEY AND ITS LICENSORS AND WAIVED BY YOU.

- WILEY shall have the right to terminate this Agreement immediately upon breach of this Agreement by you.

- You shall indemnify, defend and hold harmless WILEY, its Licensors and their respective directors, officers, agents and employees, from and against any actual or threatened claims, demands, causes of action or proceedings arising from any breach of this Agreement by you.

- IN NO EVENT SHALL WILEY OR ITS LICENSORS BE LIABLE TO YOU OR ANY OTHER PARTY OR ANY OTHER PERSON OR ENTITY FOR ANY SPECIAL, CONSEQUENTIAL, INCIDENTAL, INDIRECT, EXEMPLARY OR PUNITIVE DAMAGES, HOWEVER CAUSED, ARISING OUT OF OR IN CONNECTION WITH THE DOWNLOADING, PROVISIONING, VIEWING OR USE OF THE MATERIALS REGARDLESS OF THE FORM OF ACTION, WHETHER FOR BREACH OF CONTRACT, BREACH OF WARRANTY, TORT, NEGLIGENCE, INFRINGEMENT OR OTHERWISE (INCLUDING, WITHOUT LIMITATION, DAMAGES BASED ON LOSS OF PROFITS, DATA, FILES, USE, BUSINESS OPPORTUNITY OR CLAIMS OF THIRD PARTIES), AND WHETHER OR NOT THE PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION SHALL APPLY

NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY PROVIDED HEREIN.

- Should any provision of this Agreement be held by a court of competent jurisdiction to be illegal, invalid, or unenforceable, that provision shall be deemed amended to achieve as nearly as possible the same economic effect as the original provision, and the legality, validity and enforceability of the remaining provisions of this Agreement shall not be affected or impaired thereby.

- The failure of either party to enforce any term or condition of this Agreement shall not constitute a waiver of either party's right to enforce each and every term and condition of this Agreement. No breach under this agreement shall be deemed waived or excused by either party unless such waiver or consent is in writing signed by the party granting such waiver or consent. The waiver by or consent of a party to a breach of any provision of this Agreement shall not operate or be construed as a waiver of or consent to any other or subsequent breach by such other party.

- This Agreement may not be assigned (including by operation of law or otherwise) by you without WILEY's prior written consent.

- Any fee required for this permission shall be non-refundable after thirty (30) days from receipt by the CCC.

- These terms and conditions together with CCC's Billing and Payment terms and conditions (which are incorporated herein) form the entire agreement between you and WILEY concerning this licensing transaction and (in the absence of fraud) supersedes all prior agreements and representations of the parties, oral or written. This Agreement may not be amended except in writing signed by both parties. This Agreement shall be binding upon and inure to the benefit of the parties' successors, legal representatives, and authorized assigns.

- In the event of any conflict between your obligations established by these terms and conditions and those established by CCC's Billing and Payment terms and conditions, these terms and conditions shall prevail.

- WILEY expressly reserves all rights not specifically granted in the combination of (i) the license details provided by you and accepted in the course of this licensing transaction, (ii) these terms and conditions and (iii) CCC's Billing and Payment terms and conditions.

- This Agreement will be void if the Type of Use, Format, Circulation, or Requestor Type was misrepresented during the licensing process.

- This Agreement shall be governed by and construed in accordance with the laws of the State of New York, USA, without regards to such state's conflict of law rules. Any legal action, suit or proceeding arising out of or relating to these Terms and Conditions or the breach thereof shall be instituted in a court of competent jurisdiction in New York County

in the State of New York in the United States of America and each party hereby consents and submits to the personal jurisdiction of such court, waives any objection to venue in such court and consents to service of process by registered or certified mail, return receipt requested, at the last known address of such party.

WILEY OPEN ACCESS TERMS AND CONDITIONS

Wiley Publishes Open Access Articles in fully Open Access Journals and in Subscription journals offering Online Open. Although most of the fully Open Access journals publish open access articles under the terms of the Creative Commons Attribution (CC BY) License only, the subscription journals and a few of the Open Access Journals offer a choice of Creative Commons Licenses. The license type is clearly identified on the article.

The Creative Commons Attribution License

The Creative Commons Attribution License (CC-BY) allows users to copy, distribute and transmit an article, adapt the article and make commercial use of the article. The CC-BY license permits commercial and non-

Creative Commons Attribution Non-Commercial License

The Creative Commons Attribution Non-Commercial (CC-BY-NC)License permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.(see below)

Creative Commons Attribution-Non-Commercial-NoDerivs License

The Creative Commons Attribution Non-Commercial-NoDerivs License (CC-BY-NC-ND) permits use, distribution and reproduction in any medium, provided the original work is properly cited, is not used for commercial purposes and no modifications or adaptations are made. (see below)

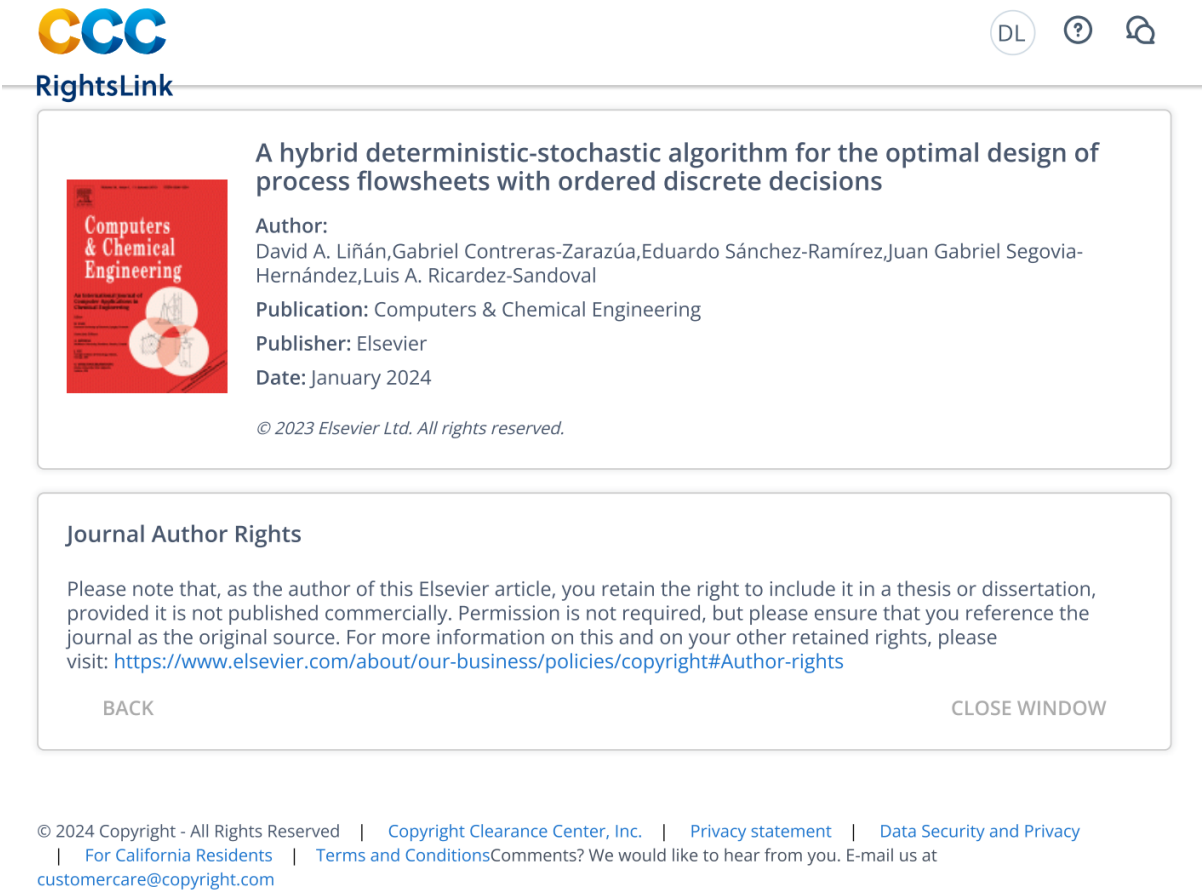
Use by commercial "for-profit" organizations

Use of Wiley Open Access articles for commercial, promotional, or marketing purposes requires further explicit permission from Wiley and will be subject to a fee.

Further details can be found on Wiley Online Library http://olabout.wiley.com/WileyCDA/Section/id-410895.html
Other Terms and Conditions:
v1.10 Last updated September 2015
Questions? customercare@copyright.com.

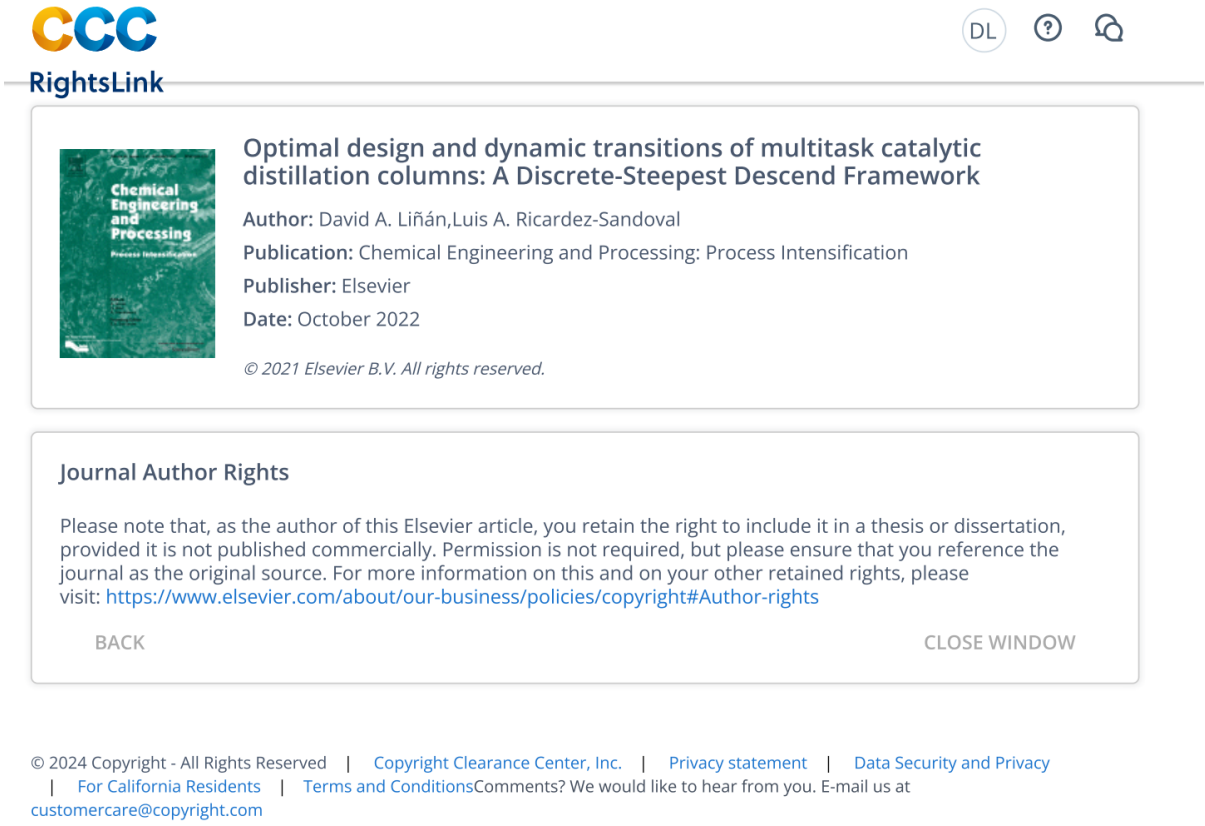
Chapter 5

Reprinted from publication Liñán DA, Contreras-Zarazúa G, Sánchez-Ramírez E, Segovia-Hernández JG, Ricardez-Sandoval LA. A hybrid deterministic-stochastic algorithm for the optimal design of process flowsheets with ordered discrete decisions. *Computers & Chemical Engineering*. 2024;180:108501. doi: https://doi.org/10.1016/j.compchemeng.2023.108501. Copyright 2023, permission is not required from Elsevier (see more details below).



Chapter 6

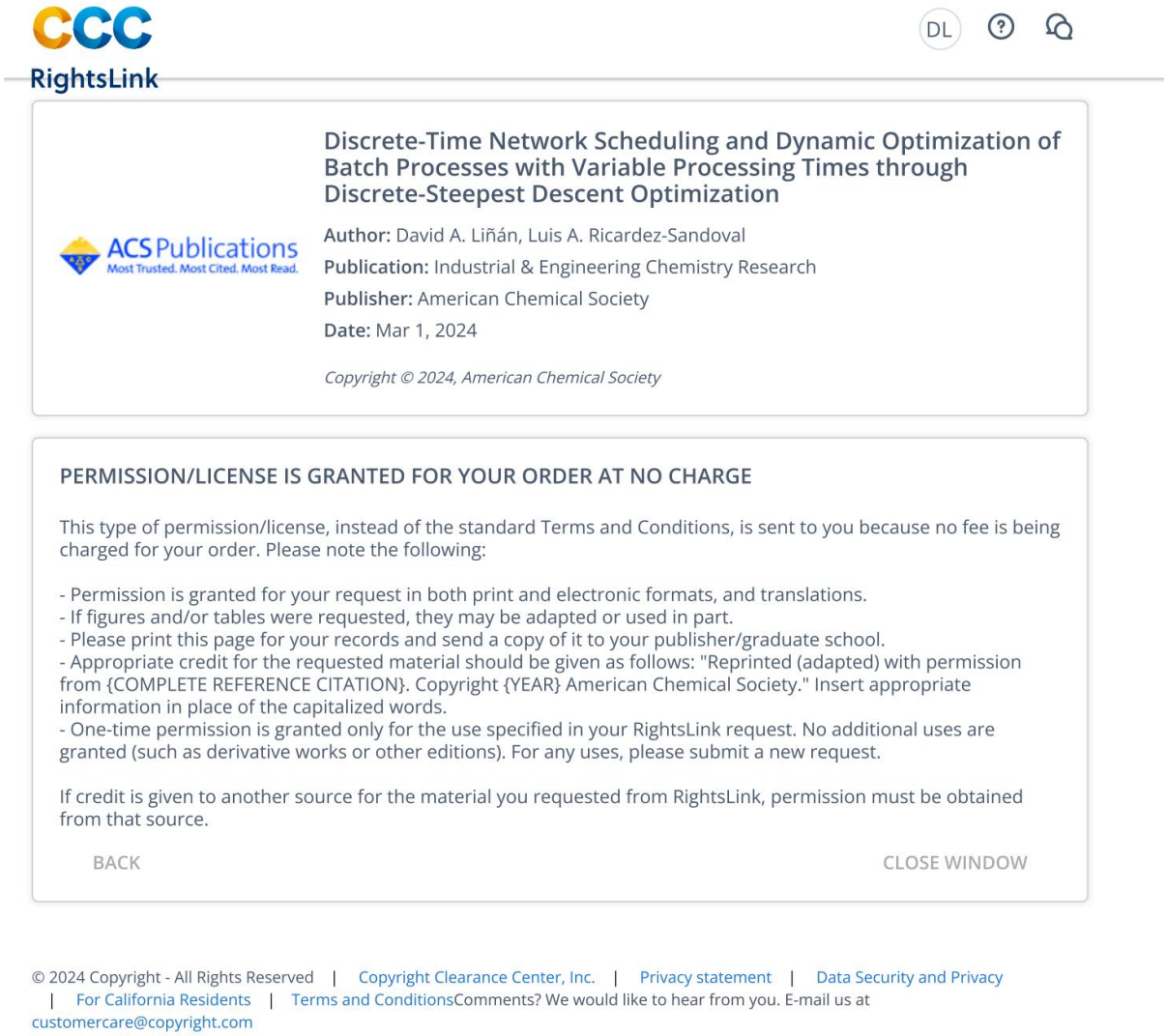
Reprinted from publication Liñán DA, Ricardez-Sandoval LA. Optimal design and dynamic transitions of multitask catalytic distillation columns: A Discrete-Steepest Descend Framework. *Chemical Engineering and Processing - Process Intensification*. 2022;180:108655. doi: https://doi.org/10.1016/j.cep.2021.108655. Copyright 2021, permission is not required from Elsevier (see more details below).



© 2024 Copyright - All Rights Reserved | Copyright Clearance Center, Inc. | Privacy statement | Data Security and Privacy | For California Residents | Terms and ConditionsComments? We would like to hear from you. E-mail us at customercare@copyright.com

Chapter 7

Adapted with permission from Liñán DA, Ricardez-Sandoval LA. Discrete-Time Network Scheduling and Dynamic Optimization of Batch Processes with Variable Processing Times through Discrete-Steepest Descent Optimization. *Ind Eng Chem Res*. 2024;63(10):4478-4495. doi: https://doi.org/10.1021/acs.iecr.3c03455. Copyright 2024 American Chemical Society.



Chapter 8

JOHN WILEY AND SONS LICENSE
TERMS AND CONDITIONS
Jul 29, 2024

This Agreement between David A. Liñán ("You") and John Wiley and Sons ("John Wiley and Sons") consists of your license details and the terms and conditions provided by John Wiley and Sons and Copyright Clearance Center.

License Number	5838331501198
License date	Jul 29, 2024
Licensed Content Publisher	John Wiley and Sons
Licensed Content Publication	AIChE Journal
Licensed Content Title	Multicut logic-based Benders decomposition for discrete-time scheduling and dynamic optimization of network batch plants
Licensed Content Author	David A. Liñán, Luis A. Ricardez-Sandoval
Licensed Content Date	May 22, 2024
Licensed Content Volume	0
Licensed Content Issue	0
Licensed Content Pages	18
Type of use	Dissertation/Thesis
Requestor type	Author of this Wiley article
Format	Electronic
Portion	Full article
Will you be translating?	No
Title of new work	Optimal Integrated Decision-Making in Chemical Engineering: MINLP Advances and Applications
Institution name	University of Waterloo
Expected presentation date	Sep 2024
Order reference number	Chapter 8
The Requesting Person / Organization to Appear on the License	David A. Liñán
Requestor Location	Mr. David Liñán 81802-158 king street north Waterloo, ON N2J0E5 Canada Attn: Mr. David Liñán

Publisher Tax ID	EU826007151
Total	0.00 CAD

Terms and Conditions

TERMS AND CONDITIONS

This copyrighted material is owned by or exclusively licensed to John Wiley & Sons, Inc. or one of its group companies (each a"Wiley Company") or handled on behalf of a society with which a Wiley Company has exclusive publishing rights in relation to a particular work (collectively "WILEY"). By clicking "accept" in connection with completing this licensing transaction, you agree that the following terms and conditions apply to this transaction (along with the billing and payment terms and conditions established by the Copyright Clearance Center Inc., ("CCC's Billing and Payment terms and conditions"), at the time that you opened your RightsLink account (these are available at any time at http://myaccount.copyright.com).

Terms and Conditions

- The materials you have requested permission to reproduce or reuse (the "Wiley Materials") are protected by copyright.

- You are hereby granted a personal, non-exclusive, non-sub licensable (on a stand-alone basis), non-transferable, worldwide, limited license to reproduce the Wiley Materials for the purpose specified in the licensing process. This license, **and any CONTENT (PDF or image file) purchased as part of your order,** is for a one-time use only and limited to any maximum distribution number specified in the license. The first instance of republication or reuse granted by this license must be completed within two years of the date of the grant of this license (although copies prepared before the end date may be distributed thereafter). The Wiley Materials shall not be used in any other manner or for any other purpose, beyond what is granted in the license. Permission is granted subject to an appropriate acknowledgement given to the author, title of the material/book/journal and the publisher. You shall also duplicate the copyright notice that appears in the Wiley publication in your use of the Wiley Material. Permission is also granted on the understanding that nowhere in the text is a previously published source acknowledged for all or part of this Wiley Material. Any third party content is expressly excluded from this permission.

- With respect to the Wiley Materials, all rights are reserved. Except as expressly granted by the terms of the license, no part of the Wiley Materials may be copied, modified, adapted (except for minor reformatting required by the new Publication), translated, reproduced, transferred or distributed, in any form or by any means, and no derivative works may be made based on the Wiley Materials without the prior permission of the respective copyright owner.**For STM Signatory Publishers clearing permission under the terms of the STM Permissions Guidelines only, the terms of the license are extended to include subsequent editions and for editions in other languages, provided such editions are for the work as a whole in situ and does not involve the separate exploitation of the permitted figures or extracts,** You may not alter, remove or suppress in any manner any copyright, trademark or other notices displayed by the Wiley Materials.

You may not license, rent, sell, loan, lease, pledge, offer as security, transfer or assign the Wiley Materials on a stand-alone basis, or any of the rights granted to you hereunder to any other person.

- The Wiley Materials and all of the intellectual property rights therein shall at all times remain the exclusive property of John Wiley & Sons Inc, the Wiley Companies, or their respective licensors, and your interest therein is only that of having possession of and the right to reproduce the Wiley Materials pursuant to Section 2 herein during the continuance of this Agreement. You agree that you own no right, title or interest in or to the Wiley Materials or any of the intellectual property rights therein. You shall have no rights hereunder other than the license as provided for above in Section 2. No right, license or interest to any trademark, trade name, service mark or other branding ("Marks") of WILEY or its licensors is granted hereunder, and you agree that you shall not assert any such right, license or interest with respect thereto

- NEITHER WILEY NOR ITS LICENSORS MAKES ANY WARRANTY OR REPRESENTATION OF ANY KIND TO YOU OR ANY THIRD PARTY, EXPRESS, IMPLIED OR STATUTORY, WITH RESPECT TO THE MATERIALS OR THE ACCURACY OF ANY INFORMATION CONTAINED IN THE MATERIALS, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTY OF MERCHANTABILITY, ACCURACY, SATISFACTORY QUALITY, FITNESS FOR A PARTICULAR PURPOSE, USABILITY, INTEGRATION OR NON-INFRINGEMENT AND ALL SUCH WARRANTIES ARE HEREBY EXCLUDED BY WILEY AND ITS LICENSORS AND WAIVED BY YOU.

- WILEY shall have the right to terminate this Agreement immediately upon breach of this Agreement by you.

- You shall indemnify, defend and hold harmless WILEY, its Licensors and their respective directors, officers, agents and employees, from and against any actual or threatened claims, demands, causes of action or proceedings arising from any breach of this Agreement by you.

- IN NO EVENT SHALL WILEY OR ITS LICENSORS BE LIABLE TO YOU OR ANY OTHER PARTY OR ANY OTHER PERSON OR ENTITY FOR ANY SPECIAL, CONSEQUENTIAL, INCIDENTAL, INDIRECT, EXEMPLARY OR PUNITIVE DAMAGES, HOWEVER CAUSED, ARISING OUT OF OR IN CONNECTION WITH THE DOWNLOADING, PROVISIONING, VIEWING OR USE OF THE MATERIALS REGARDLESS OF THE FORM OF ACTION, WHETHER FOR BREACH OF CONTRACT, BREACH OF WARRANTY, TORT, NEGLIGENCE, INFRINGEMENT OR OTHERWISE (INCLUDING, WITHOUT LIMITATION, DAMAGES BASED ON LOSS OF PROFITS, DATA, FILES, USE, BUSINESS OPPORTUNITY OR CLAIMS OF THIRD PARTIES), AND WHETHER OR NOT THE PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION SHALL APPLY

NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY PROVIDED HEREIN.

- Should any provision of this Agreement be held by a court of competent jurisdiction to be illegal, invalid, or unenforceable, that provision shall be deemed amended to achieve as nearly as possible the same economic effect as the original provision, and the legality, validity and enforceability of the remaining provisions of this Agreement shall not be affected or impaired thereby.

- The failure of either party to enforce any term or condition of this Agreement shall not constitute a waiver of either party's right to enforce each and every term and condition of this Agreement. No breach under this agreement shall be deemed waived or excused by either party unless such waiver or consent is in writing signed by the party granting such waiver or consent. The waiver by or consent of a party to a breach of any provision of this Agreement shall not operate or be construed as a waiver of or consent to any other or subsequent breach by such other party.

- This Agreement may not be assigned (including by operation of law or otherwise) by you without WILEY's prior written consent.

- Any fee required for this permission shall be non-refundable after thirty (30) days from receipt by the CCC.

- These terms and conditions together with CCC's Billing and Payment terms and conditions (which are incorporated herein) form the entire agreement between you and WILEY concerning this licensing transaction and (in the absence of fraud) supersedes all prior agreements and representations of the parties, oral or written. This Agreement may not be amended except in writing signed by both parties. This Agreement shall be binding upon and inure to the benefit of the parties' successors, legal representatives, and authorized assigns.

- In the event of any conflict between your obligations established by these terms and conditions and those established by CCC's Billing and Payment terms and conditions, these terms and conditions shall prevail.

- WILEY expressly reserves all rights not specifically granted in the combination of (i) the license details provided by you and accepted in the course of this licensing transaction, (ii) these terms and conditions and (iii) CCC's Billing and Payment terms and conditions.

- This Agreement will be void if the Type of Use, Format, Circulation, or Requestor Type was misrepresented during the licensing process.

- This Agreement shall be governed by and construed in accordance with the laws of the State of New York, USA, without regards to such state's conflict of law rules. Any legal action, suit or proceeding arising out of or relating to these Terms and Conditions or the breach thereof shall be instituted in a court of competent jurisdiction in New York County

in the State of New York in the United States of America and each party hereby consents and submits to the personal jurisdiction of such court, waives any objection to venue in such court and consents to service of process by registered or certified mail, return receipt requested, at the last known address of such party.

WILEY OPEN ACCESS TERMS AND CONDITIONS

Wiley Publishes Open Access Articles in fully Open Access Journals and in Subscription journals offering Online Open. Although most of the fully Open Access journals publish open access articles under the terms of the Creative Commons Attribution (CC BY) License only, the subscription journals and a few of the Open Access Journals offer a choice of Creative Commons Licenses. The license type is clearly identified on the article.

The Creative Commons Attribution License

The Creative Commons Attribution License (CC-BY) allows users to copy, distribute and transmit an article, adapt the article and make commercial use of the article. The CC-BY license permits commercial and non-

Creative Commons Attribution Non-Commercial License

The Creative Commons Attribution Non-Commercial (CC-BY-NC)License permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.(see below)

Creative Commons Attribution-Non-Commercial-NoDerivs License

The Creative Commons Attribution Non-Commercial-NoDerivs License (CC-BY-NC-ND) permits use, distribution and reproduction in any medium, provided the original work is properly cited, is not used for commercial purposes and no modifications or adaptations are made. (see below)

Use by commercial "for-profit" organizations

Use of Wiley Open Access articles for commercial, promotional, or marketing purposes requires further explicit permission from Wiley and will be subject to a fee.

Further details can be found on Wiley Online Library http://olabout.wiley.com/WileyCDA/Section/id-410895.html
Other Terms and Conditions:
v1.10 Last updated September 2015
Questions? customercare@copyright.com.

Appendix A

Reprinted from publication Liñán DA, Ricardez-Sandoval LA. A Discrete-Steepest Descent Framework for the Simultaneous Process and Control Design of Multigrade Reactive Distillation Columns. *IFAC-PapersOnLine*. 2022;55(7):370-375. doi: https://doi.org/10.1016/j.ifacol.2022.07.472. Copyright 2022 The authors. This is an open access article under the CC-BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/). More details are provided below (see https://www.sciencedirect.com/journal/ifac-papersonline/publish/guide-for-authors, https://www.ifac-control.org/publications/copyright-conditions).



Appendix B

Reprinted from publication Liñán DA, Reynoso-Donzelli S, Ricardez-Sandoval LA. Optimal scheduling and open-loop control of network batch processes under variable processing times using Generalized Benders Decomposition. *IEEE Xplore, American Control Conference (ACC)*. 2024, pp. 4466-4471, doi: https://doi.org/10.23919/ACC60939.2024.10644393. ACC papers are published on IEEE Xplore and are copyrighted by the American Automatic Control Council (AACC). More details are provided below (see https://a2c2.org/events/acc).



References

1. Rafiei M, Ricardez-Sandoval LA. New frontiers, challenges, and opportunities in integration of design and control for enterprise-wide sustainability. *Computers & Chemical Engineering*. 2020;132:106610. doi:10.1016/j.compchemeng.2019.106610

2. Burnak B, Diangelakis NA, Pistikopoulos EN. Towards the Grand Unification of Process Design, Scheduling, and Control—Utopia or Reality? *Processes*. 2019;7(7):461. doi:10.3390/pr7070461

3. Bernal DE, Ovalle D, Liñán DA, Ricardez-Sandoval LA, Gómez JM, Grossmann IE. Process Superstructure Optimization through Discrete Steepest Descent Optimization: a GDP Analysis and Applications in Process Intensification. In: Yamashita Y, Kano M, eds. *Computer Aided Chemical Engineering*. Vol 49. 14 International Symposium on Process Systems Engineering. Elsevier; 2022:1279-1284. doi:10.1016/B978-0-323-85159-6.50213-X

4. Liñán DA, Bernal DE, Ricardez-Sandoval LA, Gómez JM. Optimal design of superstructures for placing units and streams with multiple and ordered available locations. Part I: A new mathematical framework. *Computers & Chemical Engineering*. 2020;137:106794. doi:10.1016/j.compchemeng.2020.106794

5. Lin EYH, Bricker DL. Connecting special ordered inequalities and transformation and reformulation technique in multiple choice programming. *Computers & Operations Research*. 2002;29(10):1441-1446. doi:10.1016/S0305-0548(00)00104-0

6. Murota K. *Discrete Convex Analysis*. Society for Industrial and Applied Mathematics; 2003.

7. Liñán DA, Bernal DE, Gómez JM, Ricardez-Sandoval LA. Optimal synthesis and design of catalytic distillation columns: A rate-based modeling approach. *Chemical Engineering Science*. 2021;231:116294. doi:10.1016/j.ces.2020.116294

8. Rahmaniani R, Crainic TG, Gendreau M, Rei W. The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*. 2017;259(3):801-817. doi:10.1016/j.ejor.2016.12.005

9. Hooker JN, Ottosson G. Logic-based Benders decomposition. *Math Program, Ser A*. 2003;96(1):33-60. doi:10.1007/s10107-003-0375-9

10. Hooker JN. Logic-Based Benders Decomposition for Large-Scale Optimization. In: Velásquez-Bermúdez JM, Khakifirooz M, Fathi M, eds. *Large Scale Optimization in Supply Chains and Smart Manufacturing: Theory and Applications*. Springer Optimization and Its Applications. Springer International Publishing; 2019:1-26. doi:10.1007/978-3-030-22788-3_1

11. Drud AS. CONOPT—A Large-Scale GRG Code. *ORSA Journal on Computing*. 1994;6(2):207-216. doi:10.1287/ijoc.6.2.207

12. Liñán DA, Bernal DE, Ricardez-Sandoval LA, Gómez JM. Optimal design of superstructures for placing units and streams with multiple and ordered available locations. Part II: Rigorous design of catalytic distillation columns. *Computers & Chemical Engineering*. 2020;139:106845. doi:10.1016/j.compchemeng.2020.106845

13. Chen Q, Johnson ES, Bernal DE, et al. Pyomo.GDP: an ecosystem for logic based modeling and optimization development. *Optim Eng*. 2022;23(1):607-642. doi:10.1007/s11081-021-09601-7

14. Liñán DA, Ricardez-Sandoval L. Trends and perspectives in deterministic MINLP optimization for integrated planning, scheduling, control and design of chemical processes. *Manuscript under consideration in Reviews in Chemical Engineering (revce20240064)*. Published online 2024.

15. Liñán DA, Ricardez-Sandoval LA. A Benders decomposition framework for the optimization of disjunctive superstructures with ordered discrete decisions. *AIChE Journal*. 2023;69(5):e18008. doi:10.1002/aic.18008

16. Liñán DA, Contreras-Zarazúa G, Sánchez-Ramírez E, Segovia-Hernández JG, Ricardez-Sandoval LA. A hybrid deterministic-stochastic algorithm for the optimal design of process flowsheets with ordered discrete decisions. *Computers & Chemical Engineering*. 2024;180:108501. doi:10.1016/j.compchemeng.2023.108501

17. Liñán DA, Ricardez-Sandoval LA. Optimal design and dynamic transitions of multitask catalytic distillation columns: A Discrete-Steepest Descend Framework. *Chemical Engineering and Processing - Process Intensification*. 2022;180:108655. doi:10.1016/j.cep.2021.108655

18. Liñán DA, Ricardez-Sandoval LA. Discrete-Time Network Scheduling and Dynamic Optimization of Batch Processes with Variable Processing Times through Discrete-Steepest Descent Optimization. *Ind Eng Chem Res*. 2024;63(10):4478-4495. doi:10.1021/acs.iecr.3c03455

19. Liñán DA, Ricardez-Sandoval LA. Multicut logic-based Benders decomposition for discrete-time scheduling and dynamic optimization of network batch plants. *AIChE Journal*. Published online 2024:e18491. doi:10.1002/aic.18491

20. Liñán DA, Ricardez-Sandoval LA. A Discrete-Steepest Descent Framework for the Simultaneous Process and Control Design of Multigrade Reactive Distillation Columns. *IFAC-PapersOnLine*. 2022;55(7):370-375. doi:10.1016/j.ifacol.2022.07.472

21. Liñán DA, Reynoso-Donzelli S, Ricardez-Sandoval LA. Optimal scheduling and open-loop control of network batch processes under variable processing times using Generalized Benders Decomposition. In: *2024 American Control Conference (ACC)*. ; 2024:4466-4471. doi:10.23919/ACC60939.2024.10644393

22. Audet C, Hare W. *Derivative-Free and Blackbox Optimization*. 1st ed. 2017 edition. Springer; 2017.

23. Palma-Flores O, Ricardez-Sandoval LA. Integration of design and NMPC-based control for chemical processes under uncertainty: An MPCC-based framework. *Computers & Chemical Engineering*. 2022;162:107815. doi:10.1016/j.compchemeng.2022.107815

24. Rafiei M, Ricardez-Sandoval LA. Stochastic Back-Off Approach for Integration of Design and Control Under Uncertainty. *Ind Eng Chem Res*. 2018;57(12):4351-4365. doi:10.1021/acs.iecr.7b03935

25. Buitimea-Cerón GA, Medina-Herrera N, Jiménez-Gutiérrez A, Tututi-Avila S. Design of a multiproduct reactive distillation system for DEC and EMC production. *Chemical Engineering and Processing - Process Intensification*. 2023;192:109493. doi:10.1016/j.cep.2023.109493

26. Alcántara-Avila JR, Tanaka M, Ramírez Márquez C, et al. Design of a Multitask Reactive Distillation with Intermediate Heat Exchangers for the Production of Silane and Chlorosilane Derivates. *Ind Eng Chem Res*. 2016;55(41):10968-10977. doi:10.1021/acs.iecr.6b02277

27. Thotla S, Agarwal V, Mahajani SM. Simultaneous production of diacetone alcohol and mesityl oxide from acetone using reactive distillation. *Chemical Engineering Science*. 2007;62(18):5567-5574. doi:10.1016/j.ces.2007.01.045

28. Medina-Herrera N, Jiménez-Gutiérrez A, Ricardez-Sandoval LA, Tututi-Avila S. An approach for dynamic transitions in multiproduct reactive distillation columns. *Chemical Engineering and Processing - Process Intensification*. 2020;153:107967. doi:10.1016/j.cep.2020.107967

29. Segovia-Hernández JG, Hernández S, Bonilla Petriciolet A. Reactive distillation: A review of optimal design using deterministic and stochastic techniques. *Chemical Engineering and Processing: Process Intensification*. 2015;97:134-143. doi:10.1016/j.cep.2015.09.004

30. Palma-Flores O, Ricardez-Sandoval LA, Biegler LT. Simultaneous design and NMPC control under uncertainty and structural decisions: A discrete-steepest descent algorithm. *AIChE Journal*. 2023;69(11):e18188. doi:10.1002/aic.18188

31. Burre J, Bongartz D, Mitsos A. Comparison of MINLP formulations for global superstructure optimization. *Optim Eng*. Published online January 13, 2022. doi:10.1007/s11081-021-09707-y

32. Liberti L, Kucherenko S. Comparison of deterministic and stochastic approaches to global optimization. *International Transactions in Operational Research*. 2005;12(3):263-285. doi:10.1111/j.1475-3995.2005.00503.x

33. Skiborowski M, Rautenberg M, Marquardt W. A Hybrid Evolutionary–Deterministic Optimization Approach for Conceptual Design. *Ind Eng Chem Res*. 2015;54(41):10054-10072. doi:10.1021/acs.iecr.5b01995

34. Reynoso-Donzelli S, Ricardez-Sandoval LA. A reinforcement learning approach with masked agents for chemical process flowsheet design. *AIChE Journal*. 2024;n/a(n/a):e18584. doi:10.1002/aic.18584

35. Reynoso-Donzelli S, Ricardez-Sandoval LA. Enhancing process flowsheet design through masked hybrid Proximal Policy Optimization. *IFAC-PapersOnLine*. 2024;58(14):646-651. doi:10.1016/j.ifacol.2024.08.410

36. Kronqvist J, Bernal DE, Lundell A, Grossmann IE. A review and comparison of solvers for convex MINLP. *Optim Eng*. 2019;20(2):397-455. doi:10.1007/s11081-018-9411-8

37. Franke MB. Design of Dividing-Wall Columns by Mixed-Integer Nonlinear Programming Optimization. *Chemie Ingenieur Technik*. 2017;89(5):582-597. doi:10.1002/cite.201700005

38. Kruber KF, Grueters T, Skiborowski M. Advanced hybrid optimization methods for the design of complex separation processes. *Computers & Chemical Engineering*. 2021;147:107257. doi:10.1016/j.compchemeng.2021.107257

39. Costa ALH, Bagajewicz MJ. 110th Anniversary: On the Departure from Heuristics and Simplified Models toward Globally Optimal Design of Process Equipment. *Ind Eng Chem Res*. 2019;58(40):18684-18702. doi:10.1021/acs.iecr.9b02611

40. Harjunkoski I, Maravelias CT, Bongers P, et al. Scope for industrial applications of production scheduling models and solution methods. *Computers & Chemical Engineering*. 2014;62:161-193.

41. Maravelias CT. *Chemical Production Scheduling: Mixed-Integer Programming Models and Methods*. Cambridge University Press; 2021. doi:10.1017/9781316650998

42. Andrés-Martínez O, Ricardez-Sandoval LA. Integration of planning, scheduling, and control: A review and new perspectives. *The Canadian Journal of Chemical Engineering*. 2022;100(9):2057-2070. doi:10.1002/cjce.24501

43. Caspari A, Tsay C, Mhamdi A, Baldea M, Mitsos A. The integration of scheduling and control: Top-down vs. bottom-up. *Journal of Process Control*. 2020;91:50-62. doi:10.1016/j.jprocont.2020.05.008

44. Castro PM, Grossmann IE, Zhang Q. Expanding scope and computational challenges in process scheduling. *Computers & Chemical Engineering*. 2018;114:14-42. doi:10.1016/j.compchemeng.2018.01.020

45. Lozano Santamaria F, Macchietto S. Stability of Optimal Closed-Loop Cleaning Scheduling and Control with Application to Heat Exchanger Networks under Fouling. *Processes*. 2020;8(12):1623. doi:10.3390/pr8121623

46. Simkoff JM, Baldea M. Production scheduling and linear MPC: Complete integration via complementarity conditions. *Computers & Chemical Engineering*. 2019;125:287-305. doi:10.1016/j.compchemeng.2019.01.024

47. Pattison RC, Touretzky CR, Harjunkoski I, Baldea M. Moving horizon closed-loop production scheduling using dynamic process models. *AIChE Journal*. 2017;63(2):639-651. doi:10.1002/aic.15408

48. Mitrai I, Daoutidis P. A multicut generalized benders decomposition approach for the integration of process operations and dynamic optimization for continuous systems. *Computers & Chemical Engineering*. 2022;164:107859. doi:10.1016/j.compchemeng.2022.107859

49. Baldea M, Harjunkoski I. Integrated production scheduling and process control: A systematic review. *Computers & Chemical Engineering*. 2014;71:377-390. doi:10.1016/j.compchemeng.2014.09.002

50. Georgiadis GP, Elekidis AP, Georgiadis MC. Optimization-Based Scheduling for the Process Industries: From Theory to Real-Life Industrial Applications. *Processes*. 2019;7(7):438.

51. Rodríguez Vera HU, Ricardez-Sandoval LA. Integration of Scheduling and Control for Chemical Batch Plants under Stochastic Uncertainty: A Back-Off Approach. *Ind Eng Chem Res*. 2022;61(12):4363-4378. doi:10.1021/acs.iecr.1c04386

52. Valdez-Navarro YI, Ricardez-Sandoval LA. Integration between Dynamic Optimization and Scheduling of Batch Processes under Uncertainty: A Back-off Approach. *IFAC-PapersOnLine*. 2019;52(1):655-660. doi:10.1016/j.ifacol.2019.06.137

53. Shobrys DE, White DC. Planning, scheduling and control systems: why cannot they work together. *Computers & Chemical Engineering*. 2002;26(2):149-160. doi:10.1016/S0098-1354(01)00737-2

54. Pistikopoulos EN, Akundi SS, Kenefake D, Diangelakis NA. The quest towards the integration of process control, process operations, and process operability –Industrial need or academic curiosity? *Computers & Chemical Engineering*. 2024;180:108470. doi:10.1016/j.compchemeng.2023.108470

55. Liberti L, Sager S, Wiegele A. Preface. *Math Program*. 2021;188(2):411-419. doi:10.1007/s10107-021-01687-2

56. Geoffrion AM. Generalized Benders decomposition. *J Optim Theory Appl*. 1972;10(4):237-260. doi:10.1007/BF00934810

57. Grossmann IE. Mixed-integer programming approach for the synthesis of integrated process flowsheets. *Computers & Chemical Engineering*. 1985;9(5):463-482. doi:10.1016/0098-1354(85)80023-5

58. Duran MA, Grossmann IE. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*. 1986;36(3):307-339. doi:10.1007/BF02592064

59. Boukouvala F, Misener R, Floudas CA. Global optimization advances in Mixed-Integer Nonlinear Programming, MINLP, and Constrained Derivative-Free Optimization, CDFO. *European Journal of Operational Research*. 2016;252(3):701-727. doi:10.1016/j.ejor.2015.12.018

60. Ji N, Gu X. Integration of scheduling and control for batch process based on generalized Benders decomposition approach with genetic algorithm. *Computers & Chemical Engineering*. 2021;145:107166. doi:10.1016/j.compchemeng.2020.107166

61. Chu Y, You F. Integrated Scheduling and Dynamic Optimization of Complex Batch Processes with General Network Structure Using a Generalized Benders Decomposition Approach. *Ind Eng Chem Res*. 2013;52(23):7867-7885. doi:10.1021/ie400475s

62. Nie Y, Biegler LT, Villa CM, Wassick JM. Discrete Time Formulation for the Integration of Scheduling and Dynamic Optimization. *Ind Eng Chem Res*. 2015;54(16):4303-4315. doi:10.1021/ie502960p

63. Westerlund T, Pettersson F. An extended cutting plane method for solving convex MINLP problems. *Computers & Chemical Engineering*. 1995;19:131-136. doi:10.1016/0098-1354(95)87027-X

64. Kronqvist J, Lundell A, Westerlund T. The extended supporting hyperplane algorithm for convex mixed-integer nonlinear programming. *J Glob Optim*. 2016;64(2):249-272. doi:10.1007/s10898-015-0322-3

65. Vaselenak JA, Grossmann IE, Westerberg AW. An embedding formulation for the optimal scheduling and design of multipurpose batch plants. *Ind Eng Chem Res*. 1987;26(1):139-148. doi:10.1021/ie00061a027

66. Faqir NM, Karimi IA. Optimal design of batch plants with single production routes. *Ind Eng Chem Res*. 1989;28(8):1191-1202. doi:10.1021/ie00092a012

67. Bansal V, Sakizlis V, Ross R, Perkins JD, Pistikopoulos EN. New algorithms for mixed-integer dynamic optimization. *Computers & Chemical Engineering*. 2003;27(5):647-668. doi:10.1016/S0098-1354(02)00261-2

68. Luyben ML, Floudas CA. Analyzing the interaction of design and control—1. A multiobjective framework and application to binary distillation synthesis. *Computers & Chemical Engineering*. 1994;18(10):933-969. doi:10.1016/0098-1354(94)E0013-D

69. Mohideen MJ, Perkins JD, Pistikopoulos EN. Optimal design of dynamic systems under uncertainty. *AIChE Journal*. 1996;42(8):2251-2272. doi:10.1002/aic.690420814

70. Lin X, Floudas CA. Design, synthesis and scheduling of multipurpose batch plants via an effective continuous-time formulation. *Computers & Chemical Engineering*. 2001;25(4):665-674. doi:10.1016/S0098-1354(01)00663-9

71. Vasudevan A, Narayanasamy S, Paramasivan G. Simultaneous Design and Control of a Ternary Reactive Distillation Column with Inert Material. *Chemical Engineering & Technology*. 2018;41(11):2233-2240. doi:10.1002/ceat.201700708

72. Flores-Tlacuahuac A, Grossmann IE. Simultaneous Scheduling and Control of Multiproduct Continuous Parallel Lines. *Ind Eng Chem Res*. 2010;49(17):7909-7921. doi:10.1021/ie100024p

73. Nie Y, Biegler LT, Wassick JM. Integrated scheduling and dynamic optimization of batch processes using state equipment networks. *AIChE Journal*. 2012;58(11):3416-3432. doi:10.1002/aic.13738

74. Zhuge J, Ierapetritou MG. Integration of Scheduling and Control with Closed Loop Implementation. *Ind Eng Chem Res*. 2012;51(25):8550-8565. doi:10.1021/ie3002364

75. Lozano Santamaria F, Macchietto S. Integration of Optimal Cleaning Scheduling and Control of Heat Exchanger Networks Undergoing Fouling: Model and Formulation. *Ind Eng Chem Res*. 2018;57(38):12842-12860. doi:10.1021/acs.iecr.8b01701

76. Zhao X, Marquardt W. Reactor network synthesis with guaranteed robust stability. *Computers & Chemical Engineering*. 2016;86:75-89. doi:10.1016/j.compchemeng.2015.12.005

77. Lasschuit W, Thijssen N. Supporting supply chain planning and scheduling decisions in the oil and chemical industry. *Computers & Chemical Engineering*. 2004;28(6):863-870. doi:10.1016/j.compchemeng.2003.09.026

78. Asteasuain M, Brandolin A, Sarmoria C, Bandoni A. Simultaneous Design and Control of a Semibatch Styrene Polymerization Reactor. *Ind Eng Chem Res*. 2004;43(17):5233-5247. doi:10.1021/ie040021s

79. Flores-Tlacuahuac A, Biegler LT. Simultaneous mixed-integer dynamic optimization for integrated design and control. *Computers & Chemical Engineering*. 2007;31(5):588-600. doi:10.1016/j.compchemeng.2006.08.010

80. Meidanshahi V, Adams TA. Integrated design and control of semicontinuous distillation systems utilizing mixed integer dynamic optimization. *Computers & Chemical Engineering*. 2016;89:172-183. doi:10.1016/j.compchemeng.2016.03.022

81. Flores-Tlacuahuac A, Grossmann IE. Simultaneous Cyclic Scheduling and Control of a Multiproduct CSTR. *Ind Eng Chem Res*. 2006;45(20):6698-6712. doi:10.1021/ie051293d

82. Terrazas-Moreno S, Flores-Tlacuahuac A, Grossmann IE. Simultaneous cyclic scheduling and optimal control of polymerization reactors. *AIChE Journal*. 2007;53(9):2301-2315. doi:10.1002/aic.11247

83. Al-Qahtani K, Elkamel A, Ponnambalam K. Robust Optimization for Petrochemical Network Design under Uncertainty. *Ind Eng Chem Res*. 2008;47(11):3912-3919. doi:10.1021/ie0713184

84. Lu Y yue, Hu Y dong, Xu D mei, Wu L ying. Optimum design of reverse osmosis seawater desalination system considering membrane cleaning and replacing. *Journal of Membrane Science*. 2006;282(1):7-13. doi:10.1016/j.memsci.2006.04.019

85. Tanvir MS, Mujtaba IM. Optimisation of design and operation of MSF desalination process using MINLP technique in gPROMS. *Desalination*. 2008;222(1):419-430. doi:10.1016/j.desal.2007.02.068

86. Gutiérrez-Limón MA, Flores-Tlacuahuac A, Grossmann IE. MINLP Formulation for Simultaneous Planning, Scheduling, and Control of Short-Period Single-Unit Processing Systems. *Ind Eng Chem Res*. 2014;53(38):14679-14694. doi:10.1021/ie402563j

87. Peng Y, Ricardez-Sandoval L. Integration of Planning, Scheduling, and Control for Multi-product Chemical Systems under Preventive Maintenance. *IFAC-PapersOnLine*. 2021;54(3):536-541. doi:10.1016/j.ifacol.2021.08.297

88. Terrazas-Moreno S, Flores-Tlacuahuac A, Grossmann IE. Simultaneous design, scheduling, and optimal control of a methyl-methacrylate continuous polymerization reactor. *AIChE Journal*. 2008;54(12):3160-3170. doi:10.1002/aic.11658

89. Corsano G, Aguirre PA, Montagna JM. Multiperiod design and planning of multiproduct batch plants with mixed-product campaigns. *AIChE Journal*. 2009;55(9):2356-2369. doi:10.1002/aic.11854

90. Cafaro DC, Grossmann IE. Strategic planning, design, and development of the shale gas supply chain network. *AIChE Journal*. 2014;60(6):2122-2142. doi:10.1002/aic.14405

91. Ma Y, Yang Z, El-Khoruy A, et al. Simultaneous Synthesis and Design of Reaction–Separation–Recycle Processes Using Rigorous Models. *Ind Eng Chem Res*. 2021;60(19):7275-7290. doi:10.1021/acs.iecr.1c00250

92. Akulker H, Aydin E. Optimal design and operation of a multi-energy microgrid using mixed-integer nonlinear programming: Impact of carbon cap and trade system and taxing on equipment selections. *Applied Energy*. 2023;330:120313. doi:10.1016/j.apenergy.2022.120313

93. Liu C, Zhao C, Xu Q. Integration of electroplating process design and operation for simultaneous productivity maximization, energy saving, and freshwater minimization. *Chemical Engineering Science*. 2012;68(1):202-214. doi:10.1016/j.ces.2011.09.024

94. Valdez-Navarro YI, Ricardez-Sandoval LA. A Novel Back-off Algorithm for Integration of Scheduling and Control of Batch Processes under Uncertainty. *Ind Eng Chem Res*. 2019;58(48):22064-22083. doi:10.1021/acs.iecr.9b04963

95. Burnak B, Diangelakis NA, Katz J, Pistikopoulos EN. Integrated process design, scheduling, and control using multiparametric programming. *Computers & Chemical Engineering*. 2019;125:164-184. doi:10.1016/j.compchemeng.2019.03.004

96. Pappas I, Kenefake D, Burnak B, et al. Multiparametric Programming in Process Systems Engineering: Recent Developments and Path Forward. *Front Chem Eng*. 2021;2. doi:10.3389/fceng.2020.620168

97. Chu Y, You F. Integrated Scheduling and Dynamic Optimization by Stackelberg Game: Bilevel Model Formulation and Efficient Solution Algorithm. *Ind Eng Chem Res*. 2014;53(13):5564-5581. doi:10.1021/ie404272t

98. Koller RW, Ricardez-Sandoval LA, Biegler LT. Stochastic back-off algorithm for simultaneous design, control, and scheduling of multiproduct systems under uncertainty. *AIChE Journal*. 2018;64(7):2379-2389. doi:10.1002/aic.16092

99. Orçun S, Altinel IK, Hortaçsu Ö. General continuous time models for production planning and scheduling of batch processing plants: mixed integer linear program formulations and computational issues. *Computers & Chemical Engineering*. 2001;25(2):371-389. doi:10.1016/S0098-1354(00)00663-3

100. Tan W, Khoshnevis B. A linearized polynomial mixed integer programming model for the integration of process planning and scheduling. *Journal of Intelligent Manufacturing*. 2004;15(5):593-605. doi:10.1023/B:JIMS.0000037710.80847.b6

101. Menezes BC, Kelly JD, Grossmann IE, Vazacopoulos A. Generalized capital investment planning of oil-refineries using MILP and sequence-dependent setups. *Computers & Chemical Engineering*. 2015;80:140-154. doi:10.1016/j.compchemeng.2015.05.013

102. Zhuge J, Ierapetritou MG. An integrated framework for scheduling and control using fast model predictive control. *AIChE Journal*. 2015;61(10):3304-3319. doi:10.1002/aic.14914

103. Kelley MT, Pattison RC, Baldick R, Baldea M. An efficient MILP framework for integrating nonlinear process dynamics and control in optimal production scheduling calculations. *Computers & Chemical Engineering*. 2018;110:35-52. doi:10.1016/j.compchemeng.2017.11.021

104. Kelley MT, Pattison RC, Baldick R, Baldea M. An MILP framework for optimizing demand response operation of air separation units. *Applied Energy*. 2018;222:951-966. doi:10.1016/j.apenergy.2017.12.127

105. Chu Y, You F. Integrated Planning, Scheduling, and Dynamic Optimization for Batch Processes: MINLP Model Formulation and Efficient Solution Methods via Surrogate Modeling. *Ind Eng Chem Res*. 2014;53(34):13391-13411. doi:10.1021/ie501986d

106. Yue D, You F. Planning and scheduling of flexible process networks under uncertainty with stochastic inventory: MINLP models and algorithm. *AIChE Journal*. 2013;59(5):1511-1532. doi:10.1002/aic.13924

107. Stein O, Oldenburg J, Marquardt W. Continuous reformulations of discrete–continuous optimization problems. *Computers & Chemical Engineering*. 2004;28(10):1951-1966. doi:10.1016/j.compchemeng.2004.03.011

108. Baumrucker BT, Renfro JG, Biegler LT. MPEC problem formulations and solution strategies with chemical engineering applications. *Computers & Chemical Engineering*. 2008;32(12):2903-2913. doi:10.1016/j.compchemeng.2008.02.010

109. Santamaria FL, Macchietto S. Integration of optimal cleaning scheduling and control of heat exchanger networks under fouling: MPCC solution. *Computers & Chemical Engineering*. 2019;126:128-146. doi:10.1016/j.compchemeng.2019.04.012

110. Andrés-Martínez O, Ricardez-Sandoval LA. A nested online scheduling and nonlinear model predictive control framework for multi-product continuous systems. *AIChE Journal*. 2022;68(5):e17665. doi:10.1002/aic.17665

111. Zhao X, Marquardt W. Closed-loop reactor network synthesis with guaranteed robustness. *Computers & Chemical Engineering*. 2017;107:294-317. doi:10.1016/j.compchemeng.2017.06.010

112. Ramos MA, Gómez JM, Reneaume JM. Simultaneous Optimal Design and Control of an Extractive Distillation System for the Production of Fuel Grade Ethanol Using a Mathematical Program with Complementarity Constraints. *Ind Eng Chem Res*. 2014;53(2):752-764. doi:10.1021/ie402232w

113. Guignard M. Lagrangean relaxation. *Top*. 2003;11(2):151-200. doi:10.1007/BF02579036

114. Mora-Mariano D, Gutiérrez-Limón MA, Flores-Tlacuahuac A. A Lagrangean decomposition optimization approach for long-term planning, scheduling and control. *Computers & Chemical Engineering*. 2020;135:106713. doi:10.1016/j.compchemeng.2019.106713

115. Terrazas-Moreno S, Flores-Tlacuahuac A, Grossmann IE. Lagrangean heuristic for the scheduling and control of polymerization reactors. *AIChE Journal*. 2008;54(1):163-182. doi:10.1002/aic.11343

116. Fisher ML. An Applications Oriented Guide to Lagrangian Relaxation. *Interfaces*. 1985;15(2):10-21.

117. Yongheng J, Rodriguez MA, Harjunkoski I, Grossmann IE. Optimal supply chain design and management over a multi-period horizon under demand uncertainty. Part II: A Lagrangean decomposition algorithm. *Computers & Chemical Engineering*. 2014;62:211-224. doi:10.1016/j.compchemeng.2013.11.014

118. Mouret S, Grossmann IE, Pestiaux P. A new Lagrangian decomposition approach applied to the integration of refinery planning and crude-oil scheduling. *Computers & Chemical Engineering*. 2011;35(12):2750-2766. doi:10.1016/j.compchemeng.2011.03.026

119. Yang H, Bernal DE, Franzoi RE, et al. Integration of crude-oil scheduling and refinery planning by Lagrangean Decomposition. *Computers & Chemical Engineering*. 2020;138:106812. doi:10.1016/j.compchemeng.2020.106812

120. Chen Q, Liu Y, Seastream G, Siirola JD, Grossmann IE. Pyosyn: A new framework for conceptual design modeling and optimization. *Computers & Chemical Engineering*. 2021;153:107414. doi:10.1016/j.compchemeng.2021.107414

121. Trespalacios F, Grossmann IE. Review of Mixed-Integer Nonlinear and Generalized Disjunctive Programming Methods. *Chemie Ingenieur Technik*. 2014;86(7):991-1012. doi:10.1002/cite.201400037

122. Montaña Rosado ID, Gómez Ramírez JM. *Implementación Del Método -d-Sda y Variables de Decisión Externa- Para El Diseño Óptimo de Columnas de Destilación Extractiva*. Uniandes; 2021. Accessed May 20, 2021. https://biblioteca.uniandes.edu.co/

123. Ovalle D, Liñán DA, Lee A, et al. Logic-Based Discrete-Steepest Descent: A Solution Method for Process Synthesis Generalized Disjunctive Programs. Published online May 8, 2024. Accessed July 23, 2024. http://arxiv.org/abs/2405.05358

124. Ciré AA, Çoban E, Hooker JN. Logic-based Benders decomposition for planning and scheduling: a computational analysis. *The Knowledge Engineering Review*. 2016;31(5):440-451. doi:10.1017/S0269888916000254

125. Juvin C, Houssin L, Lopez P. Logic-based Benders decomposition for the preemptive flexible job-shop scheduling problem. *Computers & Operations Research*. 2023;152:106156. doi:10.1016/j.cor.2023.106156

126. Zhu X, Son J, Zhang X, Wu J. Constraint programming and logic-based Benders decomposition for the integrated process planning and scheduling problem. *Omega*. 2023;117:102823. doi:10.1016/j.omega.2022.102823

127. Elçi Ö, Hooker J. Stochastic Planning and Scheduling with Logic-Based Benders Decomposition. *INFORMS Journal on Computing*. 2022;34(5):2428-2442. doi:10.1287/ijoc.2022.1184

128. Hooker J. *Logic-Based Benders Decomposition: Theory and Applications*. 2024th edition. Springer Nature; 2023.

129. Srinivas M, Rangaiah GP. An integrated stochastic method for global optimization of continuous functions. In: Marquardt W, Pantelides C, eds. *Computer Aided Chemical Engineering*. Vol 21. 16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process Systems Engineering. Elsevier; 2006:439-444. doi:10.1016/S1570-7946(06)80085-4

130. Munawar SA, Gudi RD. A Nonlinear Transformation Based Hybrid Evolutionary Method for MINLP Solution. *Chemical Engineering Research and Design*. 2005;83(10):1218-1236. doi:10.1205/cherd.04286

131. Staudt PB, de P. Soares R. Reliability vs. Efficiency When Solving Multiphase Equilibrium Problems with Hybrid Optimization Codes. In: de Brito Alves RM, do Nascimento CAO, Biscaia EC, eds. *Computer Aided Chemical Engineering*. Vol 27. 10th International Symposium on Process Systems Engineering: Part A. Elsevier; 2009:585-590. doi:10.1016/S1570-7946(09)70318-9

132. Chia DN, Duanmu F, Sorensen E. Optimal Design of Distillation Columns Using a Combined Optimisation Approach. In: Türkay M, Gani R, eds. *Computer Aided Chemical Engineering*. Vol 50. 31 European Symposium on Computer Aided Process Engineering. Elsevier; 2021:153-158. doi:10.1016/B978-0-323-88506-5.50025-5

133. Herrera Velázquez JJ, Zavala Durán FM, Chávez Díaz LA, Cabrera Ruiz J, Alcántara Avila JR. Hybrid two-step optimization of internally heat-integrated distillation columns. *Journal of the Taiwan Institute of Chemical Engineers*. 2022;130:103967. doi:10.1016/j.jtice.2021.06.061

134. Urselmann M, Barkmann S, Sand G, Engell S. A Memetic Algorithm for Global Optimization in Chemical Process Synthesis Problems. *IEEE Transactions on Evolutionary Computation*. 2011;15(5):659-683. doi:10.1109/TEVC.2011.2150753

135. Urselmann M, Janus T, Foussette C, et al. Derivative-Free Chemical Process Synthesis by Memetic Algorithms Coupled to Aspen Plus Process Models. In: Kravanja Z, Bogataj M, eds. *Computer Aided Chemical Engineering*. Vol 38. 26 European Symposium on Computer Aided Process Engineering. Elsevier; 2016:187-192. doi:10.1016/B978-0-444-63428-3.50036-9

136. Urselmann M, Barkmann S, Sand G, Engell S. Optimization-based design of reactive distillation columns using a memetic algorithm. *Computers & Chemical Engineering*. 2011;35(5):787-805. doi:10.1016/j.compchemeng.2011.01.038

137. Zhou T, Zhou Y, Sundmacher K. A hybrid stochastic–deterministic optimization approach for integrated solvent and process design. *Chemical Engineering Science*. 2017;159:207-216. doi:10.1016/j.ces.2016.03.011

138. Gómez JM, Reneaume JM, Roques M, Meyer M, Meyer X. A Mixed Integer Nonlinear Programming Formulation for Optimal Design of a Catalytic Distillation Column Based on a Generic Nonequilibrium Model. *Ind Eng Chem Res*. 2006;45(4):1373-1388. doi:10.1021/ie0504506

139. Holtbruegge J, Kuhlmann H, Lutze P. Process analysis and economic optimization of intensified process alternatives for simultaneous industrial scale production of dimethyl carbonate and propylene glycol. *Chemical Engineering Research and Design*. 2015;93:411-431. doi:10.1016/j.cherd.2014.05.002

140. Hernández-Pérez LG, Ramírez-Márquez C, Segovia-Hernández JG, Ponce-Ortega JM. Simultaneous structural and operating optimization of process flowsheets combining process simulators and metaheuristic techniques: The case of solar-grade silicon process. *Computers & Chemical Engineering*. 2020;140:106946. doi:10.1016/j.compchemeng.2020.106946

141. Javaloyes-Antón J, Kronqvist J, Caballero JA. Simulation-based optimization of distillation processes using an extended cutting plane algorithm. *Computers & Chemical Engineering*. 2022;159:107655. doi:10.1016/j.compchemeng.2021.107655

142. Charpentier JC. Among the trends for a modern chemical engineering, the third paradigm: The time and length multiscale approach as an efficient tool for process intensification and product

design and engineering. *Chemical Engineering Research and Design*. 2010;88(3):248-254. doi:10.1016/j.cherd.2009.03.008

143. Kumar V, Nigam KDP. Process intensification in green synthesis. *Green Processing and Synthesis*. 2012;1(1):79-107. doi:10.1515/greenps-2011-0003

144. Kurt SK, Vural Gürsel I, Hessel V, Nigam KDP, Kockmann N. Liquid–liquid extraction system with microstructured coiled flow inverter and other capillary setups for single-stage extraction applications. *Chemical Engineering Journal*. 2016;284:764-777. doi:10.1016/j.cej.2015.08.099

145. Mansouri SS, Huusom JK, Gani R, Sales-Cruz M. Systematic integrated process design and control of binary element reactive distillation processes. *AIChE Journal*. 2016;62(9):3137-3154. doi:10.1002/aic.15322

146. Nigam KDP, Larachi F. Process intensification in trickle-bed reactors. *Chemical Engineering Science*. 2005;60(22):5880-5894. doi:10.1016/j.ces.2005.04.061

147. Singh J, Montesinos-Castellanos A, Nigam KDP. Process Intensification for Compact and Micro Heat Exchangers through Innovative Technologies: A Review. *Ind Eng Chem Res*. 2019;58(31):13819-13847. doi:10.1021/acs.iecr.9b02082

148. Uhlemann J, Costa R, Charpentier JC. Product design and engineering — past, present, future trends in teaching, research and practices: academic and industry points of view. *Current Opinion in Chemical Engineering*. 2020;27:10-21. doi:10.1016/j.coche.2019.10.003

149. Stamenić M, Dikić V, Mandić M, Todić B, Bukur DB, Nikačević NM. Multiscale and Multiphase Model of Fixed-Bed Reactors for Fischer–Tropsch Synthesis: Optimization Study. *Ind Eng Chem Res*. 2018;57(9):3149-3162. doi:10.1021/acs.iecr.7b04914

150. Stamenić M, Dikić V, Mandić M, Todić B, Bukur DB, Nikačević NM. Multiscale and Multiphase Model of Fixed Bed Reactors for Fischer–Tropsch Synthesis: Intensification Possibilities Study. *Ind Eng Chem Res*. 2017;56(36):9964-9979. doi:10.1021/acs.iecr.7b02467

151. Onel O, Niziolek AM, Butcher H, Wilhite BA, Floudas CA. Multi-scale approaches for gas-to-liquids process intensification: CFD modeling, process synthesis, and global optimization. *Computers & Chemical Engineering*. 2017;105:276-296. doi:10.1016/j.compchemeng.2017.01.016

152. Zhang F, Damjanac B, Maxwell S. Investigating Hydraulic Fracturing Complexity in Naturally Fractured Rock Masses Using Fully Coupled Multiscale Numerical Modeling. *Rock Mech Rock Eng*. 2019;52(12):5137-5160. doi:10.1007/s00603-019-01851-3

153. da Rosa CA, Braatz RD. Multiscale Modeling and Simulation of Macromixing, Micromixing, and Crystal Size Distribution in Radial Mixers/Crystallizers. *Ind Eng Chem Res*. 2018;57(15):5433-5441. doi:10.1021/acs.iecr.8b00359

154. Kwon JSI, Nayhouse M, Christofides PD. Multiscale, Multidomain Modeling and Parallel Computation: Application to Crystal Shape Evolution in Crystallization. *Ind Eng Chem Res*. 2015;54(47):11903-11914. doi:10.1021/acs.iecr.5b02942

155. Choi HK, Kwon JSI. Multiscale modeling and control of fiber length in pulp digester. In: *2020 American Control Conference (ACC)*. ; 2020:4343-4348. doi:10.23919/ACC45564.2020.9147895

156. Golmon S, Maute K, Dunn ML. Multiscale design optimization of lithium ion batteries using adjoint sensitivity analysis. *International Journal for Numerical Methods in Engineering*. 2012;92(5):475-494. doi:10.1002/nme.4347

157. Kashkooli AG, Farhad S, Lee DU, et al. Multiscale modeling of lithium-ion battery electrodes based on nano-scale X-ray computed tomography. *Journal of Power Sources*. 2016;307:496-509. doi:10.1016/j.jpowsour.2015.12.134

158. Karakasidis TE, Charitidis CA. Multiscale modeling in nanomaterials science. *Materials Science and Engineering: C*. 2007;27(5):1082-1089. doi:10.1016/j.msec.2006.06.029

159. Urrea-Quintero JH, Ochoa S, Hernández H. A reduced-order multiscale model of a free-radical semibatch emulsion polymerization process. *Computers & Chemical Engineering*. 2019;127:11-24. doi:10.1016/j.compchemeng.2019.04.029

160. Bruix A, Margraf JT, Andersen M, Reuter K. First-principles-based multiscale modelling of heterogeneous catalysis. *Nature Catalysis*. 2019;2(8):659-670. doi:10.1038/s41929-019-0298-3

161. Keil FJ. Multiscale Modelling in Computational Heterogeneous Catalysis. In: Kirchner B, Vrabec J, eds. *Multiscale Molecular Methods in Applied Chemistry*. Topics in Current Chemistry. Springer; 2012:69-107. doi:10.1007/128_2011_128

162. Liu DJ, Evans JW. Realistic multisite lattice-gas modeling and KMC simulation of catalytic surface reactions: Kinetics and multiscale spatial behavior for CO-oxidation on metal (100) surfaces. *Progress in Surface Science*. 2013;88(4):393-521. doi:10.1016/j.progsurf.2013.10.001

163. Salciccioli M, Stamatakis M, Caratzoulas S, Vlachos DG. A review of multiscale modeling of metal-catalyzed reactions: Mechanism development for complexity and emergent behavior. *Chemical Engineering Science*. 2011;66(19):4319-4355. doi:10.1016/j.ces.2011.05.050

164. Sengar A, Kuipers JAM, van Santen RA, Padding JT. Towards a particle based approach for multiscale modeling of heterogeneous catalytic reactors. *Chemical Engineering Science*. 2019;198:184-197. doi:10.1016/j.ces.2018.10.038

165. Christofides PD, Armaou A, Lou Y, Varshney A. *Control and Optimization of Multiscale Process Systems*. Birkhäuser Basel; 2009. doi:10.1007/978-0-8176-4793-3

166. Ricardez-Sandoval LA. Current challenges in the design and control of multiscale systems. *The Canadian Journal of Chemical Engineering*. 2011;89(6):1324-1341. doi:10.1002/cjce.20607

167. Vlachos DG. A Review of Multiscale Analysis: Examples from Systems Biology, Materials Engineering, and Other Fluid–Surface Interacting Systems. In: Marin GB, ed. *Advances in Chemical Engineering*. Vol 30. Advances in Chemical Engineering. Academic Press; 2005:1-61. doi:10.1016/S0065-2377(05)30001-9

168. Vlachos DG. Multiscale integration hybrid algorithms for homogeneous–heterogeneous reactors. *AIChE Journal*. 1997;43(11):3031-3041. doi:10.1002/aic.690431115

169. Kimaev G, Ricardez-Sandoval LA. Nonlinear model predictive control of a multiscale thin film deposition process using artificial neural networks. *Chemical Engineering Science*. 2019;207:1230-1245. doi:10.1016/j.ces.2019.07.044

170. Rasoulian S, Ricardez-Sandoval LA. Robust multivariable estimation and control in an epitaxial thin film growth process under uncertainty. *Journal of Process Control*. 2015;34:70-81. doi:10.1016/j.jprocont.2015.07.002

171. Rasoulian S, Ricardez-Sandoval LA. Uncertainty analysis and robust optimization of multiscale process systems with application to epitaxial thin film growth. *Chemical Engineering Science*. 2014;116:590-600. doi:10.1016/j.ces.2014.05.027

172. Bangi MSF, Kwon JSI. Deep hybrid modeling of chemical process: Application to hydraulic fracturing. *Computers & Chemical Engineering*. 2020;134:106696. doi:10.1016/j.compchemeng.2019.106696

173. Chaffart D, Ricardez-Sandoval LA. Optimization and control of a thin film growth process: A hybrid first principles/artificial neural network based multiscale modelling approach. *Computers & Chemical Engineering*. 2018;119:465-479. doi:10.1016/j.compchemeng.2018.08.029

174. Ding Y, Zhang Y, Ren YM, Orkoulas G, Christofides PD. Machine learning-based modeling and operation for ALD of SiO2 thin-films using data from a multiscale CFD simulation. *Chemical Engineering Research and Design*. 2019;151:131-145. doi:10.1016/j.cherd.2019.09.005

175. Narasingam A, Sang-Il Kwon J. Data-driven identification of interpretable reduced-order models using sparse regression. *Computers & Chemical Engineering*. 2018;119:101-111. doi:10.1016/j.compchemeng.2018.08.010

176. Siddhamshetty P, Yang S, Kwon JSI. Modeling of hydraulic fracturing and designing of online pumping schedules to achieve uniform proppant concentration in conventional oil reservoirs. *Computers & Chemical Engineering*. 2018;114:306-317. doi:10.1016/j.compchemeng.2017.10.032

177. García-Sánchez M, Sales-Cruz M, Lopez-Arenas T, Viveros-García T, Pérez-Cisneros ES. An Intensified Reactive Separation Process for Bio-Jet Diesel Production. *Processes*. 2019;7(10):655. doi:10.3390/pr7100655

178. Huang K, Yuan Y, Zang X, et al. Controllability Evaluation for Reactive Distillation Columns with Multiple Reactive Sections Disproportionating Trichlorosilane to Silane. *Ind Eng Chem Res*. 2018;57(3):1102-1108. doi:10.1021/acs.iecr.7b03666

179. Kaur J, Chugh T, Sangal VK. Energy efficient global optimisation of reactive dividing wall distillation column. *Indian Chemical Engineer*. 2020;62(1):15-27. doi:10.1080/00194506.2019.1623089

180. Kiss AA. Novel Catalytic Reactive Distillation Processes for a Sustainable Chemical Industry. *Top Catal*. 2019;62(17):1132-1148. doi:10.1007/s11244-018-1052-9

181. Segovia-Hernández JG, Sánchez-Ramírez E, Alcocer-García H, Quíroz-Ramírez JJ, Udugama IA, Mansouri SS. Analysis of intensified sustainable schemes for biobutanol purification. *Chemical Engineering and Processing - Process Intensification*. 2020;147:107737. doi:10.1016/j.cep.2019.107737

182. Kiss AA, Jobson M, Gao X. Reactive Distillation: Stepping Up to the Next Level of Process Intensification. *Ind Eng Chem Res*. 2019;58(15):5909-5918. doi:10.1021/acs.iecr.8b05450

183. Almeida-Rivera CP, Swinkels PLJ, Grievink J. Designing reactive distillation processes: present and future. *Computers & Chemical Engineering*. 2004;28(10):1997-2020. doi:10.1016/j.compchemeng.2004.03.014

184. Segovia-Hernández JG, Hernández S, Bonilla Petriciolet A. Reactive distillation: A review of optimal design using deterministic and stochastic techniques. *Chemical Engineering and Processing: Process Intensification*. 2015;97:134-143. doi:10.1016/j.cep.2015.09.004

185. Tian Y, Demirel SE, Hasan MMF, Pistikopoulos EN. An overview of process systems engineering approaches for process intensification: State of the art. *Chemical Engineering and Processing - Process Intensification*. 2018;133:160-210. doi:10.1016/j.cep.2018.07.014

186. Taylor R, Krishna R. *Multicomponent Mass Transfer*. 1 edition. Wiley-Interscience; 1993.

187. Taylor R, Krishna R, Kooijman H. Real world modeling of distillation. *Chemical Engineering Progress*. 2003;99(7):28-39.

188. Springer PAM, Baur R, Krishna R. Composition Trajectories for Heterogeneous Azeotropic Distillation in a Bubble-Cap Tray Column: Influence of Mass Transfer. *Chemical Engineering Research and Design*. 2003;81(4):413-426. doi:10.1205/026387603765173682

189. Al-Janabi NKM. *Simulation and Experimental Investigation to Produce Ethyl Tert-Butyl Ether Using Reactive Distillation*. M.Sc. Thesis. College of Engineering of Nahrain University; 2010. https://nahrainuniv.edu.iq/en/node/6760

190. Buchaly C, Kreis P, Górak A. Hybrid separation processes—Combination of reactive distillation with membrane separation. *Chemical Engineering and Processing: Process Intensification*. 2007;46(9):790-799. doi:10.1016/j.cep.2007.05.023

191. He J, Xu B, Zhang W, Zhou C, Chen X. Experimental study and process simulation of n-butyl acetate produced by transesterification in a catalytic distillation column. *Chemical Engineering and Processing: Process Intensification*. 2010;49(1):132-137. doi:10.1016/j.cep.2009.12.004

192. Higler A, Krishna R, Taylor R. Nonequilibrium Modeling of Reactive Distillation: A Dusty Fluid Model for Heterogeneously Catalyzed Processes. *Ind Eng Chem Res*. 2000;39(6):1596-1607. doi:10.1021/ie990547q

193. Klöker M, Kenig EY, Hoffmann A, Kreis P, Górak A. Rate-based modelling and simulation of reactive separations in gas/vapour–liquid systems. *Chemical Engineering and Processing: Process Intensification*. 2005;44(6):617-629. doi:10.1016/j.cep.2003.12.011

194. Rouzineau D, Meyer M, Prevost M. Non equilibrium model and experimental validation for reactive distillation. In: Kraslawski A, Turunen I, eds. *Computer Aided Chemical Engineering*. Vol 14. European Symposium on Computer Aided Process Engineering-13. Elsevier; 2003:893-898. doi:10.1016/S1570-7946(03)80230-4

195. Singh D, Gupta RK, Kumar V. Simulation studies on homogenously catalyzed finishing reactive distillation for ethyl acetate production. *Chemical Engineering Communications*. 2020;207(1):109-122. doi:10.1080/00986445.2019.1574766

196. Sundmacher K, Hoffmann U. Development of a new catalytic distillation process for fuel ethers via a detailed nonequilibrium model. *Chemical Engineering Science*. 1996;51(10):2359-2368. doi:10.1016/0009-2509(96)00092-9

197. Xu Y, Zheng Y, Ng FTT, Rempel GL. A three-phase nonequilibrium dynamic model for catalytic distillation. *Chemical Engineering Science*. 2005;60(20):5637-5647. doi:10.1016/j.ces.2005.05.025

198. Zheng Y, Ng FTT, Rempel GL. Process Analysis for the Production of Diacetone Alcohol via Catalytic Distillation. *Ind Eng Chem Res*. 2003;42(17):3962-3972. doi:10.1021/ie0207921

199. Zheng Y, Ng FTT, Rempel GL. Catalytic Distillation: A Three-Phase Nonequilibrium Model for the Simulation of the Aldol Condensation of Acetone. *Ind Eng Chem Res*. 2001;40(23):5342-5349. doi:10.1021/ie001104l

200. Mencarelli L, Chen Q, Pagot A, Grossmann IE. A review on superstructure optimization approaches in process system engineering. *Computers & Chemical Engineering*. 2020;136:106808. doi:10.1016/j.compchemeng.2020.106808

201. Gómez JM. *Optimisation numérique du fonctionnement, du dimensionnement et de la structure d'une colonne de distillation catalytique représentée par un modèle de transfert*. Ph.D. Thesis. L'Université de Pau et des Pays de l'Adour; 2005.

202. González-Rugerio CA, Fuhrmeister R, Sudhoff D, Pilarczyk J, Górak A. Optimal design of catalytic distillation columns: A case study on synthesis of TAEE. *Chemical Engineering Research and Design*. 2014;92(3):391-404. doi:10.1016/j.cherd.2013.08.030

203. Gómez-Castro FI, Segovia-Hernández JG, Hernández S, Gutiérrez-Antonio C, Briones-Ramírez A, Gamiño-Arroyo Z. Design of non-equilibrium stage separation systems by a stochastic optimization approach for a class of mixtures. *Chemical Engineering and Processing: Process Intensification*. 2015;88:58-69. doi:10.1016/j.cep.2014.11.001

204. Smith R. *Chemical Process: Design and Integration*. 1 edition. Wiley; 2005.

205. Dalaouti N, Seferlis P. A unified modeling framework for the optimal design and dynamic simulation of staged reactive separation processes. *Computers & Chemical Engineering*. 2006;30(8):1264-1277. doi:10.1016/j.compchemeng.2006.02.018

206. Damartzis T, Seferlis P. Optimal Design of Staged Three-Phase Reactive Distillation Columns Using Nonequilibrium and Orthogonal Collocation Models. *Ind Eng Chem Res*. 2010;49(7):3275-3285. doi:10.1021/ie901260b

207. Seferlis P, Grievink J. Optimal Design and Sensitivity Analysis of Reactive Distillation Units Using Collocation Models. *Ind Eng Chem Res*. 2001;40(7):1673-1685. doi:10.1021/ie0005093

208. Gorak A, Sorensen E. *Distillation: Fundamentals and Principles*. Elsevier Science; 2014.

209. Kenig EY. Multicomponent multiphase film-like systems: A modelling approach. *Computers & Chemical Engineering*. 1997;21:S355-S360. doi:10.1016/S0098-1354(97)87527-8

210. Gazzani M, Manzolini G, Macchi E, Ghoniem AF. Reduced order modeling of the Shell–Prenflo entrained flow gasifier. *Fuel*. 2013;104:822-837. doi:10.1016/j.fuel.2012.06.117

211. Hossein Sahraei M, Duchesne MA, Yandon R, Majeski A, Hughes RW, Ricardez-Sandoval LA. Reduced order modeling of a short-residence time gasifier. *Fuel*. 2015;161:222-232. doi:10.1016/j.fuel.2015.07.096

212. Monaghan RFD, Ghoniem AF. A dynamic reduced order model for simulating entrained flow gasifiers: Part I: Model development and description. *Fuel*. 2012;91(1):61-80. doi:10.1016/j.fuel.2011.07.015

213. Sahraei MH, Duchesne MA, Hughes RW, Ricardez-Sandoval LA. Dynamic reduced order modeling of an entrained-flow slagging gasifier using a new recirculation ratio correlation. *Fuel*. 2017;196:520-531. doi:10.1016/j.fuel.2017.01.079

214. Ciric AR, Gu D. Synthesis of nonequilibrium reactive distillation processes by MINLP optimization. *AIChE Journal*. 1994;40(9):1479-1487. doi:10.1002/aic.690400907

215. Kooijman HA, Taylor R. Modelling mass transfer in multicomponent distillation. *The Chemical Engineering Journal and the Biochemical Engineering Journal*. 1995;57(2):177-188. doi:10.1016/0923-0467(94)02952-0

216. Taylor R, Krishna R. Modelling reactive distillation. *Chemical Engineering Science*. 2000;55(22):5183-5229. doi:10.1016/S0009-2509(00)00120-2

217. Taylor R, Kooijman HA. Chapter 3 - Mass Transfer in Distillation. In: Górak A, Sorensen E, eds. *Distillation*. Academic Press; 2014:97-143. doi:10.1016/B978-0-12-386547-2.00003-X

218. Taylor R, Krishna R. Modeling of Homogeneous and Heterogeneous Reactive Distillation Processes. In: *Reactive Distillation*. John Wiley & Sons, Ltd; 2003:215-240. doi:10.1002/3527600523.ch9

219. Froment GF, Bischoff KB, Wilde JD. *Chemical Reactor Analysis and Design*. 3 edition. Wiley; 2010.

220. Higler AP, Taylor R, Krishna R. Nonequilibrium modelling of reactive distillation: Multiple steady states in MTBE synthesis. *Chemical Engineering Science*. 1999;54(10):1389-1395. doi:10.1016/S0009-2509(99)00056-1

221. Sundmacher K, Zhang RS, Hoffmann U. Mass transfer effects on kinetics of nonideal liquid phase ethyl tert-butyl ether formation. *Chemical Engineering & Technology*. 1995;18(4):269-277. doi:10.1002/ceat.270180408

222. Syed FH, Datta R. Nonideal Liquid-Phase Intraparticle Transport and Reaction. *Ind Eng Chem Res*. 2002;41(7):1754-1762. doi:10.1021/ie010407s

223. Thiel C, Sundmacher K, Hoffmann U. Residue curve maps for heterogeneously catalysed reactive distillation of fuel ethers MTBE and TAME. *Chemical Engineering Science*. 1997;52(6):993-1005. doi:10.1016/S0009-2509(96)00454-X

224. Thiel C, Sundmacher K, Hoffmann U. Synthesis of ETBE: Residue curve maps for the heterogeneously catalysed reactive distillation process. *Chemical Engineering Journal*. 1997;66(3):181-191. doi:10.1016/S1385-8947(96)03173-7

225. Kooijman HA, Taylor R. *The ChemSep Book*. H.A. Kooijman and R. Taylor; 2000.

226. Taylor R, Kooijman HA. Composition Derivatives of Activity Coefficient Models (for the Estimation of Thermodynamic Factors in Diffusion). *Chemical Engineering Communications*. 1991;102(1):87-106. doi:10.1080/00986449108910851

227. Poling BE, Prausnitz JM, O'Connell JP. *The Properties of Gases and Liquids*. 5 edition. McGraw-Hill Education; 2000.

228. Moulijn JA, van Leeuwen PWNM, van Santen RA, eds. Chapter 3 Chemical Kinetics of Catalysed Reactions. In: *Studies in Surface Science and Catalysis*. Vol 79. Catalysis. Elsevier; 1993:69-86. doi:10.1016/S0167-2991(08)63807-0

229. Caetano NS, Loureiro JM, Rodrigues AE. MTBE synthesis catalysed by acid ion exchange resins: Kinetic studies and modeling of multiphase batch reactors. *Chemical Engineering Science*. 1994;49(24, Part A):4589-4604. doi:10.1016/S0009-2509(05)80043-0

230. Albright L, ed. *Albright's Chemical Engineering Handbook*. 1 edition. CRC Press; 2008.

231. Liu J, Yang B, Lu S, Yi C. Multi-scale study of reactive distillation. *Chemical Engineering Journal*. 2013;225:280-291. doi:10.1016/j.cej.2013.03.046

232. Barttfeld M, Aguirre PA, Grossmann IE. A decomposition method for synthesizing complex column configurations using tray-by-tray GDP models. *Computers & Chemical Engineering*. 2004;28(11):2165-2188. doi:10.1016/j.compchemeng.2004.03.006

233. Grossmann IE, Trespalacios F. Systematic modeling of discrete-continuous optimization models through generalized disjunctive programming. *AIChE Journal*. 2013;59(9):3276-3295. doi:10.1002/aic.14088

234. Jackson JR, Grossmann IE. A disjunctive programming approach for the optimal design of reactive distillation columns. *Computers & Chemical Engineering*. 2001;25(11):1661-1673. doi:10.1016/S0098-1354(01)00730-X

235. Yeomans H, Grossmann IE. Optimal Design of Complex Distillation Columns Using Rigorous Tray-by-Tray Disjunctive Programming Models. *Ind Eng Chem Res*. 2000;39(11):4326-4335. doi:10.1021/ie0001974

236. Bernal DE, Carrillo-Diaz C, Gómez JM, Ricardez-Sandoval LA. Simultaneous Design and Control of Catalytic Distillation Columns Using Comprehensive Rigorous Dynamic Models. *Industrial & Engineering Chemistry Research*. 2018;57(7):2587-2608. doi:10.1021/acs.iecr.7b04205

237. Murota K. On Steepest Descent Algorithms for Discrete Convex Functions. *SIAM J Optim*. 2004;14(3):699-707. doi:10.1137/S1052623402419005

238. Türkay M, Grossmann IE. Logic-based MINLP algorithms for the optimal synthesis of process networks. *Computers & Chemical Engineering*. 1996;20(8):959-978. doi:10.1016/0098-1354(95)00219-7

239. Sneesby MG, Tadé MO, Datta R, Smith TN. ETBE Synthesis via Reactive Distillation. 1. Steady-State Simulation and Design Aspects. *Ind Eng Chem Res*. 1997;36(5):1855-1869. doi:10.1021/ie960283x

240. Luyben WL, Yu CC. *Reactive Distillation Design and Control*. Wiley-AIChE; 2008.

241. Miranda M, Reneaume JM, Meyer X, Meyer M, Szigeti F. Integrating process design and control: An application of optimal control to chemical processes. *Chemical Engineering and Processing: Process Intensification*. 2008;47(11):2004-2018. doi:10.1016/j.cep.2007.11.010

242. Bussieck MR, Meeraus A. General Algebraic Modeling System (GAMS). In: Kallrath J, ed. *Modeling Languages in Mathematical Optimization*. Applied Optimization. Springer US; 2004:137-157. doi:10.1007/978-1-4613-0215-5_8

243. Baur R, Higler AP, Taylor R, Krishna R. Comparison of equilibrium stage and nonequilibrium stage models for reactive distillation. *Chemical Engineering Journal*. 2000;76(1):33-47. doi:10.1016/S1385-8947(99)00114-X

244. Baur R, Krishna R. Hardware selection and design aspects for reactive distillation columns. A case study on synthesis of TAME. *Chemical Engineering and Processing: Process Intensification*. 2002;41(5):445-462. doi:10.1016/S0255-2701(01)00166-0

245. Ugray Z, Lasdon L, Plummer JC, Bussieck M. Dynamic filters and randomized drivers for the multi-start global optimization algorithm MSNLP. *Optimization Methods and Software*. 2009;24(4-5):635-656. doi:10.1080/10556780902912389

246. Farkas T, Rev E, Lelkes Z. Structural multiplicity and redundancy in chemical process synthesis with MINLP. In: Barbosa-Póvoa A, Matos H, eds. *Computer Aided Chemical Engineering*. Vol 18. European Symposium on Computer-Aided Process Engineering-14. Elsevier; 2004:403-408. doi:10.1016/S1570-7946(04)80133-0

247. Drud A, ARKI Consulting and Development A/S. CONOPT. 2020. Accessed May 31, 2019. https://www.gams.com/latest/docs/S_CONOPT.html

248. Peccini A, Jesus LFS, Secchi AR, Bagajewicz MJ, Costa ALH. Globally optimal distillation column design using set trimming and enumeration techniques. *Computers & Chemical Engineering*. 2023;174:108254. doi:10.1016/j.compchemeng.2023.108254

249. Bonami P, Kilinç M, Linderoth J. Algorithms and Software for Convex Mixed Integer Nonlinear Programs. In: Lee J, Leyffer S, eds. *Mixed Integer Nonlinear Programming*. The IMA Volumes in Mathematics and its Applications. Springer New York; 2012:1-39.

250. Lastusilta T. GAMS MINLP Solver comparisons and some improvements to the AlphaECP algorithm. Published online December 2, 2011. Accessed August 24, 2020. https://www.doria.fi/handle/10024/72508

251. Westerlund T, Lundqvist K. *Alpha-ECP, Version 5.101: An Interactive MINLP-Solver Based on the Extended Cutting Plane Method*. Process Design Laboratory, Abo Akademi University; 2005. URL http://www.abo.fi/~twesterl/A-ECPManual.pdf

252. Kocis GR, Grossmann IE. Computational experience with dicopt solving MINLP problems in process systems engineering. *Computers & Chemical Engineering*. 1989;13(3):307-315. doi:10.1016/0098-1354(89)85008-2

253. Bussieck M, Drud A. SBB: A New Solver for Mixed Integer Nonlinear Programming. Presented at: 2001; OR2001.

254. Kılınç MR, Sahinidis NV. Exploiting integrality in the global optimization of mixed-integer nonlinear programming problems with BARON. *Optimization Methods and Software*. 2018;33(3):540-562. doi:10.1080/10556788.2017.1350178

255. Sahinidis NV. BARON: A general purpose global optimization software package. *J Glob Optim*. 1996;8(2):201-205. doi:10.1007/BF00138693

256. Tawarmalani M, Sahinidis NV. A polyhedral branch-and-cut approach to global optimization. *Math Program*. 2005;103(2):225-249. doi:10.1007/s10107-005-0581-8

257. Lin Y, Schrage L. The global solver in the LINDO API. *Optimization Methods and Software*. 2009;24(4-5):657-668. doi:10.1080/10556780902753221

258. Achterberg T. SCIP: solving constraint integer programs. *Math Prog Comp*. 2009;1(1):1-41. doi:10.1007/s12532-008-0001-1

259. Gamrath G, Anderson D, Bestuzheva K, et al. The SCIP Optimization Suite 7.0. Published online 2020. Accessed August 24, 2020. https://research.tue.nl/en/publications/the-scip-optimization-suite-70

260. Vigerske S, Gleixner A. SCIP: global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. *Optimization Methods and Software*. 2018;33(3):563-593. doi:10.1080/10556788.2017.1335312

261. Ghouse JH, Chen Q, Zamarripa MA, et al. A comparative study between GDP and NLP formulations for conceptual design of distillation columns. In: Eden MR, Ierapetritou MG, Towler GP, eds. *Computer Aided Chemical Engineering*. Vol 44. 13 International Symposium on Process Systems Engineering (PSE 2018). Elsevier; 2018:865-870. doi:10.1016/B978-0-444-64241-7.50139-7

262. Li X, Tomasgard A, Barton PI. Nonconvex Generalized Benders Decomposition for Stochastic Separable Mixed-Integer Nonlinear Programs. *J Optim Theory Appl*. 2011;151(3):425. doi:10.1007/s10957-011-9888-1

263. Li C, Grossmann IE. A generalized Benders decomposition-based branch and cut algorithm for two-stage stochastic programs with nonconvex constraints and mixed-binary first and second stage variables. *J Glob Optim*. 2019;75(2):247-272. doi:10.1007/s10898-019-00816-8

264. Füllner C, Rebennack S. Non-convex nested Benders decomposition. *Math Program*. 2022;196(1):987-1024. doi:10.1007/s10107-021-01740-0

265. Franke MB. Mixed-integer optimization of distillation sequences with Aspen Plus: A practical approach. *Computers & Chemical Engineering*. 2019;131:106583. doi:10.1016/j.compchemeng.2019.106583

266. Bagajewicz MJ, Manousiouthakis V. On the Generalized Benders Decomposition. *Computers & Chemical Engineering*. 1991;15(10):691-700. doi:10.1016/0098-1354(91)85015-M

267. Sahinidis NV, Grossmann IE. Convergence properties of generalized benders decomposition. *Computers & Chemical Engineering*. 1991;15(7):481-491. doi:10.1016/0098-1354(91)85027-R

268. Li X, Chen Y, Barton PI. Nonconvex Generalized Benders Decomposition with Piecewise Convex Relaxations for Global Optimization of Integrated Process Design and Operation Problems. *Ind Eng Chem Res*. 2012;51(21):7287-7299. doi:10.1021/ie201262f

269. Hong J, Van Duc Long N, Harvianto GR, Haider J, Lee M. Design and optimization of multi-effect-evaporation-assisted distillation configuration for recovery of 2,3-butanediol from fermentation broth. *Chemical Engineering and Processing - Process Intensification*. 2019;136:107-115. doi:10.1016/j.cep.2019.01.002

270. Fischer KL, Freund H. Intensification of load flexible fixed bed reactors by optimal design of staged reactor setups. *Chemical Engineering and Processing - Process Intensification*. 2021;159:108183. doi:10.1016/j.cep.2020.108183

271. Okampo EJ, Nwulu N. Optimisation of renewable energy powered reverse osmosis desalination systems: A state-of-the-art review. *Renewable and Sustainable Energy Reviews*. 2021;140:110712. doi:10.1016/j.rser.2021.110712

272. Contreras-Zarazúa G, Jasso-Villegas ME, Ramírez-Márquez C, Sánchez-Ramírez E, Vázquez-Castillo JA, Segovia-Hernandez JG. Design and intensification of distillation processes for furfural and co-products purification considering economic, environmental, safety and control issues. *Chemical Engineering and Processing - Process Intensification*. 2021;159:108218. doi:10.1016/j.cep.2020.108218

273. Lee HY, Chen YY, Eiamsuttitam P, Alcántara-Avila JR. Comparison of optimization methods for the design and control of reactive distillation with inter condensers. *Computers & Chemical Engineering*. 2022;164:107871. doi:10.1016/j.compchemeng.2022.107871

274. Benders JF. Partitioning procedures for solving mixed-variables programming problems. *Numer Math*. 1962;4(1):238-252. doi:10.1007/BF01386316

275. Chen X, Li M. Discrete Convex Analysis and Its Applications in Operations: A Survey. *Production and Operations Management*. 2021;30(6):1904-1926. doi:10.1111/poms.13234

276. Raidl GR, Baumhauer T, Hu B. Speeding Up Logic-Based Benders' Decomposition by a Metaheuristic for a Bi-Level Capacitated Vehicle Routing Problem. In: Blesa MJ, Blum C, Voß S, eds. *Hybrid Metaheuristics*. Lecture Notes in Computer Science. Springer International Publishing; 2014:183-197. doi:10.1007/978-3-319-07644-7_14

277. Raidl GR, Baumhauer T, Hu B. Boosting an Exact Logic-Based Benders Decomposition Approach by Variable Neighborhood Search. *Electronic Notes in Discrete Mathematics*. 2015;47:149-156. doi:10.1016/j.endm.2014.11.020

278. Iimura T, Murota K, Tamura A. Discrete fixed point theorem reconsidered. *Journal of Mathematical Economics*. 2005;41(8):1030-1036. doi:10.1016/j.jmateco.2005.03.001

279. Simionescu PA. A collection of bivariate nonlinear optimisation test problems with graphical representations. *International Journal of Mathematical Modelling and Numerical Optimisation*.

Published online September 1, 2020. Accessed June 28, 2022. https://www.inderscienceonline.com/doi/10.1504/IJMMNO.2020.110704

280. Zhang X, Song Z, Zhou T. Rigorous design of reaction-separation processes using disjunctive programming models. *Computers & Chemical Engineering*. 2018;111:16-26. doi:10.1016/j.compchemeng.2017.12.013

281. Contreras-Zarazúa G, Martin-Martin M, Sánchez-Ramirez E, Segovia-Hernández JG. Furfural production from agricultural residues using different intensified separation and pretreatment alternatives. Economic and environmental assessment. *Chemical Engineering and Processing - Process Intensification*. 2022;171:108569. doi:10.1016/j.cep.2021.108569

282. González-Navarrete C, Sánchez-Ramírez E, Ramírez-Márquez C, Hernández S, Cossío-Vargas E, Segovia-Hernández JG. Innovative Reactive Distillation Process for the Sustainable Purification of Lactic Acid. *Ind Eng Chem Res*. 2022;61(1):621-637. doi:10.1021/acs.iecr.1c04050

283. Rangaiah GP, ed. *Multi-Objective Optimization: Techniques and Applications in Chemical Engineering*. Har/Cdr edition. World Scientific Publishing Company; 2008.

284. Rangaiah GP, Feng Z, Hoadley AF. Multi-Objective Optimization Applications in Chemical Process Engineering: Tutorial and Review. *Processes*. 2020;8(5):508. doi:10.3390/pr8050508

285. Romero-García AG, Mora-Morales C, Chargoy-Amador JP, Ramírez-Corona N, Sánchez-Ramírez E, Segovia-Hernández JG. Implementing CO2 capture process in power plants: Optimization procedure and environmental impact. *Chemical Engineering Research and Design*. 2022;180:232-242. doi:10.1016/j.cherd.2022.02.023

286. Sánchez-Ramírez E, Hernández S, Romero-García AG, Alcocer-García H, Segovia – Hernández JG. Synthesis and Optimization of Sustainable Processes Based on Liquid-Liquid Extraction to Purify Methyl Ethyl Ketone. *Chemical Engineering and Processing - Process Intensification*. 2022;171:108522. doi:10.1016/j.cep.2021.108522

287. Wang Z, Parhi SS, Rangaiah GP, Jana AK. Analysis of Weighting and Selection Methods for Pareto-Optimal Solutions of Multiobjective Optimization in Chemical Engineering Applications. *Ind Eng Chem Res*. 2020;59(33):14850-14867. doi:10.1021/acs.iecr.0c00969

288. Wang Z, Rangaiah GP. Application and Analysis of Methods for Selecting an Optimal Solution from the Pareto-Optimal Front obtained by Multiobjective Optimization. *Ind Eng Chem Res*. 2017;56(2):560-574. doi:10.1021/acs.iecr.6b03453

289. Zielinski K, Laur R. Stopping Criteria for Differential Evolution in Constrained Single-Objective Optimization. In: Chakraborty UK, ed. *Advances in Differential Evolution*. Studies in Computational Intelligence. Springer; 2008:111-138. doi:10.1007/978-3-540-68830-3_4

290. Deb K. *Multi-Objective Optimization Using Evolutionary Algorithms*. 1st edition. Wiley; 2009.

291. Spall JC. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. 1st edition. Wiley-Interscience; 2003.

292. Chen X, Li M. M♮-Convexity and Its Applications in Operations. *Operations Research*. 2021;69(5):1396-1408. doi:10.1287/opre.2020.2070

293. Murota K, Tamura A. Recent progress on integrally convex functions. *Japan J Indust Appl Math*. Published online April 27, 2023. doi:10.1007/s13160-023-00589-4

294. Zhang H, Zheng Z, Lavaei J. Gradient-Based Algorithms for Convex Discrete Optimization via Simulation. *Operations Research*. Published online April 28, 2022. doi:10.1287/opre.2022.2295

295. Srinivas M, Rangaiah GP. Differential Evolution with Tabu List for Global Optimization: Evaluation of Two Versions on Benchmark and Phase Stability Problems. In: *Differential Evolution in Chemical Engineering*. Vol Volume 6. Advances in Process Systems Engineering. World Scientific; 2016:91-127. doi:10.1142/9789813207523_0004

296. Sharma S, Rangaiah GP, Cheah KS. Multi-objective optimization using MS Excel with an application to design of a falling-film evaporator system. *Food and Bioproducts Processing*. 2012;90(2):123-134. doi:10.1016/j.fbp.2011.02.005

297. Srinivas M, Rangaiah GP. Differential Evolution with Tabu List for Solving Nonlinear and Mixed-Integer Nonlinear Programming Problems. *Ind Eng Chem Res*. 2007;46(22):7126-7135. doi:10.1021/ie070007q

298. Storn R. On the usage of differential evolution for function optimization. In: *Proceedings of North American Fuzzy Information Processing*. ; 1996:519-523. doi:10.1109/NAFIPS.1996.534789

299. Douglas J. *Conceptual Design of Chemical Processes*. 1st edition. McGraw-Hill Science/Engineering/Math; 1988.

300. Contreras-Zarazúa G, Sánchez-Ramírez E, Vázquez-Castillo JA, et al. Inherently Safer Design and Optimization of Intensified Separation Processes for Furfural Production. *Ind Eng Chem Res*. 2019;58(15):6105-6120. doi:10.1021/acs.iecr.8b03646

301. Carlson EC. Don't Gamble With Physical Properties For Simulations. *Chemical Engineering Progress*. 1996;92(10):35-46.

302. Barker G. Chapter 11 - Towers. In: Barker G, ed. *The Engineer's Guide to Plant Layout and Piping Design for the Oil and Gas Industries*. 1st ed. Gulf Professional Publishing; 2018:285-308. doi:10.1016/B978-0-12-814653-8.00011-4

303. Torres-Vinces L, Contreras-Zarazua G, Huerta-Rosas B, Sánchez-Ramírez E, Segovia-Hernández JG. Methyl Ethyl Ketone Production through an Intensified Process. *Chemical Engineering & Technology*. 2020;43(7):1433-1441. doi:10.1002/ceat.201900664

304. Kim M, Lee YH, Han IS, Han C. Clustering-Based Hybrid Soft Sensor for an Industrial Polypropylene Process with Grade Changeover Operation. *Ind Eng Chem Res*. 2005;44(2):334-342. doi:10.1021/ie049803b

305. Liu Y, Chen J. Integrated soft sensor using just-in-time support vector regression and probabilistic analysis for quality prediction of multi-grade processes. *Journal of Process Control*. 2013;23(6):793-804. doi:10.1016/j.jprocont.2013.03.008

306. Choi HK, Son SH, Sang-Il Kwon J. Inferential Model Predictive Control of Continuous Pulping under Grade Transition. *Ind Eng Chem Res*. 2021;60(9):3699-3710. doi:10.1021/acs.iecr.0c06216

307. Baldea M, Edgar TF. Dynamic process intensification. *Current Opinion in Chemical Engineering*. 2018;22:48-53. doi:10.1016/j.coche.2018.08.003

308. Skiborowski M. Process synthesis and design methods for process intensification. *Current Opinion in Chemical Engineering*. 2018;22:216-225. doi:10.1016/j.coche.2018.11.004

309. Swartz CLE, Kawajiri Y. Design for dynamic operation - A review and new perspectives for an increasingly dynamic plant operating environment. *Computers & Chemical Engineering*. 2019;128:329-339. doi:10.1016/j.compchemeng.2019.06.002

310. Shah MR. Intensification of polyester synthesis by continuous reactive distillation. Published online 2011. doi:10.6100/IR718721

311. Biegler LT. Integrated Optimization Strategies for Dynamic Process Operations. *Theor Found Chem Eng*. 2017;51(6):910-927. doi:10.1134/S004057951706001X

312. Shao F, Li X, Zhu L, Gong H, Chen X. Optimal Scheduling of the Multigrade Parallel Distillation Column System with a Continuous-Time Formulation. *Ind Eng Chem Res*. 2019;58(51):23225-23237. doi:10.1021/acs.iecr.9b04258

313. Damodaran S, Chioua M. Monitoring of transitions in multi-grade continuous processes. In: *2015 European Control Conference (ECC)*. ; 2015:3635-3640. doi:10.1109/ECC.2015.7331095

314. Liu J, Liu T, Chen J. Quality prediction for multi-grade processes by just-in-time latent variable modeling with integration of common and special features. *Chemical Engineering Science*. 2018;191:31-41. doi:10.1016/j.ces.2018.06.035

315. Subramanian S, Uztürk D, Georgakis C. An Optimization-Based Approach for the Operability Analysis of Continuously Stirred Tank Reactors. *Ind Eng Chem Res*. 2001;40(20):4238-4252. doi:10.1021/ie001111+

316. Lima FV, Jia Z, Ierapetritou M, Georgakis C. Similarities and differences between the concepts of operability and flexibility: The steady-state case. *AIChE Journal*. 2010;56(3):702-716. doi:https://doi.org/10.1002/aic.12021

317. Iftakher A, Mansouri SS, Nahid A, et al. Integrated design and control of reactive distillation processes using the driving force approach. *AIChE Journal*. 2021;67(6):e17227. doi:10.1002/aic.17227

318. Tula AK, Wang J, Chen X, Mansouri SS, Gani R. ProCACD: A computer-aided versatile tool for process control. *Computers & Chemical Engineering*. 2020;136:106771. doi:10.1016/j.compchemeng.2020.106771

319. Lee HY, Chen CY, Chen JL, et al. Design and control of diphenyl carbonate reactive distillation process with thermally coupled and heat-integrated stages configuration. *Computers & Chemical Engineering*. 2019;121:130-147. doi:10.1016/j.compchemeng.2018.10.009

320. Tian Y, Pappas I, Burnak B, Katz J, Pistikopoulos EN. Simultaneous design & control of a reactive distillation system – A parametric optimization & control approach. *Chemical Engineering Science*. 2021;230:116232. doi:10.1016/j.ces.2020.116232

321. Keller T, Holtbruegge J, Górak A. Transesterification of dimethyl carbonate with ethanol in a pilot-scale reactive distillation column. *Chemical Engineering Journal*. 2012;180:309-322. doi:10.1016/j.cej.2011.11.072

322. Keller T, Holtbruegge J, Niesbach A, Górak A. Transesterification of Dimethyl Carbonate with Ethanol To Form Ethyl Methyl Carbonate and Diethyl Carbonate: A Comprehensive Study on Chemical Equilibrium and Reaction Kinetics. *Ind Eng Chem Res*. 2011;50(19):11073-11086. doi:10.1021/ie2014982

323. Ramírez-Márquez C, Sánchez-Ramírez E, Quiroz-Ramírez JJ, et al. Dynamic behavior of a multi-tasking reactive distillation column for production of silane, dichlorosilane and monochlorosilane. *Chemical Engineering and Processing: Process Intensification*. 2016;108:125-138. doi:10.1016/j.cep.2016.08.005

324. de la Fuente RLN, Tlacuahuac AF. Optimal Start-Up and Product Transition Policies of a Reactive Distillation Column. *Ind Eng Chem Res*. 2007;46(7):2092-2111. doi:10.1021/ie061312z

325. Medina-Herrera N, Tututi-Avila S, Jiménez-Gutiérrez A, Segovia-Hernández JG. Optimal design of a multi-product reactive distillation system for silanes production. *Computers & Chemical Engineering*. 2017;105:132-141. doi:10.1016/j.compchemeng.2017.01.014

326. Haßkerl D, Lindscheid C, Subramanian S, Diewald P, Tatulea-Codrean A, Engell S. Economics optimizing control of a multi-product reactive distillation process under model uncertainty. *Computers & Chemical Engineering*. 2018;118:25-48. doi:10.1016/j.compchemeng.2018.07.003

327. Haßkerl D, Lindscheid C, Subramanian S, Markert S, Górak A, Engell S. Dynamic Performance Optimization of a Pilot-Scale Reactive Distillation Process by Economics Optimizing Control. *Ind Eng Chem Res*. 2018;57(36):12165-12181. doi:10.1021/acs.iecr.8b01815

328. Couper JR, Penney WR, Fair JR, Walas SM, eds. Chapter 1 - Introduction. In: *Chemical Process Equipment (Second Edition)*. Gulf Professional Publishing; 2005:1-16. doi:10.1016/B978-075067510-9/50033-4

329. Towler G, Sinnott R. Chapter 1 - Introduction to Design. In: Towler G, Sinnott R, eds. *Chemical Engineering Design (Second Edition)*. Butterworth-Heinemann; 2013:3-32. doi:10.1016/B978-0-08-096659-5.00001-8

330. Koller RW, Ricardez-Sandoval LA. A dynamic optimization framework for integration of design, control and scheduling of multi-product chemical processes under disturbance and uncertainty. *Computers & Chemical Engineering*. 2017;106:147-159. doi:10.1016/j.compchemeng.2017.05.007

331. Pohorecki R, Bridgwater J, Gallegos MMRG and C. Process Economics. In: *Chemical Engineering and Chemical Process Technology - Volume IV: Process Development, Modeling, Optimization, Control and Process Management*. EOLSS Publications; 2010.

332. Marlin TE. Chapter 11, Digital Implementation of Process Control. In: *Process Control: Designing Processes and Control Systems for Dynamic Performance*. McGraw-Hill; 1995.

333. Seborg DE, Mellichamp DA, Edgar TF, III FJD. Chapter 17, Digital Sampling, Filtering, and Control. In: *Process Dynamics and Control*. John Wiley & Sons; 2010.

334. White V, Perkins JD, Espie DM. Switchability analysis. *Computers & Chemical Engineering*. 1996;20(4):469-474. doi:10.1016/0098-1354(95)00037-2

335. Sneesby MG, Tadé MO, Smith TN. Multiplicity and Pseudo-Multiplicity in MTBE and ETBE Reactive Distillation. *Chemical Engineering Research and Design*. 1998;76(4):525-531. doi:10.1205/026387698525009

336. Gazzaneo V, Carrasco JC, Vinson DR, Lima FV. Process Operability Algorithms: Past, Present, and Future Developments. *Ind Eng Chem Res*. 2020;59(6):2457-2470. doi:10.1021/acs.iecr.9b05181

337. D7618 Standard Specification for Ethyl Tertiary-Butyl Ether (ETBE) for Blending with Aviation Spark-Ignition Engine Fuel. Published online 2017. doi:10.1520/D7618-13R17

338. D8235 Standard Specification for Ethyl Tertiary-Butyl Ether (ETBE) for Blending with Gasolines for Use as Automotive Spark-Ignition Engine Fuel. Published online 2018. doi:10.1520/D8235-18

339. de Menezes EW, Cataluña R, Samios D, Silva R da. Addition of an azeotropic ETBE/ethanol mixture in eurosuper-type gasolines. *Fuel*. 2006;85(17):2567-2577. doi:10.1016/j.fuel.2006.04.014

340. de Menezes EW, Cataluña R. Optimization of the ETBE (ethyl tert-butyl ether) production process. *Fuel Processing Technology*. 2008;89(11):1148-1152. doi:10.1016/j.fuproc.2008.05.006

341. Jensen KL, Datta R. Ethers from Ethanol. 1. Equilibrium Thermodynamic Analysis of the Liquid-Phase Ethyl tert-Butyl Ether Reaction (ETBE). *Ind Eng Chem Res*. 1995;34(1):392-399. doi:10.1021/ie00040a043

342. Domingues L, Pinheiro CIC, Oliveira NMC. Optimal design of reactive distillation systems: Application to the production of ethyl tert-butyl ether (ETBE). *Computers & Chemical Engineering*. 2014;64:81-94. doi:10.1016/j.compchemeng.2014.01.014

343. Bernal Neira DE, Gómez Ramírez JM, Álvarez Solano ÓA, Ramos Castillo MA. *Optimal Design and Control of a Catalytic Distillation Column : Case Study : Ethyl Tert-Butyl Ether (ETBE) Synthesis Column*. Uniandes; 2016. Accessed May 28, 2021. https://biblioteca.uniandes.edu.co/

344. Biegler LT. 10. Simultaneous Methods for Dynamic Optimization. In: *Nonlinear Programming*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics; 2010:287-324. doi:10.1137/1.9780898719383.ch10

345. Bisowarno BH, Tian YC, Tadé MO. Interaction of separation and reactive stages on ETBE reactive distillation columns. *AIChE Journal*. 2004;50(3):646-653. doi:10.1002/aic.10058

346. Luo Y, Zhang Q, Zhu L, Chen X. Optimal operation of parallel distillation systems with multiple product grades: An industrial case study. *Computers & Chemical Engineering*. 2018;111:210-224. doi:10.1016/j.compchemeng.2018.01.009

347. Peters M, Timmerhaus K, West R, Peters M. Chapter 3 - Process Design Development. In: *Plant Design and Economics for Chemical Engineers*. ; 2002.

348. Luyben WL. Distillation Economic Optimization. In: *Distillation Design and Control Using Aspen™ Simulation*. John Wiley & Sons, Ltd; 2013:81-94. doi:10.1002/9781118510193.ch04

349. Sneesby MG, Tadé MO, Datta R, Smith TN. ETBE Synthesis via Reactive Distillation. 2. Dynamic Simulation and Control Aspects. *Ind Eng Chem Res*. 1997;36(5):1870-1881. doi:10.1021/ie970053y

350. Maiti D, Acharya A, Chakraborty M, Konar A, Janarthanan R. Tuning PID and FOPID Controllers using the Integral Time Absolute Error Criterion. *arXiv:08110083 [cs]*. Published online November 1, 2008. Accessed June 13, 2021. http://arxiv.org/abs/0811.0083

351. Graham D, Lathrop RC. The synthesis of "optimum" transient response: Criteria and standard forms. *Transactions of the American Institute of Electrical Engineers, Part II: Applications and Industry*. 1953;72(5):273-288. doi:10.1109/TAI.1953.6371346

352. Schultz WC, Rideout VC. Control system performance measures: Past, present, and future. *IRE Transactions on Automatic Control*. 1961;AC-6(1):22-35. doi:10.1109/TAC.1961.6429306

353. Barona N. Batch Processing. In: Meyers RA, ed. *Encyclopedia of Physical Science and Technology (Third Edition)*. Academic Press; 2003:41-56. doi:10.1016/B0-12-227410-5/00049-1

354. Moreno-Benito M, Frankl K, Espuña A, Marquardt W. A modeling strategy for integrated batch process development based on mixed-logic dynamic optimization. *Computers & Chemical Engineering*. 2016;94:287-311. doi:10.1016/j.compchemeng.2016.07.030

355. Chu Y, You F. Integration of Scheduling and Dynamic Optimization of Batch Processes under Uncertainty: Two-Stage Stochastic Programming Approach and Enhanced Generalized Benders Decomposition Algorithm. *Ind Eng Chem Res*. 2013;52(47):16851-16869. doi:10.1021/ie402621t

356. Ji N, Gu X. Integration of scheduling and control for the no-wait batch process: A decomposition method. *Chemical Engineering Science*. 2022;260:117939. doi:10.1016/j.ces.2022.117939

357. Santander O, Giannikopoulos I, Stadtherr MA, Baldea M. An Integrated Stochastic Deep Learning–Short-Term Production Scheduling–Optimal Control Framework for General Batch Processes. *Ind Eng Chem Res*. 2023;62(5):2124-2137. doi:10.1021/acs.iecr.2c02638

358. Santander O, Baldea M. Control-aware batch process scheduling. *Computers & Chemical Engineering*. 2021;152:107360. doi:10.1016/j.compchemeng.2021.107360

359. Cao K, Son SH, Moon J, Kwon JSI. A closed-loop integration of scheduling and control for hydraulic fracturing using offset-free model predictive control. *Applied Energy*. 2021;302:117487. doi:10.1016/j.apenergy.2021.117487

360. Kondili E, Pantelides CC, Sargent RWH. A general algorithm for short-term scheduling of batch operations—I. MILP formulation. *Computers & Chemical Engineering*. 1993;17(2):211-227. doi:10.1016/0098-1354(93)80015-F

361. Velez S, Maravelias CT. Reformulations and Branching Methods for Mixed-Integer Programming Chemical Production Scheduling Models. *Ind Eng Chem Res*. 2013;52(10):3832-3841. doi:10.1021/ie303421h

362. Shah N, Pantelides CC, Sargent RWH. A general algorithm for short-term scheduling of batch operations—II. Computational issues. *Computers & Chemical Engineering*. 1993;17(2):229-244. doi:10.1016/0098-1354(93)80016-G

363. Maravelias CT. Mixed-Time Representation for State-Task Network Models. *Ind Eng Chem Res*. 2005;44(24):9129-9145. doi:10.1021/ie0500117

364. Sundaramoorthy A, Maravelias CT. Computational Study of Network-Based Mixed-Integer Programming Approaches for Chemical Production Scheduling. *Ind Eng Chem Res*. 2011;50(9):5023-5040.

365. Velez S, Merchan AF, Maravelias CT. On the solution of large-scale mixed integer programming scheduling models. *Chemical Engineering Science*. 2015;136:139-157. doi:10.1016/j.ces.2015.05.021

366. Nie Y, Biegler LT. Integrated Dynamic Optimization and Scheduling of Polymerization Processes with First Principle Models. *Chemie Ingenieur Technik*. 2017;89(11):1490-1502. doi:10.1002/cite.201700053

367. Srinivasan B, Palanki S, Bonvin D. Dynamic optimization of batch processes: I. Characterization of the nominal solution. *Computers & Chemical Engineering*. 2003;27(1):1-26. doi:10.1016/S0098-1354(02)00116-3

368. Nicholson B, Siirola JD, Watson JP, Zavala VM, Biegler LT. pyomo.dae: a modeling and automatic discretization framework for optimization with differential and algebraic equations. *Math Prog Comp*. 2018;10(2):187-223. doi:10.1007/s12532-017-0127-0

369. Bernal DE, Vigerske S, Trespalacios F, Grossmann IE. Improving the performance of DICOPT in convex MINLP problems using a feasibility pump. *Optimization Methods and Software*. 2020;35(1):171-190. doi:10.1080/10556788.2019.1641498

370. Mendis P, Wickramasinghe C, Narayana M, Bayer C. Adaptive Model Predictive Control with Successive Linearization for Distillate Composition Control in Batch Distillation. In: *2019 Moratuwa Engineering Research Conference (MERCon)*. ; 2019:366-369. doi:10.1109/MERCon.2019.8818777

371. Blömer F, Günther HO. Scheduling of a multi-product batch process in the chemical industry. *Computers in Industry*. 1998;36(3):245-259. doi:10.1016/S0166-3615(98)00075-X

372. Durand GA, Bandoni JA. Improving the Computational Performance of STN/RTN Formulations for Short-Term Scheduling. *Ind Eng Chem Res*. 2010;49(24):12477-12495. doi:10.1021/ie901093y

373. Verderame PM, Floudas CA. Integrated Operational Planning and Medium-Term Scheduling for Large-Scale Industrial Batch Plants. *Ind Eng Chem Res*. 2008;47(14):4845-4860. doi:10.1021/ie8001414

374. Verderame PM, Floudas CA. Integration of Operational Planning and Medium-Term Scheduling for Large-Scale Industrial Batch Plants under Demand and Processing Time Uncertainty. *Ind Eng Chem Res*. 2010;49(10):4948-4965. doi:10.1021/ie901973e

375. Li P, Garcia HA, Wozny G, Reuter E. Optimization of a Semibatch Distillation Process with Model Validation on the Industrial Site. *Ind Eng Chem Res*. 1998;37(4):1341-1350. doi:10.1021/ie970695l

376. Kaptein M, van den Heuvel E. A First Look at Data. In: Kaptein M, van den Heuvel E, eds. *Statistics for Data Scientists : An Introduction to Probability, Statistics, and Data Analysis*.

Undergraduate Topics in Computer Science. Springer International Publishing; 2022:1-37. doi:10.1007/978-3-030-10531-0_1

377. Di Pretoro A, Montastruc L, Joulia X, Manenti F. Accounting for dynamics in flexible process design: A switchability index. *Computers & Chemical Engineering*. 2021;145:107149. doi:10.1016/j.compchemeng.2020.107149

378. Khaledi R, Young BR. Modeling and Model Predictive Control of Composition and Conversion in an ETBE Reactive Distillation Column. *Ind Eng Chem Res*. 2005;44(9):3134-3145.

379. Varelmann T, Lipow AW, Baldea M, Mitsos A. Advanced feasibility cuts in decoupled cooperative optimization of power flow. *Computers & Chemical Engineering*. 2022;157:107635. doi:10.1016/j.compchemeng.2021.107635

380. Floudas CA. *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. Oxford University Press; 1995.

381. Murota K, Shioura A. Relationship of M-/L-convex functions with discrete convex functions by Miller and Favati–Tardella. *Discrete Applied Mathematics*. 2001;115(1):151-176. doi:10.1016/S0166-218X(01)00222-0

382. Ross KA. *Elementary Analysis: The Theory of Calculus*. 2nd ed. 2013 edition. Springer; 2013.

383. Chinneck JW. *Feasibility and Infeasibility in Optimization: Algorithms and Computational Methods*. 2008th edition. Springer; 2007.

384. Srinivas M, Rangaiah GP. Differential Evolution with Tabu List for Global Optimization and Its Application to Phase Equilibrium and Parameter Estimation Problems. *Ind Eng Chem Res*. 2007;46(10):3410-3421. doi:10.1021/ie0612459

385. Uijen WAT. *Flooding in Distillation Columns : Choking and Froth Height Limitation*. Eindhoven University of Technology; 1999. https://research.tue.nl/en/studentTheses/flooding-in-distillation-columns-choking-and-froth-height-limitat

386. Olsen I, Endrestøl GO, Sira T. A rigorous and efficient distillation column model for engineering and training simulators. *Computers & Chemical Engineering*. 1997;21:S193-S198. doi:10.1016/S0098-1354(97)87501-1

387. Ruiz CA, Cameron IT, Gani R. A generalized dynamic model for distillation columns—III. Study of startup operations. *Computers & Chemical Engineering*. 1988;12(1):1-14. doi:10.1016/0098-1354(88)85001-4

388. Hoffmann C, Weigert J, Esche E, Repke JU. A pressure-driven, dynamic model for distillation columns with smooth reformulations for flexible operation. *Computers & Chemical Engineering*. 2020;142:107062. doi:10.1016/j.compchemeng.2020.107062

389. Maravelias CT. General framework and modeling approach classification for chemical production scheduling. *AIChE Journal*. 2012;58(6):1812-1828. doi:10.1002/aic.13801

390. Grossmann IE, Quesada I, Raman R, Voudouris VT. Mixed-Integer Optimization Techniques for the Design and Scheduling of Batch Processes. In: Reklaitis GV, Sunol AK, Rippin DWT, Hortaçsu Ö, eds. *Batch Processing Systems Engineering*. NATO ASI Series. Springer; 1996:451-494. doi:10.1007/978-3-642-60972-5_21

391. Raidl GR. Decomposition based hybrid metaheuristics. *European Journal of Operational Research*. 2015;244(1):66-76. doi:10.1016/j.ejor.2014.12.005

392. Karlsson E, Rönnberg E. Strengthening of Feasibility Cuts in Logic-Based Benders Decomposition. In: Stuckey PJ, ed. *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*. Lecture Notes in Computer Science. Springer International Publishing; 2021:45-61. doi:10.1007/978-3-030-78230-6_3

393. Saharidis GKD, Ierapetritou MG. Improving benders decomposition using maximum feasible subsystem (MFS) cut generation strategy. *Computers & Chemical Engineering*. 2010;34(8):1237-1245. doi:10.1016/j.compchemeng.2009.10.002

394. Lundell A, Kronqvist J, Westerlund T. The supporting hyperplane optimization toolkit for convex MINLP. *J Glob Optim*. 2022;84(1):1-41. doi:10.1007/s10898-022-01128-0

395. Silebi CA, Schiesser WE. *Dynamic Modeling of Transport Process Systems*. 1st edition. Academic Press; 1992.

Appendix A

A Discrete-Steepest Descent Framework for the Simultaneous Process and Control Design of Multigrade Reactive Distillation Columns

Traditional methodologies for the design of chemical processes where design and control considerations are treated separately are evolving into more sophisticated techniques where process dynamics are accounted for in the early conceptual design stages[377]. One process intensification strategy that has gained attention is the integration of reaction and separation into a single reactive distillation (RD) unit, which offers many advantages such as reductions in capital cost, energy integration, improvements in selectivity and conversion, and shift of azeotropic equilibria[138]. Nonetheless, the integration of these processes inherently produces unwanted operational and controllability difficulties due to the susceptibility of the system to state multiplicity, process gain nonlinearity, control interactions, and process gain bidirectionality[349,378]. Thus, different strategies that account for the controllability of RD systems at the design stage are still in development.

In the process industry, a process design is expected to accommodate changes in product specification, production demands, and to efficiently reject disturbances. For this purpose, the expected operating points and closed-loop dynamics of the system must be taken into consideration when designing a RD column. Previous studies have shown that an optimal design identified from steady state calculations may not necessarily have a feasible dynamic performance during changes in the operation. The most common approach to circumvent this issue is to overdesign the system using process heuristics or trial and error procedures that may ultimately lead to a design that is not the most profitable. Thus, comprehensive design strategies are still required to obtain *flexible* RD units, i.e., systems capable of accommodating for disturbances during operation with respect to nominal operating conditions[377].

As discussed in chapter 6, a recent trend in the field of RD is the economic design of systems that are expected to operate at different steady states, i.e., multigrade RD columns. This is because RD columns are often required to sequentially produce different products in a single piece of equipment, which inherently introduces the need to perform dynamic transitions between different steady states. The sequential dynamic transition between different steady states is referred to as *multi-period*

operation. To consider this operation mode, a modular Discrete-Steepest Descent Algorithm (D-SDA) was developed in chapter 6 to identify profitable designs in continuous multi-period RD units. This strategy is based on a master-subproblem decomposition of the original mixed-integer dynamic optimization (MIDO) problem where discrete decisions are solved separately from the continuous design decisions using a discrete analog version of the continuous steepest descent method. That strategy returns an economic RD design (i.e., discrete configuration and internal dimensions) that also accommodates dynamic transitions between the different RD products. Open-loop control actions were performed in that study. To the authors' knowledge, the optimal RD design that considers both multi-period steady-state operation and the dynamic operability in closed-loop for RD columns has not been investigated in the literature.

The aim of this appendix is to apply the modular D-SDA strategy for the simultaneous design and control of a multi-product RD column. The proposed strategy aims to return the optimal process design specifications, process operating conditions and the controller tuning parameters that result in a dynamically feasible and economically optimal operation of a RD system. The features of the proposed strategy are demonstrated through a case study involving dynamic transitions during the production of different grades of ethyl tert-butyl-ether (ETBE). Previous works have only dealt with the simultaneous design and control of RD systems around a nominal operating point, and they rarely incorporate discrete decisions explicitly in the formulation. Hence, this work brings novelty to the current literature since RD systems are required to operate in a flexible manner in closed-loop at different steady-state conditions to guarantee feasible multiproduct/multigrade production and with an adequate disturbance rejection performance. To the authors' knowledge, this is the first study that addresses the simultaneous economic design and control of a multi-period RD system using deterministic optimization.

This study is organized as follows. Section A.1 states the MIDO formulation considered in this work (i.e., the dynamic multi-period problem), and its steady state counterpart (i.e., the steady-state multi-period problem). The modular D-SDA framework is presented in Section A.2. Section A.3 applies the proposed framework to perform the optimal design and control of an ETBE catalytic distillation system under two different scenarios. Concluding remarks and future work are provided at the end.

A.1 Problem statement

This work considers the optimal design and closed-loop operation of a RD column that is expected to change its operating conditions multiple times over a single year of operation. There are n_K periods

throughout the year, where each period features a steady state operation and a dynamic transition that moves the system from one operating point to another. The Equivalent Annual Cost (EAC) is used as the objective function and a superstructure formulation coupled with a nonlinear system of Differential-Algebraic Equations (DAE), which are used to represent the nonlinear behavior of the system through the Mass, Equilibrium, Summation, and enthalpy (MESH) model. PID control equations can also be incorporated into the DAE equations of the model. Binary variables are included to account for the feed/feeds location, number of stages and distribution of reactive stages along the RD column. This work assumes that the operation of the system returns to its initial operating condition after a year of operation. Thus, the year of operation goes from $t_0 = 0$ to $t_p = t_{n_K}$, where t_p is typically set to $8{,}000\ h/year$. The time range $[0, t_p]$ is subdivided into multiple periods (time slots) $[t_0, t_1], [t_1, t_2], \dots, [t_{n_K-1}, t_{n_K}]$. Each period $[t_{k-1}, t_k] \in K = \{1,2, \dots, n_K\}$ includes a dynamic transition (from t_{k-1} to t_k^{TR} (superscript TR denotes the transition time) and the steady-state operating point achieved after the transition occurs (i.e., from t_k^{TR} to t_k). This work considers the objective function $f_{OBJ} = f_{OBJ}^{SS} + f_{OBJ}^{TR}$ derived in chapter 6 and that can be represented as the addition of a steady state term (f_{OBJ}^{SS} in (A.1A)) and a dynamic term (f_{OBJ}^{TR} in (A.1B)), i.e.,

$$f_{OBJ}^{SS} = C_{INV} + \sum_{k\in K}\left[\left(C_{OP}(t_k) - C_{IN}(t_k)\right)(t_k - t_{k-1})\right] \tag{A.1A}$$

$$f_{OBJ}^{TR} = \sum_{k\in K}\left[\int_{t_{k-1}}^{t_k^{TR}} (C_{OP}(t) - C_{OP}(t_k) + C_{IN}(t_k))dt\right] \tag{A.1B}$$

C_{INV} represents the annualized investment costs, C_{OP} is the time-dependent operating costs per unit of time, and C_{IN} accounts for the incomes to the process per unit of time. Based on this objective function, the dynamic multi-period problem is stated in problem (A.2), with constraints summarized in Eqs. (A.2A)-(A.2G). $\boldsymbol{v}^{SS} = \left[\boldsymbol{v}_1^{SS}, \boldsymbol{v}_2^{SS}, \dots, \boldsymbol{v}_{n_K}^{SS}\right]$ contains vectors of the time-dependent variables (including differential ($\boldsymbol{\chi}$), algebraic ($\boldsymbol{\zeta}$) and manipulated variables ($\boldsymbol{\mu}$)) that represent the steady state operation of the system at time t_k, $\forall k \in K$ (see Eq. (A.2B)); $\boldsymbol{v}^{TR}$ is a vector function with time-dependent variables ($\boldsymbol{\chi}, \boldsymbol{\zeta}$ and $\boldsymbol{\mu}$) evaluated during dynamic transitions $\left(t_{k-1}, t_k^{TR}\right], \forall k \in K$ (see Eq. (A.2C)); $\boldsymbol{\varphi}$ is the vector of time independent continuous variables (usually related to continuous design variables); $\boldsymbol{\tau}$ represents the controller tuning parameters, and $\boldsymbol{y}$ is a vector of binary variables, i.e., those that define integer decisions such as the number of stages in an RD column.

$$\min_{\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\tau},\boldsymbol{\varphi},\boldsymbol{y}} f_{OBJ} = f_{OBJ}^{SS} + f_{OBJ}^{TR} \tag{A.2}$$

$s.t.$

$$[\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}] \in S^{SS}, [\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\varphi},\boldsymbol{\tau},\boldsymbol{y}] \in S^{TR}, \boldsymbol{y} \in Y \tag{A.2A}$$

$$\boldsymbol{v}_k^{SS} = [\boldsymbol{\chi}(t_k),\boldsymbol{\mu}(t_k),\boldsymbol{\zeta}(t_k)], \forall k \in K \tag{A.2B}$$

$$\boldsymbol{v}^{TR}(t) = [\boldsymbol{\chi}(t),\boldsymbol{\mu}(t),\boldsymbol{\zeta}(t)], \forall t \in \left(t_{k-1}, t_k^{TR}\right], \forall k \in K \tag{A.2C}$$

$$S^{SS} = \left\{ [\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}]: \begin{matrix} \boldsymbol{g}^D(\boldsymbol{\varphi},\boldsymbol{y}) \le \boldsymbol{0} \\ \begin{bmatrix} \boldsymbol{h}^{SS}\left(\boldsymbol{v}_1^{SS},\boldsymbol{\varphi},\boldsymbol{y}\right) \le \boldsymbol{0} \\ \vdots \\ \boldsymbol{h}^{SS}\left(\boldsymbol{v}_{n_K}^{SS},\boldsymbol{\varphi},\boldsymbol{y}\right) \le \boldsymbol{0} \end{bmatrix} \end{matrix} \right\} \tag{A.2D}$$

$$S^{TR} = \left\{ [\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\varphi},\boldsymbol{\tau},\boldsymbol{y}]: \begin{matrix} \begin{bmatrix} \boldsymbol{h}_1^{TR} \le \boldsymbol{0} \\ \vdots \\ \boldsymbol{h}_{n_K}^{TR} \le \boldsymbol{0} \end{bmatrix} \\ \left(\left.\frac{d\boldsymbol{\chi}}{dt}\right|_{t_k^{TR}}\right)^2 \le \varepsilon_1, \forall k \in K \\ \boldsymbol{\mu}\left(t_k^{TR}\right) = \boldsymbol{\mu}(t_k), \forall k \in K \\ \boldsymbol{f}\left(\boldsymbol{\chi}(t_0),\boldsymbol{\mu}(t_0),\boldsymbol{\zeta}(t_0)\right) = \boldsymbol{0} \end{matrix} \right\} \tag{A.2E}$$

$$\boldsymbol{h}_k^{TR} = \boldsymbol{h}^{TR}\left(\frac{d\boldsymbol{\chi}(t)}{dt}, \boldsymbol{v}^{TR}(t), \boldsymbol{\varphi}, \boldsymbol{\tau}, \boldsymbol{y}\right), \forall t \in \left(t_{k-1}, t_k^{TR}\right] \tag{A.2F}$$

$$Y = \{\boldsymbol{y}: \boldsymbol{\Omega}(\boldsymbol{y}) \le \boldsymbol{0}, \boldsymbol{y} \in \{0,1\}^{n_y}\} \tag{A.2G}$$

The constraints of the problem are included in the feasible regions S^{SS}, S^{TR} and Y. The first subregion S^{SS} in Eq. (A.2D) typically involves equations that are either independent of time or related to the steady state operation of the system, i.e., design constraints that define the dimensions of the unit ($\boldsymbol{g}^D$); and the nonlinear MESH equations at steady state, purity, and demand requirements in $\boldsymbol{h}^{SS}$. The second subregion S^{TR} in Eq. (A.2E) includes process dynamics and control equations $\boldsymbol{h}_k^{TR}$ for every transition k included in the formulation (see Eq. (A.2F)). $\boldsymbol{h}_k^{TR}$ incorporates the dynamic MESH equations, the controller equations, and the anticipated disturbances for each time slot k. Note that the effect of uncertainty and unanticipated disturbances is not considered in the present formulation. Additional constraints are imposed in S^{TR} to define the initial operating point of the system ($\boldsymbol{f} = \boldsymbol{0}$); the switchability constraints $\left(\left(\left.d\boldsymbol{\chi}/dt\right|_{t_k^{TR}}\right)^2 \le \varepsilon_1\right)$ required to stabilize the system using a small value of

ε_1, and continuity constraints ($\boldsymbol{\mu}\left(t_k^{TR}\right) = \boldsymbol{\mu}(t_k)$) to connect consecutive periods. If required, additional constraints to avoid the phenomenon of multiple steady states can be added to S^{TR} (see chapter 6). The last subregion Y in Eq. (A.2G) contains the logic constraints $\boldsymbol{\Omega}(\boldsymbol{y})$ enforced over the n_y binary variables of the problem, e.g., the feed location must be located between the reflux and boil-up stages. The advantage of formulating the problem as shown in problem (A.2) is that, when neglecting the costs associated with the dynamic transitions ($f_{OBJ}^{TR} = 0$), the vector of time dependent dynamic variables ($\boldsymbol{v}^{TR}$) and the constraints associated with the dynamic transitions (S^{TR}) can be ignored. Thus, when f_{OBJ}^{TR} vanishes (i.e., when the profit loss due to dynamic transitions is neglected), the optimization problem reduces to the steady-state multi-period problem (A.3), which solution provides a lower bound for problem (A.2). A detailed description of problem (A.3) and its interaction with problem (A.2) can be found in chapter 6.

$$\min_{\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}} f_{OBJ}^{SS}(\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}) \tag{A.3}$$

$$s.t.$$

$$[\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}] \in S^{SS}, \boldsymbol{y} \in Y$$

where S^{SS} and Y are given by Eq. (A.2D) and Eq. (A.2G), respectively. Optimization problems (A.2) and (A.3) and the relationship between them is the key idea pursued in this work to propose an algorithmic framework to address the simultaneous design and control of multigrade RD columns.

A.2 Methodology

The lowest EAC of a flexible RD column can be obtained in the ideal case when a column has instantaneous dynamics when performing transitions between steady states, as in problem (A.3). This suggests that solving the steady-state optimization problem (A.3) is a good starting point for the multi-period MIDO problem shown in (A.2). Accordingly, a modular strategy depicted in Figure A.1 is proposed to seek for an optimal design and control scheme that can accommodate transient changes in the operation of RD columns. Modules 1 and 2 provide an initialization to Module 3, which solves the process design and control problem simultaneously.

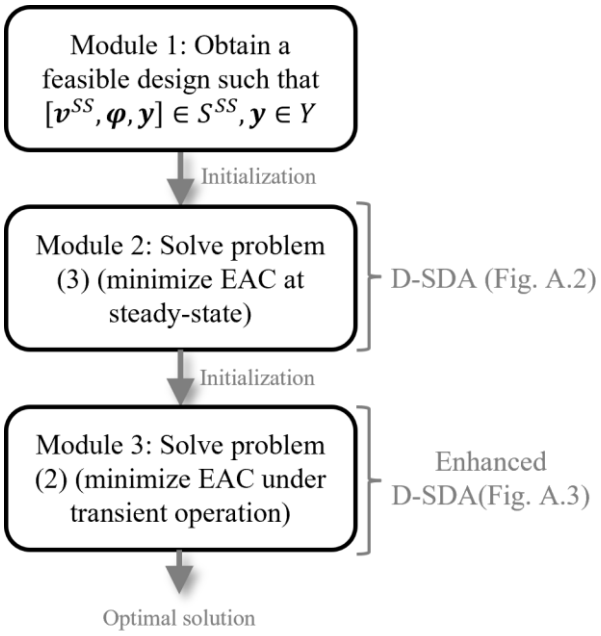


Figure A.1. The modular D-SDA approach

To exemplify the strategy shown in Figure A.1, consider the design of a RD column that must change its operating conditions to produce products A and B in closed-loop. The first module examines the feasibility of producing A and B at steady-state using the same RD equipment defined by $\boldsymbol{\varphi}$ and $\boldsymbol{y}$. This module is included because a RD design that cannot satisfy the desired product specifications/demands should be discarded. If feasibility is achieved, the second module optimizes the EAC of the RD unit at steady state by manipulating the process design variables ($\boldsymbol{\varphi}, \boldsymbol{y}$) and steady-state operation variables $\boldsymbol{v}^{SS}$. This design also corresponds to the best possible EAC that can be achieved in an ideal case when the dynamic transitions are instantaneous. As shown in Figure A.1, this point serves as initialization for the last module where discrete ($\boldsymbol{\varphi}$) and continuous design variables ($\boldsymbol{\varphi}$), steady state operation variables ($\boldsymbol{v}^{SS}$), dynamic operation variables ($\boldsymbol{v}^{TR}$), and controller tuning parameters ($\boldsymbol{\tau}$) are modified to guarantee an optimal dynamic transition from A to B. Hence, this modularization reduces the problem's complexity by sequentially approaching the optimal solution of the multi-period MIDO shown in problem (A.2).

The solution of each module in Figure A.1 is challenging due to the nonlinearities of the phenomenological RD model, the presence of binary variables that multiply continuous terms in the formulation, the numerical difficulties associated with zero flows when a stage is removed/added to

optimize the number of stages, and the control structure which introduces feedback introduced the system. The key idea to address these challenges is the reformulation of the binary decisions $\boldsymbol{y}$ with integer external variables $\boldsymbol{e}$, which explicitly optimize the discrete decisions of the problem in a master optimization layer. For instance, a RD superstructure formulation would typically include multiple binary decisions, e.g., $\boldsymbol{y}_1$, $\boldsymbol{y}_2$ and $\boldsymbol{y}_3$, which specify the number of stages, the feed location and the size of the reactive zone, respectively. In this case, each of these binary vectors can be reformulated with $\boldsymbol{e} = [e_1, e_2, e_3]$ as $\boldsymbol{y}_i = \boldsymbol{y}_i(e_i), \forall i \in \{1,2,3\}$, where e_1 is the integer decision that defines the number of stages, e_2 defines the feed location, and e_3 is the number of stages in the reactive zone. These external variables are explored using the D-SDA and enhanced D-SDA algorithms presented in Figures A.2 and A.3, respectively, which require the solution of the dynamic and steady-state NLP subproblems shown in (A.4A) and (A.4B), by systematically updating $\boldsymbol{e}$ with $\boldsymbol{e}^*$, where $\boldsymbol{e}^*$ denotes the most updated value of the external variables on each iteration.

$$\min_{\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}} f_{OBJ}(\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})) \tag{A.4A}$$

$$s.t.$$

$$[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})] \in S^{SS}, [\boldsymbol{v}^{SS}, \boldsymbol{v}^{TR}, \boldsymbol{\varphi}, \boldsymbol{\tau}, \boldsymbol{y}(\boldsymbol{e})] \in S^{TR}$$

$$\min_{\boldsymbol{v}^{SS}, \boldsymbol{\varphi}} f_{OBJ}^{SS}(\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})) \tag{A.4B}$$

$$s.t.$$

$$[\boldsymbol{v}^{SS}, \boldsymbol{\varphi}, \boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$$

The D-SDA algorithm (Figure A.2) compares the optimal solution of the NLP subproblem (A.4B) at $\boldsymbol{e}^*$ ($f_{OBJ}^{SS}(\boldsymbol{e}^*)$) and the surrounding discrete configurations referred to as the *neighborhood* $N_\infty(\boldsymbol{e}^*) = \{\boldsymbol{e} \in \mathbb{Z}^{n_e}: \|\boldsymbol{e} - \boldsymbol{e}^*\|_\infty \leq 1\}$, where n_e is the number of external variables (see Steps 1 and 2 in Figure A.2). For instance, the neighbors of a single feed located at stage 8 are feed-streams at stages 7 and 6. If a specific neighbor ($\boldsymbol{e}^o$) minimizes the objective function over $N_\infty(\boldsymbol{e}^*)$, it specifies the new steepest descent direction ($\boldsymbol{\delta} = \boldsymbol{e}^o - \boldsymbol{e}^*$) over which the line search is sequentially performed (Step 3 in Figure A.2). The line search starts at $\boldsymbol{e}^*$ and stops when f_{OBJ}^{SS} starts worsening (Step 4 in Figure A.2). At this point, $\boldsymbol{e}^*$ is updated and a new iteration is performed (Step 5 in Figure A.2).

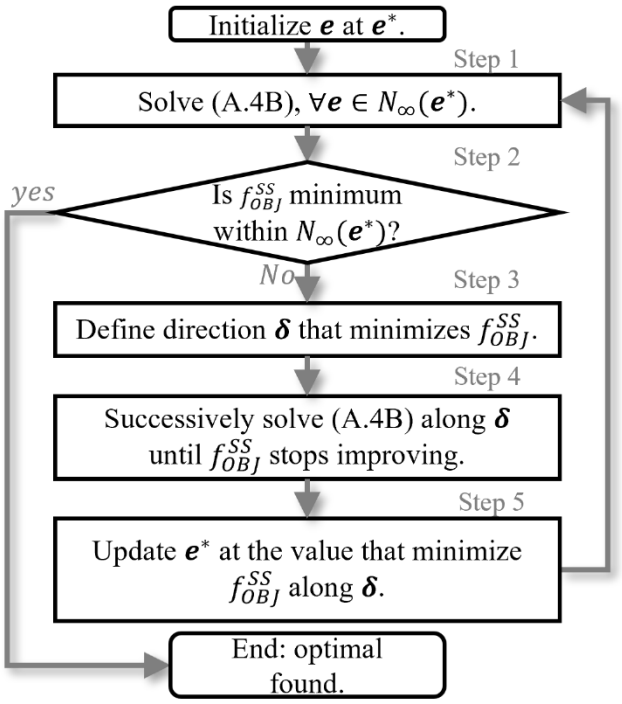


Figure A.2. The D-SDA. $f_{OBJ}^{SS} = +\infty$ for infeasible problems.

The optimal continuous and discrete variables from Figure A.2 are used to initialize the last module in Figure A.1, which explicitly solves the dynamic multi-period problem in (A.2). The steady-state multi-period subproblems (A.4B) converge much faster than dynamic-multi-period subproblems (A.4A) due to their difference in size and nonconvexity. The enhanced D-SDA (Figure A.3) aims to avoid the solution of every dynamic problem based on their steady state information, i.e., problem (A.4B) provides a lower bound for problem (A.4A) for any value of $\boldsymbol{e}$, which suggests that problem (A.4B) should be solved first to decide if problem (A.4A) needs to be solved. This feature is incorporated both in the neighborhood verification steps (i.e., Steps 2 and 3 in Figure A.3) and the line search (Step 7 in Figure A.3). Set $N'_\infty(\boldsymbol{e}^*) = N_\infty(\boldsymbol{e}^*) \backslash \{\boldsymbol{\alpha} \in N_\infty(\boldsymbol{e}^*): f_{OBJ}^{SS}(\boldsymbol{\alpha}) \geq f_{OBJ}(\boldsymbol{e}^*)\}$ is a reduced neighborhood that excludes those neighbors $\boldsymbol{\alpha}$ with a steady state objective function $f_{OBJ}^{SS}(\boldsymbol{\alpha})$ that is equal or worse than the current objective function $f_{OBJ}(\boldsymbol{e}^*)$. To illustrate the advantages of using $N'_\infty(\boldsymbol{e}^*)$, consider a hypothetical case where $|N_\infty(\boldsymbol{e}^*)|$ is 3^{n_e} and $n_e = 9$. Assume that an average solution time for problem (A.4A) is one hour. If $N_\infty(\boldsymbol{e}^*)$ is used, the solution time of Steps 4 and 5 will be around two years. Hence, $N_\infty{}'(\boldsymbol{e}^*)$ reduces the combinatorial complexity of the problem since it will only explore those neighbors that have the potential to improve process economics. To further decrease

the computational costs, a relative gap can be introduced in the solution strategy. This gap is introduced in Step 3 of Figure A.3, i.e., $gap(\boldsymbol{e}^*) = (f_{OBJ}(\boldsymbol{e}^*) - LO(\boldsymbol{e}^*))/|f_{OBJ}(\boldsymbol{e}^*)|$; $LO(\boldsymbol{e}^*)$ is the tightest lower bound of the problem, and is defined with respect to the steady-state multi-period subproblems as $LO(\boldsymbol{e}^*) = \min_{\boldsymbol{e} \in N_\infty(\boldsymbol{e}^*)} \{f_{OBJ}^{SS}(\boldsymbol{e}) : \boldsymbol{e} \neq \boldsymbol{e}^*\}$. Once this gap is below a tolerance level $\varepsilon_{gap} \geq 0$, the search procedure stops. Note that the methodology presented in this section does not guarantee global optimality; hence, the process design obtained with the present method depends on the initial design provided by the first module shown in Figure A.1.

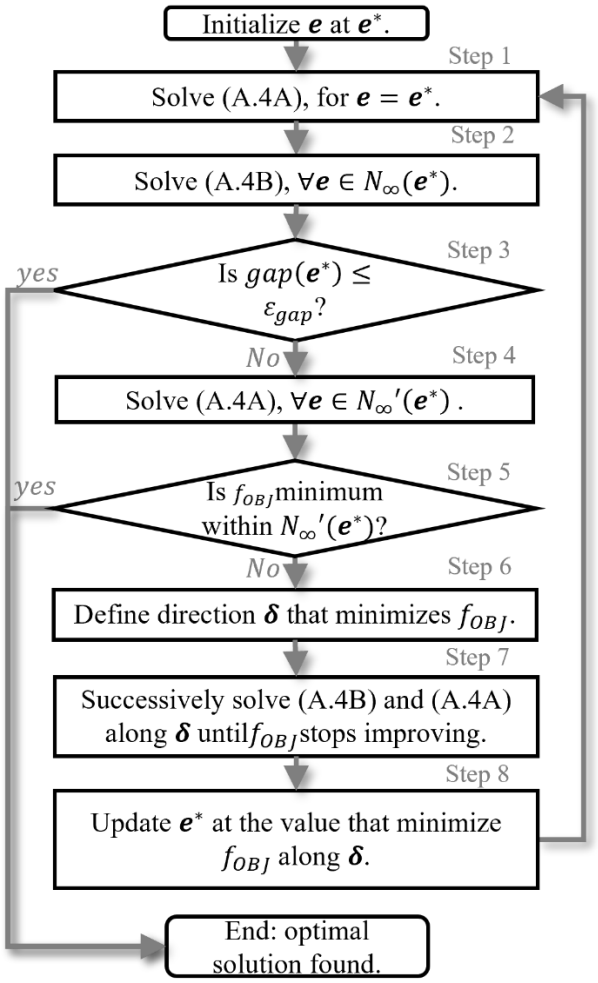


Figure A.3. The enhanced D-SDA. $f_{OBJ} = +\infty$ for infeasible problems.

A.3 Case study

The methodology proposed in the previous section was tested using a case study featuring the optimal design and control of a catalytic distillation (CD) column that produces multiple grades of ETBE from isobutene and ethanol. The column considers 3 catalytic stages and uses a split feed mode, with a feed of pure ethanol ($1.712\ mol/min$ at $342K$) and a second feed consisting of a n-butene/isobutene mixture ($5.774\ mol/min$ at $323K$). The discrete decisions formulated as external variables ($\boldsymbol{e}$) are: the number of stages, the location of both feeds, and the optimal distribution of reactive stages along the column. The continuous design variables ($\boldsymbol{\varphi}$) are the column diameter, the tray spacing and the weir height. The top operating pressure of the column is fixed at 9.5 bar and constant liquid mass accumulation in the condenser and the reboiler is assumed, which leaves two degrees of freedom to control the process. Conventional feedback controllers are considered in this work. The objective function follows the form shown in Eq. (A.1A) and (A.1B) and is as follows:

$$C_{INV} = C_0 + C_1 \tag{A.5A}$$

$$C_{OP}(t) = C_{Ethanol} + C_{Butenes} + C_{Reboiler} + C_{Condenser} \tag{A.5B}$$

$$C_{IN}(t) = C_{ETBE} \tag{A.5C}$$

where C_{INV} in Eq. (A.5A) includes a fixed investment cost of the condenser and the reboiler (C_0), and the cost of purchasing and installing the distillation vessel, the trays along the column and the catalytic section (C_1). The operating costs in Eq. (A.5B) include the cost of the ethanol ($C_{Ethanol}$) and butenes ($C_{Butenes}$) feed streams, and the operating costs of the reboiler and the condenser ($C_{Reboiler}$ and $C_{Condenser}$). The process' incomes in Eq. (A.5C) consider the revenue of selling the ETBE. The CD column is modeled using the MESH model coupled with geometrical constraints between the process design variables, e.g., a height to diameter ratio, empirical correlations to consider the pressure drop of the system, and hydrodynamic constraints, e.g., weeping and flooding. The reader is referred to chapter 6 for additional modelling details. The aim in this case study is to specify a design and control strategy that can meet the desired product and process specifications at the lowest EAC.

A.3.1 Scenario I: Optimal design under product changeovers

This scenario aims to design a CD column that satisfies a schedule with changes in ETBE composition (x_{ETBE}) demands during a full year of operation. The four periods ($K = \{1,2,3,4\}$) considered for this problem are: 1) transition from $x_{ETBE} = 63\%$ to $x_{ETBE} = 83\%$ and operation at $x_{ETBE} = 83\%$ for two months, 2) transition from $x_{ETBE} = 83\%$ to $x_{ETBE} = 95\%$ and operation at $x_{ETBE} = 95\%$ for four months, 3) transition from $x_{ETBE} = 95\%$ to $x_{ETBE} = 83\%$ and operation at $x_{ETBE} = 83\%$ for two months, and 4) transition from $x_{ETBE} = 83\%$ to $x_{ETBE} = 63\%$ and operation at $x_{ETBE} = 63\%$ for four months. A PID controller pairing the reboiler duty (Q_r) and x_{ETBE} is considered whereas the reflux ratio (*RR*) is fixed at an economically optimal point that satisfies flooding and weeping constraints, which is an approach previously considered in the literature[349]. The sampling time and transition times were fixed at 10 min and 60 min, respectively. The nonlinearity and bidirectionality in the process gain between Q_r and x_{ETBE} under different production grades was detected from preliminary simulations. Hence, the PID controller was designed such that their tuning parameters are allowed to change depending on the period of operation, i.e., a gain-scheduling controller is designed for this scenario. This results in 12 controller tuning parameters ($\boldsymbol{\tau}$) that must be economically optimized together with the process design variables specified for the CD column.

A.3.2 Scenario II: Optimal design with a feed disturbance

This scenario assumes that the system operates at $95\% mol/mol$ production of ETBE, with a butene feed containing $40\%\ mol/mol$ of isobutene. A measured disturbance in the feed is introduced after six months of operation, which drops the isobutene composition from $40\%\ mol/mol$ to $30\%\ mol/mol$. Hence, the first time-slot (period $k = 1$) consists of a steady-state operation of the system with a butenes feed of $40\%\ mol/mol$, while the second time-slot (period $k = 2$) considers the dynamic rejection period of the disturbance followed by a steady state operation at the new operating point ($30\%\ mol/mol$) for the second half of the year. In this case two PID controllers are implemented, i.e., x_{ETBE} is controlled with Q_r while the temperature difference of the reactive zone (ΔT) is controlled by manipulating *RR*. ΔT provides a good estimation of isobutene conversion, which is directly correlated with the quality of the ETBE product[378]. The sampling time and transition times were fixed to 10 min and 130 min, respectively. This work assumes that the step disturbance is a common perturbation to the system; hence, Module 2 aims to find a design that can accommodate the optimal steady-state operation under the nominal condition (40% mol/mol) and after the dynamic rejection period (i.e., 30%

mol/mol). This assumption is needed to guarantee a feasible initialization for Module 3, which is a requirement for the D-SDA framework presented in Figures A.2 and A.3.

A.4 Results and discussion

The problems were implemented and solved in GAMS 34, using CONOPT4 as the NLP solver in an Intel CPU with 96 GB of RAM and 2.1 GHz. The problem was discretized using the orthogonal collocation method with Lagrange interpolation polynomials. Both scenarios were solved using the modular strategy shown in Figure A.1, with a relative optimality gap of $\varepsilon_{gap} = 0$ for module 3. Also, both scenarios were initialized with the design proposed in reference[12] . The first scenario converged after 111.6 hours while the second scenario required 69.29 hours of computation.

A.4.1 Scenario I: Optimal design under product changeovers

The design obtained for this scenario is a CD column with 20 stages, feeds located at stages 6 and 17 (from top to bottom, including the condenser and the reboiler), and reactive stages optimally distributed between feeds at stages 6, 12 and 17. The column has a diameter of $0.17\ m$, tray spacing of $0.16\ m$, and a weir height of $0.008\ m$. The resulting EAC for this CD column is $22{,}610\ \$/year$, with an optimal reflux ratio of 4.96. Note that when this column is designed using an overdesign methodology presented in chapter 6, the resulting design does not comply with the height to diameter constraint ($Height/Diameter \leq 20$) imposed to this system. Even if this constraint is not enforced, the overdesign system would result in a CD column with an EAC that would be at least 0.4% worst than that obtained by the proposed framework, thus highlighting the need of a methodology like that presented here to ensure dynamic feasibility of RD columns at optimal costs. Note that other control strategies were tested, e.g., control the ethanol composition at the top by manipulating RR; however, they showed negligible economic deviations.

The optimal transition of the ETBE composition (x_{ETBE}), the controller tuning parameters, and the control actions in the reboiler duty (Q_r) are shown in Figure A.4 for periods 1 and 2 (periods 3 and 4 are not shown for brevity). As shown in this figure, the controller parameters differ by one order of magnitude for the product transitions considered. This is because the process is highly nonlinear and responds in different directions depending on the operating conditions. Hence, the need for a gain-scheduling controller for this multi-grade CD column. Note that a controller with a single set of tuning

parameters was attempted but it resulted in an infeasible solution. As shown in Figure A.4, the corresponding control actions are able to drive the system to their desired set-points.

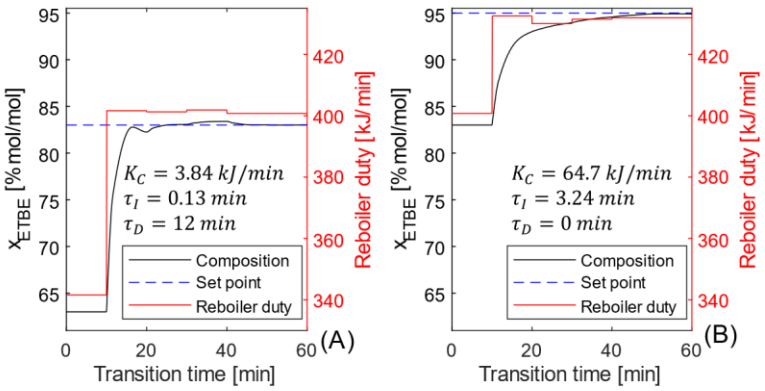


Figure A.4. Optimal transitions and PID tuning parameters: proportional gain (K_C), integral time (τ_I), and derivative time (τ_D). (A): period 1, (B): period 2.

A.4.2 Scenario II: Optimal design with a feed disturbance

To illustrate the benefits of the modular D-SDA approach presented in Figure A.1, the results for this scenario are compared with an optimal steady-state design obtained around a nominal operating condition with a feed that consists of $40\%\ mol/mol$ of isobutene. This design was obtained by minimizing the EAC, using modules 1 and 2 in Figure A.1 under nominal operating conditions (i.e., process dynamics were not considered). The optimal steady-state design and that obtained by the present scenario are presented in Figure A.5. When validating the designs, it was found that optimal steady-state design around a nominal operating condition (Figure A.5A) returned a CD column that would not meet their specification goals thus resulting in a dynamically infeasible design, i.e., the ETBE composition is 5% lower than the desired specification (95%) at steady-state (not shown for brevity). While the proposed optimal design and control approach returned a larger CD column (Figure A.5B) than that obtained by nominal design (Figure A.5A), the proposed simultaneous-based design can reject disturbances and maintain the ETBE product at the desired specification, with an EAC of 21,942 $/year.

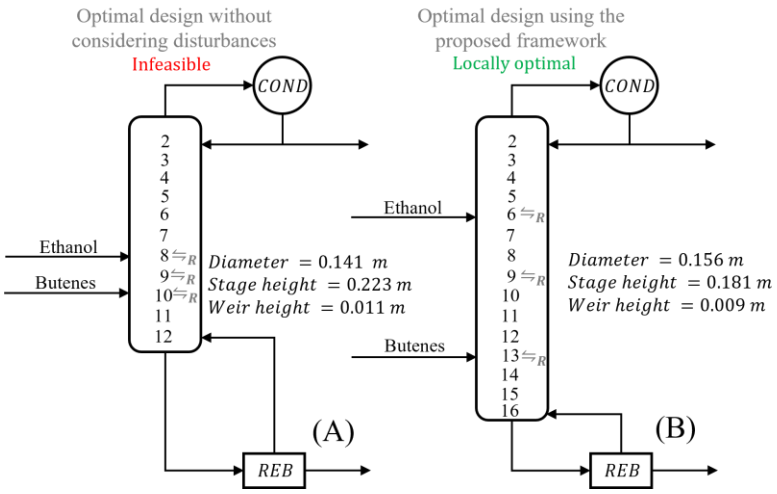


Figure A.5. (A) Optimal steady-state design (B) Optimal design using the modular D-SDA. $\rightleftharpoons_R$: reactive stages.

Figure A.6 validates the CD column design obtained by the present approach (Figure A.5B). As shown in this figure, the disturbance in the feed composition triggers a decrease in the quality of ETBE; thus, the control system reacts by making changes in both *RR* and Q_r such that it rejects this perturbation and eventually returns the system to the desired set-points. *RR* increases from 4.32 to 4.54 (not shown for brevity). The tuning parameters for the *RR*-ΔT control loop are $K_C = -0.0003K^{-1}$, $\tau_I = 1.49\ min$ and $\tau_D = 1.9 * 10^3\ min$.

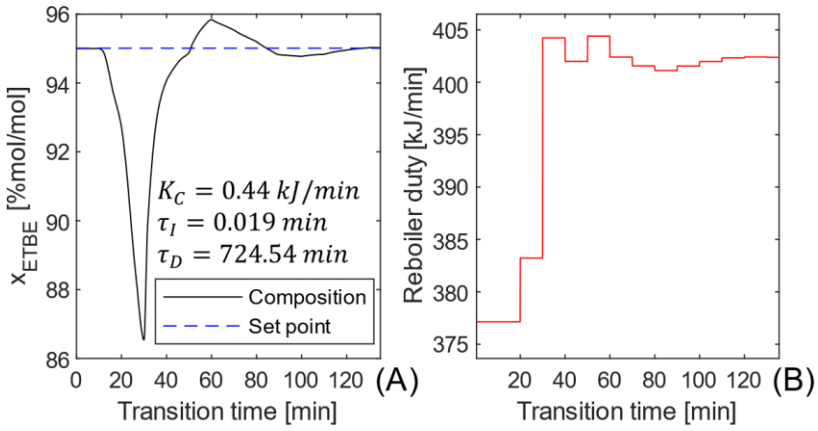


Figure A.6. Optimal transition and tuning parameters for the $Q_r - x_{ETBE}$ PID loop. (A): x_{ETBE} profile, (B): Q_r profile.

A.4 Summary

A deterministic method to simultaneously design and control RD systems that perform multiple transitions during its operation was presented in this study. The approach is based on a multi-period formulation that was solved using a D-SDA strategy. The main novelty of this work is the integration of the flexible operation of RD systems around multiple steady states in closed-loop with the optimal economic design of the process, considering both discrete and continuous design variables. A case study involving the production of ETBE returned optimal designs that are dynamically feasible compared to other optimization-based strategies that do not consider process dynamics or closed-loop operation. Future work includes the inclusion of unmeasured disturbances, parameter uncertainty and stability analysis in the proposed framework. Also, a comparison between this strategy and other design and control methods is left as future work. In addition, the D-SDA presented in this work need to be improved to guarantee optimality when initialized from an infeasible point. Also, the integration of the present approach with advanced control techniques such as model predictive control (MPC) will also be investigated in the future.

Appendix B

Optimal scheduling and open-loop control of network batch processes under variable processing times using Generalized Benders Decomposition

Sequential approaches for optimal scheduling and control have been shown to provide suboptimal (or even infeasible) operation strategies[61]. One variable that is either fixed or heuristically selected by sequential approaches is the processing time of a task, which is defined as the time needed by a unit operation within a chemical process to complete a task, e.g., the time required to finish a batch filtration process. If short processing times are selected, these will be able to accommodate more tasks within the scheduling horizon, but infeasibilities may appear if the dynamic operation of a batch process is disregarded by the sequential approach, e.g., more time may be needed to satisfy a reaction extent target. In contrast, selecting long processing times may offer more flexibility in the dynamic operation of batch processes, but excessively long tasks may not fit within a fixed scheduling horizon. Simultaneous scheduling and control approaches that consider variable processing times are thus needed to account for the interactions that emerge between scheduling and control decisions. These problems are often posed as complex mixed-integer dynamic optimization (MIDO) formulations; solving those problems is particularly challenging due to the presence of binary variables in the scheduling problem, and the interactions of processing times with discrete scheduling and continuous control decisions that emerge in the differential equations that describe the transient operation of batch units.

The simultaneous scheduling and open-loop control (SSOC) of network batch processes is considered in this work. The scheduling problem is augmented by including detailed dynamic models of batch operations whereas open-loop control actions are obtained through dynamic optimization. After discretization of the differential equations, the resulting problem takes the form of a Mixed-integer Nonlinear Programming (MINLP) problem. Previous studies have addressed this problem using a Generalized Benders Decomposition (GBD) strategy that separates the scheduling problem from the time-dependent (dynamic) decisions, thus leveraging the special structure that links the scheduling and control problems. Chu and You used GBD to address a SSOC formulated through a state task networks (STN)[61]; Nie et al. harnessed GBD to address SSOC problems linked to resource task network (RTN)[62],

whereas Ji and Gu introduced a hybrid deterministic-stochastic GBD strategy for SSOC in state equipment networks (SEN)[60]. Both discrete-time and continuous-time formulations have been used in those studies to represent scheduling decisions in SSOC. Although SSOC problems with continuous-time formulations are more prevalent in the literature, the study of SSOC for network batch processes using discrete-time formulations remains relatively scarce[62]. Solution strategies for the discrete-time SSOC of network batch processes under variable processing times have not been studied in the literature, despite being one of the main variables that influence scheduling and control actions. In contrast to continuous-time formulations, discrete-time scheduling formulations are known to offer industrially relevant modeling features such as backlog costs, time-varying prices, or time-dependent resource availability. Also, discrete-time scheduling formulations can achieve better solutions in less computational time than continuous-time formulations when solving large-scale scheduling instances[364].

Based on the above, this work brings novelty by introducing a GBD solution framework for the SSOC of discrete-time STNs with variable processing times, which is a pivotal stride towards efficient SSOC solutions. Previous implementations of GBD for SSOC[60–62] assumed subproblem feasibility for all scheduling variable combinations, which is not practical when considering machines operating under different modes thus resulting in variable processing times. To address this challenge, the proposed framework employs both optimality and feasibility cuts to handle infeasible solutions. To further enhance computational efficiency, the use of initial and auxiliary feasibility cuts is proposed. The key contributions of this work with respect to previous works are: i) the implementation and performance evaluation of feasibility cuts required to handle the infeasibilities induced by variable processing times within the GBD approach, and ii) the evaluation of the performance of GBD when adding initial and auxiliary cuts that facilitate the handling of infeasibilities and improve computational performance.

This study is organized as follows: Section B.1 presents the batch-network SSOC problem statement. Section B.2 describes the GBD proposed in this work. Section B.3 applies the proposed framework to a discrete-time SSOC problem with variable processing times adapted from the literature. Concluding remarks and future work are provided at the end.

B.1 Problem statement

A general SSOC formulation shown in (B.1) is considered in this work. This formulation assumes that dynamic models are available to predict the transient behavior of the tasks involved, e.g., differential

equations to model reaction or separation tasks. Problem (B.1) considers the minimization of an objective function that involves the addition of a scheduling-related term (f^{sch}) and a term related to process dynamics and control (f^{dyn}). Variables are thus classified into scheduling variables ($\boldsymbol{v}^{sch}$) and dynamic variables ($\boldsymbol{v}^{dyn}$), where the former account for decisions such as sequencing, timing, processing times, and batch size per task execution. $\boldsymbol{v}^{sch}$ is assumed to contain both integer and continuous variables, while $\boldsymbol{v}^{dyn}$ contains only continuous variables. $\boldsymbol{v}^{sch\prime}$ is used to represent the subset of variables in $\boldsymbol{v}^{sch}$ that appear in f^{dyn}, while variables in $\boldsymbol{v}^{dyn\prime}$ are variables in $\boldsymbol{v}^{dyn}$ that appear in $\boldsymbol{g}^{link}$, which includes those constraints that link scheduling and dynamic decisions, e.g., the batch size of a task present in both $\boldsymbol{v}^{sch}$ and $\boldsymbol{v}^{dyn}$. Constraints $\boldsymbol{g}^{sch}$ only consider scheduling constraints, e.g., unit utilization and unit capacity constraints whereas $\boldsymbol{g}^{dyn}$ only considers dynamic constraints, e.g., discretized differential equations and product quality requirements. Problem (B.1) is used in this work to enable the discrete-time SSOC modeling of batch plants with variable processing times in a general network production environment.

$$\min f^{sch}\left(\boldsymbol{v}^{sch}\right) + f^{dyn}\left(\boldsymbol{v}^{sch\prime}, \boldsymbol{v}^{dyn}\right) \tag{B.1}$$
$$s.t.$$
$$\boldsymbol{g}^{sch}\left(\boldsymbol{v}^{sch}\right) \leq \boldsymbol{0},\ \boldsymbol{g}^{dyn}\left(\boldsymbol{v}^{dyn}\right) \leq \boldsymbol{0}, \boldsymbol{g}^{link}\left(\boldsymbol{v}^{sch}, \boldsymbol{v}^{dyn\prime}\right) \leq \boldsymbol{0}$$

The discrete-time STN formulation is considered to define constraints $\boldsymbol{g}^{sch}\left(\boldsymbol{v}^{sch}\right)$[360]. The STN consists of *tasks* denoted by index $i \in I$ (e.g., reaction, separation); *units* denoted by index $j \in J$ (e.g., batch reactor, distillation column), and *states* denoted by index $m \in M$ (e.g., intermediate storage and final products). The production environment is defined in terms of binary parameters $I^{-}_{i,m}/I^{+}_{i,m}, \forall i \in I, \forall m \in M$, which take the value of 1 when task i consumes/produces material from/to state m; and binary parameters $I^{\times}_{i,j}, \forall i \in I, \forall j \in J$, which indicate if task i can be performed at unit j. Set $IJ^{\times} = \{(i,j): I^{\times}_{i,j} = 1\}$ is defined to represent the task to units mapping through tuples (i,j). A discrete-time grid is defined by the set $T = \{0,1,2,\dots,n_T\}$ with index t and $n_T = \eta/\hat{\delta}$, where η is the scheduling horizon and δ is the length of the time slots. The variables included in $\boldsymbol{v}^{sch}$ are as follows: $x_{i,j,t} \in \{0,1\}$ indicate if unit j process task i starting at time t; $b_{i,j,t} \in \mathbb{R}_{\geq 0}$ defines the batch size of task i when processed at unit j starting at time t; $s_{m,t} \in \mathbb{R}_{\geq 0}$ track the inventory of state m at time t. The scheduling model parameters include fixed costs ($c^{fix}_{i,j}$) and variable costs ($c^{var}_{i,j}$) of executing task i in unit j; cost

of raw materials (c_m^{mat}) and selling price of products (c_m^{out}). The number of time slots in the discrete time grid T required to run task i in unit j is denoted by $\tau_{i,j} = \lceil \tau_{i,j}^p / \hat{\delta} \rceil$, where $\tau_{i,j}^p$ is the processing time (with units of time) and superscript p is used to distinguish $\tau_{i,j}$ from $\tau_{i,j}^p$. The maximum (b_j^{max}) and minimum (b_j^{min}) capacity of unit j; the initial storage (s_m^{init}) and maximum storage capacity (s_m^{max}) of state m, and the fraction of material in state m consumed ($\rho_{i,m}^-$) or produced ($\rho_{i,m}^+$) by task i. As shown in (B.2A), f^{sch} represents the addition of processing and material costs minus the revenue, while scheduling constraints $\boldsymbol{g}^{sch}(\boldsymbol{v}^{sch})$ include unit utilization (B.2B), unit capacity (B.2C), inventory material balances (B.2D)-(B.2E) and capacity constraints (B.2F).

$$f^{sch} = \sum_{(i,j)\in IJ^{\times}} \sum_{t\in T} \left(c_{i,j}^{fix} x_{i,j,t} + c_{i,j}^{var} b_{i,j,t} \right) + \sum_{m\in M} \left(c_m^{mat} \left(s_m^{init} - s_{m,n_T} \right) - c_m^{out} s_{m,n_T} \right) \tag{B.2A}$$

$$\sum_{\substack{i\in I \\ I_{i,j}^{\times}=1}} \sum_{\substack{t'\in T \\ t-\tau_{i,j}+1\le t'\le t}} x_{i,j,t'} \le 1, \ \forall j \in J, \forall t \in T \tag{B.2B}$$

$$b_j^{min} x_{i,j,t} \le b_{i,j,t} \le b_j^{max} x_{i,j,t}, \ \forall (i,j) \in IJ^{\times}, \forall t \in T \tag{B.2C}$$

$$s_{m,t} = s_{m,t-1} + \sum_{\substack{i\in I \\ I_{i,m}^{+}=1}} \rho_{i,m}^{+} \sum_{\substack{j\in J \\ I_{i,j}^{\times}=1}} b_{i,j,t-\tau_{i,j}} - \sum_{\substack{i\in I \\ I_{i,m}^{-}=1}} \rho_{i,m}^{-} \sum_{\substack{j\in J \\ I_{i,j}^{\times}=1}} b_{i,j,t}, \tag{B.2D}$$

$$\forall m \in M, \forall t \in T \backslash \{0\}$$

$$s_{m,0} = s_m^{init} - \sum_{\substack{i\in I \\ I_{i,m}^{-}=1}} \rho_{i,m}^{-} \sum_{\substack{j\in J \\ I_{i,j}^{\times}=1}} b_{i,j,0}, \forall m \in M \tag{B.2E}$$

$$s_{m,t} \le s_m^{max}, \forall m \in M, \forall t \in T \tag{B.2F}$$

Tasks with variable processing times are approximated in this work using the *task mode* approach, i.e., tasks i are split into their allowed operation modes, each with a predetermined processing time, e.g., a reaction task with operation modes of 1h, 1.5h, or 2h[360]. Thus, tasks with variable processing times can be expressed as tuples $i = (\hat{\imath}, q)$, where $\hat{\imath} \in I^{dyn}$ denotes the task type and $q \in Q_{\hat{\imath}}$ its operation mode. A differential-algebraic system of equations (DAE) with end point and path constraints

is considered for pairs $(\hat{\imath}, j) \in IJ^{\times,dyn}$ for which dynamic models are available. Constraints $\boldsymbol{g}^{dyn}(\boldsymbol{v}^{dyn})$ describing a dynamic task-unit pair are shown in (B.3A)-(B.3C), where $\boldsymbol{\chi}_{\hat{\imath},j}$, $\boldsymbol{\mu}_{\hat{\imath},j}$ and $\boldsymbol{\zeta}_{\hat{\imath},j}$ are time-dependent state, control, and algebraic variables, respectively; $\dot{\boldsymbol{\chi}}_{\hat{\imath},j}$ is the derivative of $\boldsymbol{\chi}_{\hat{\imath},j}$ with respect to dimensionless time $\psi \in [0,1]$; $\boldsymbol{r}_{\hat{\imath},j}$ contains algebraic equations, path and end-point constraints; $\boldsymbol{\chi}^0_{\hat{\imath},j}$ is the initial condition of the system; $\varphi^b_{\hat{\imath},j}$ is the batch size (i.e., amount of material being processed, where superscript b stands for batch), and $\varphi^p_{\hat{\imath},j}$ is the processing time. These variables are defined for every pair $(\hat{\imath}, j) \in IJ^{\times,dyn}$. Note that variables $\varphi^b_{\hat{\imath},j}$ and $\varphi^p_{\hat{\imath},j}$ in $\boldsymbol{v}^{dyn}$ are the counterparts of $b_{i,j,t}$ and $\tau^p_{i,j}$ in the dynamic constraints of the problem, and that dimensionless time ψ covers the range $[0, \varphi^p_{\hat{\imath},j}]$. The function f^{dyn} is shown in (B.3D); this includes $x_{(\hat{\imath},q),j,t}$ and are used to turn on/off cost terms related to the dynamic operation of task type $\hat{\imath}$ in unit j at time t, i.e., variables $x_{(\hat{\imath},q),j,t}$ constitute $\boldsymbol{v}^{sch\prime}$. According to (B.3D), the dynamic cost of a task is given by the integral over dimensionless time $\psi \in [0,1]$ of the dot product of control actions $\boldsymbol{\mu}_{\hat{\imath},j}$ with cost coefficients $\boldsymbol{c}^{dyn}_{\hat{\imath},j}$, e.g., integral of power requirements multiplied by their costs. Note that this is a multiscale model with a scheduling time scale ($t \in T$) and a control time scale ($\psi \in [0,1]$).

$$\dot{\boldsymbol{\chi}}_{\hat{\imath},j} = \varphi^p_{\hat{\imath},j} \boldsymbol{h}_{\hat{\imath},j}\left(\boldsymbol{\chi}_{\hat{\imath},j}, \boldsymbol{\zeta}_{\hat{\imath},j}, \boldsymbol{\mu}_{\hat{\imath},j}, \varphi^b_{\hat{\imath},j}\right), \forall (\hat{\imath}, j) \in IJ^{\times,dyn} \tag{B.3A}$$

$$\boldsymbol{r}_{\hat{\imath},j}\left(\boldsymbol{\chi}_{\hat{\imath},j}, \boldsymbol{\zeta}_{\hat{\imath},j}, \boldsymbol{\mu}_{\hat{\imath},j}, \varphi^b_{\hat{\imath},j}\right) \leq \boldsymbol{0}, \forall (\hat{\imath}, j) \in IJ^{\times,dyn} \tag{B.3B}$$

$$\boldsymbol{\chi}_{\hat{\imath},j}(\psi = 0) = \boldsymbol{\chi}^0_{\hat{\imath},j}, \forall (\hat{\imath}, j) \in IJ^{\times,dyn} \tag{B.3C}$$

$$f^{dyn} = \sum_{(\hat{\imath},j) \in IJ^{\times,dyn}} \sum_{q \in Q_{\hat{\imath}}} \sum_{t \in T} \left(x_{(\hat{\imath},q),j,t} \varphi^p_{\hat{\imath},j} \int_0^1 \langle \boldsymbol{c}^{dyn}_{\hat{\imath},j}, \boldsymbol{\mu}_{\hat{\imath},j} \rangle \, d\psi \right) \tag{B.3D}$$

Constraints included within $\boldsymbol{g}^{link}$ are shown in (B.4A) and (B.4B), where M is a big-M coefficient. These constraints link scheduling and control decisions through $\varphi^p_{\hat{\imath},j}$ and $\varphi^b_{\hat{\imath},j}$, which belong to $\boldsymbol{v}^{dyn\prime}$. If a task is scheduled (i.e., $x_{(\hat{\imath},q),j,t} = 1$), then (B.4A) guarantees that $b_{(\hat{\imath},q),j,t} = \varphi^b_{\hat{\imath},j}$. Similarly, (B.4B) guarantees that $\tau^p_{(\hat{\imath},q),j} = \varphi^p_{\hat{\imath},j}$ when $x_{(\hat{\imath},q),j,t} = 1$.

$$-M\left(1 - x_{(\hat{\imath},q),j,t}\right) \leq b_{(\hat{\imath},q),j,t} - \varphi^b_{\hat{\imath},j} \leq M\left(1 - x_{(\hat{\imath},q),j,t}\right) \tag{B.4A}$$

$$-M\left(1-\sum_{q\in Q_i} x_{(i,q),j,t}\right) \le \sum_{q\in Q_i} x_{(i,q),j,t}\tau^{p}_{(i,q),j} - \varphi^{p}_{i,j} \le M\left(1-\sum_{q\in Q_i} x_{(i,q),j,t}\right) \quad \text{(B.4B)}$$

B.2 Solution framework

B.2.1 The GBD algorithm for SSOC

The first requirement to solve (B.1) with GBD is to decide which subset of variables will be used as complicating variables, i.e., variables that when fixed are expected to simplify the solution of the so-called subproblem. The optimization formulation used to optimize complicating variables is called the master problem. Based on the structure of (B.1), scheduling $\boldsymbol{v}^{sch}$ and linking variables $\boldsymbol{v}^{dyn\prime}$ are selected as the complicating variables which when they are fixed result in an NLP subproblem. Based on the selected complicating variables, the next step is to formulate the master problem and the subproblem. In this case, a disaggregation strategy for the complicating variables is considered. This strategy introduces dummy variables, i.e., $\overline{\boldsymbol{v}}^{sch\prime} = \boldsymbol{v}^{sch\prime}$ and $\overline{\boldsymbol{v}}^{dyn\prime} = \boldsymbol{v}^{dyn\prime}$ to guarantee that the constraints in the master problem remain linear. Given that the resulting subproblems are NLPs, Lagrange duality can be used to define the master problem as shown in (B.5), and the optimality and feasibility subproblems shown in (B.6A) and (B.6B), respectively. As shown in (B.5), z is a real variable approximating f^{dyn}; ξ denotes a cut inferred from Lagrange duality, whereas $\boldsymbol{\varepsilon}$ in (B.6B) is a vector of slack positive real variables required to define the feasibility subproblem. Superscript k is used to denote the variables that remain fixed at iteration k, while subscripts opt and fea stand for optimality and feasibility, respectively.

$$\min f^{sch}\left(\overline{\boldsymbol{v}}^{sch}\right) + z \quad \text{(B.5)}$$

$$s.t.$$

$$\xi^{k}_{opt}\left(\overline{\boldsymbol{v}}^{sch\prime}, \overline{\boldsymbol{v}}^{dyn\prime}\right) - z \le 0 \;\forall k \in K_{opt}$$

$$\xi^{k}_{fea}\left(\overline{\boldsymbol{v}}^{dyn\prime}\right) \le 0 \;\forall k \in K_{fea}$$

$$\boldsymbol{g}^{sch}\left(\overline{\boldsymbol{v}}^{sch}\right) \le \boldsymbol{0}$$

$$\boldsymbol{g}^{link}\left(\overline{\boldsymbol{v}}^{sch}, \overline{\boldsymbol{v}}^{dyn\prime}\right) \le \boldsymbol{0}$$

Master problem (B.5) optimizes complicating variables $\overline{\boldsymbol{v}}^{sch}$ and $\overline{\boldsymbol{v}}^{dyn\prime}$ to minimize an objective function, which is the addition of the original scheduling objective ($f^{sch}\left(\overline{\boldsymbol{v}}^{sch}\right)$) and an approximation of the dynamic objective (z) that is bounded below by optimality cuts ξ_{opt}^{k}. Every feasible subproblem solved in previous iterations $k \in K_{opt}$ is used to generate a new optimality cut ξ_{opt}^{k}; thus, the approximation of the dynamic cost in the master problem through z improves as GBD iterates. Another type of cuts that are often neglected when solving SSOC problems through GBD are feasibility cuts ξ_{fea}^{k}. These cuts account for the fact that not every combination of variables $\overline{\boldsymbol{v}}^{dyn\prime}$ results in feasible subproblems. Thus, cuts ξ_{fea}^{k} are generated from every infeasible subproblem solved at iteration $k \in K_{fea}$ to move away from infeasible combinations of $\overline{\boldsymbol{v}}^{dyn\prime}$. Optimality or feasibility cuts can be obtained at each iteration k from subproblems (B.6A) or (B.6B), respectively. Note that (B.6B) is solved only in the case that (B.6A) returns an infeasible solution.

$$\min f^{sch}\left(\overline{\boldsymbol{v}}^{sch,k}\right) + f^{dyn}\left(\boldsymbol{v}^{sch\prime}, \boldsymbol{v}^{dyn}\right) \tag{B.6A}$$
$$s.t.$$
$$\boldsymbol{v}^{dyn\prime} - \overline{\boldsymbol{v}}^{dyn\prime k} = \mathbf{0}$$
$$\boldsymbol{v}^{sch\prime} - \overline{\boldsymbol{v}}^{sch\prime k} = \mathbf{0}$$
$$\boldsymbol{g}^{dyn}\left(\boldsymbol{v}^{dyn}\right) \leq \mathbf{0}$$

$$\min \|\boldsymbol{\varepsilon}\|_1 \tag{B.6B}$$
$$s.t.$$
$$\boldsymbol{v}^{dyn\prime} - \overline{\boldsymbol{v}}^{dyn\prime k} = \mathbf{0}$$
$$\boldsymbol{g}^{dyn}\left(\boldsymbol{v}^{dyn}\right) - \boldsymbol{\varepsilon} \leq \mathbf{0}$$

Optimality subproblem (B.6A) optimizes f^{dyn} with variables $\boldsymbol{v}^{sch\prime}$ and $\boldsymbol{v}^{dyn\prime}$ fixed at $\overline{\boldsymbol{v}}^{sch\prime k}$ and $\overline{\boldsymbol{v}}^{dyn\prime k}$, respectively. Note that the objective function in (B.6A) also includes the constant term $f^{sch}\left(\overline{\boldsymbol{v}}^{sch,k}\right)$, which corresponds to the scheduling objective function obtained by the master problem (B.5). Subproblem (B.6B) is used to identify the source of infeasibility by minimizing the constraint violation with respect to the l_1-norm only when subproblem (B.6A) is infeasible. Optimality and feasibility cuts ξ_{opt}^{k} and ξ_{fea}^{k} derived from the solution of (B.6A) and (B.6B) can be obtained from (B.7A) and (B.7B), respectively. $\boldsymbol{\lambda}_{opt,d}^{k}$ and $\boldsymbol{\lambda}_{opt,s}^{k}$ are the optimal Lagrange multipliers of constraints

$\boldsymbol{v}^{dyn\prime} - \overline{\boldsymbol{v}}^{dyn\prime k} = \mathbf{0}$ and $\boldsymbol{v}^{sch\prime} - \overline{\boldsymbol{v}}^{sch\prime k} = \mathbf{0}$ in (6A), respectively. Similarly, $\boldsymbol{\lambda}_{fea,d}^{k}$ are the optimal Lagrange multipliers of $\boldsymbol{v}^{dyn\prime} - \overline{\boldsymbol{v}}^{dyn\prime k} = \mathbf{0}$ in (6B). These equations assume that ξ_{opt}^{k} and ξ_{fea}^{k} can be obtained from local information given by the evaluation of the Lagrange function at the optimal solution found from subproblems (6A) and (6B), respectively. Note that relying on feasibility cuts (7B) may lead to slow convergence in SSOC. To alleviate this issue, an improved strategy to handle infeasibility using initial and auxiliary cuts that take advantage of special characteristics of the problem is proposed next.

$$\xi_{opt}^{k} = f^{dyn}\left(\boldsymbol{v}^{sch\prime k}, \boldsymbol{v}^{dyn\prime k}\right) + \langle \boldsymbol{\lambda}_{opt,d}^{k}, \overline{\boldsymbol{v}}^{dyn\prime} - \boldsymbol{v}^{dyn\prime k} \rangle + \langle \boldsymbol{\lambda}_{opt,s}^{k}, \overline{\boldsymbol{v}}^{sch\prime} - \boldsymbol{v}^{sch\prime k} \rangle \tag{B.7A}$$

$$\xi_{fea}^{k} = \left\| \boldsymbol{\varepsilon}^{k} \right\|_{1} + \langle \boldsymbol{\lambda}_{fea,d}^{k}, \overline{\boldsymbol{v}}^{dyn\prime} - \boldsymbol{v}^{dyn\prime k} \rangle \tag{B.7B}$$

B.2.2 Initial and auxiliary cuts

Problems involving a large number of binary variables such as those found in SSOC are known to deteriorate the performance of GBD thus leading to high computational costs. In addition, the inclusion of variable processing times introduces infeasibilities that, when coupled with the inherent combinatorial complexity of the problem, may result in multiple infeasible subproblems. Thus, the performance of the GBD strategy for discrete-time SSOC problems with variable processing times heavily relies on the effectiveness of the strategies implemented to handle infeasibility. It has been shown that solely relying on traditional feasibility cutting strategies such as (B.7B) may result in slow performance, e.g., these may only avoid revisiting an infeasible solution in future iterations without cutting off other infeasible regions[379]. To address this issue, the performance of two types of cuts obtained based on the special structure of the (B.1) is considered in this work. The first type of cuts are initial cuts added to the master problem prior to the execution of GBD. The second type of cuts are auxiliary cuts generated using Lagrange dual information. The latter are similar to (B.7B) except that they move away from infeasible solutions based on the special characteristics of the SSOC problem.

The complicating variables $\boldsymbol{v}^{dyn\prime}$ fixed in feasibility subproblems (B.6B) are $\varphi_{i,j}^{p}$ and $\varphi_{i,j}^{b}$. If a processing time dictated by the master problem is too short, then the DAE with path and end-point constraint of a given pair $(i, j) \in IJ^{\times,dyn}$ may become infeasible. For example, if end-point concentration constraints and bounds over manipulated cooling/heating variables are imposed over the

operation of a batch reactor, the master problem will not account for this information. Thus, it may generate a processing time that is not long enough to satisfy the concentration requirement without violating bounds over manipulated variables. To address this issue, the initial cuts considered here are shown in (B.8A) and (B.8B). Prior executing GBD, all processing times $\varphi_{\hat{\imath},j}^{p}$ are minimized as shown in (B.8A), subject to the differential-algebraic equations used to model pairs $(\hat{\imath},j) \in IJ^{\times,dyn}$ in (B.3A)-(B.3C). Then, cuts as shown in (B.8B) are added to the master problem.

$$\varphi_{\hat{\imath},j}^{p,init} = \min\left\{\varphi_{\hat{\imath},j}^{p} | (\text{B. 3A}) - (\text{B. 3C})\right\} \tag{B.8A}$$

$$\bar{\varphi}_{\hat{\imath},j}^{p} \geq \varphi_{\hat{\imath},j}^{p,init}, \forall(\hat{\imath},j) \in IJ^{\times,dyn} \tag{B.8B}$$

The auxiliary feasibility cuts proposed here are obtained through (B.9A) and (B.9B). These cuts consider that the approximation of the minimum processing time can be updated as iterations progress based on the solution of new subproblems. At a given iteration k, processing times $\varphi_{\hat{\imath},j}^{p}$ are minimized through (B.9A), while keeping $\varphi_{\hat{\imath},j}^{b}$ fixed at the current master problem value $\bar{\varphi}_{\hat{\imath},j}^{b,k}$. This allows to obtain Lagrange multiplier $\lambda_{\hat{\imath},j}^{b,k}$ from constraint $\varphi_{\hat{\imath},j}^{b} - \bar{\varphi}_{\hat{\imath},j}^{b,k} = 0$, which provides sensitivity information of how the minimum processing time changes when varying batch size $\varphi_{\hat{\imath},j}^{b}$. Thus, a linear cut in (B.9B) is constructed and added to the master problem to restrict processing times. Instead of limiting processing times using a constant value, the effect of batch sizes on the minimum processing time is given by (B.9B).

$$\varphi_{\hat{\imath},j}^{p,k} = \min\left\{\varphi_{\hat{\imath},j}^{p} | (\text{B. 3A}) - (\text{B. 3C}), \varphi_{\hat{\imath},j}^{b} - \bar{\varphi}_{\hat{\imath},j}^{b,k} = 0\right\} \tag{B.9A}$$

$$\bar{\varphi}_{\hat{\imath},j}^{p} \geq \varphi_{\hat{\imath},j}^{p,k} + \lambda_{\hat{\imath},j}^{b,k}\left(\bar{\varphi}_{\hat{\imath},j}^{b} - \varphi_{\hat{\imath},j}^{b,k}\right), \forall(\hat{\imath},j) \in IJ^{\times,dyn} \tag{B.9B}$$

Note that auxiliary cuts can also be obtained by analyzing the subproblem from an alternative perspective. Instead of attributing infeasibility to poorly selected processing times for fixed batch sizes (i.e., as in (B.9A) and (B.9B)), infeasibility may be caused by oversized batches for fixed processing times. Thus, instead of finding the minimum processing time for a fixed batch size, it is also possible to find the maximum batch size for a fixed processing time as shown in (B.10A). The solution to this

problem allows to obtain auxiliary cuts in (B.10B), which provide upper bounds over batch sizes, which account for the local effect of varying processing times through Lagrange multipliers $\lambda_{\hat{\imath},j}^{p,k}$.

$$-\varphi_{\hat{\imath},j}^{b,k} = \min\left\{-\varphi_{\hat{\imath},j}^{b} | (\text{B.3A}) - (\text{B.3C}), \varphi_{\hat{\imath},j}^{p} - \bar{\varphi}_{\hat{\imath},j}^{p,k} = 0\right\} \tag{B.10A}$$

$$\bar{\varphi}_{\hat{\imath},j}^{b} \leq \varphi_{\hat{\imath},j}^{b,k} - \lambda_{\hat{\imath},j}^{p,k}\left(\bar{\varphi}_{\hat{\imath},j}^{p} - \varphi_{\hat{\imath},j}^{p,k}\right), \forall (\hat{\imath}, j) \in IJ^{\times,dyn} \tag{B.10B}$$

The initial and auxiliary cut generation strategies discussed above rely on two key conditions. First, it is assumed that optimization problems (B.8A), (B.9A), and (B.10A) are feasible and bounded. Note that the proposed initial and auxiliary cut generation strategies can be combined with standard feasibility cuts (B.7B) as a back-up in case convergence issues are detected with (B.8A), (B.9A) or (B.10A) at any iteration. Second, the proposed initial and auxiliary cut generation strategies rely on the separability of dynamic constraints (B.3A)-(B.3C). The proposed cuts cannot be generated independently for each pair $(\hat{\imath}, j) \in IJ^{\times,dyn}$ if separability does not hold, e.g., the initial conditions of one dynamic system are given by the final conditions of another dynamic system involved in the formulation.

The steps performed by the proposed GBD algorithm are summarized in Figure B.1. Lower and upper bounds (i.e., LB and UB) are updated with the best objective values found until the current iteration k for (B.5) and (B.6A), respectively, e.g., if (B.6A) is infeasible at iteration k, UB is updated with the best feasible solution found in previous iterations. The algorithm stops when the MINLP gap between UB and LB is below tolerance ϵ. The GBD strategy in Figure B.1 includes initial cuts obtained through (B.8A) and (B.8B) in step 1 of Figure B.1 to avoid infeasible combinations of processing times in the master problem. Note that one of the two strategies to generate auxiliary feasibility cuts must be selected a priori as shown in Figure 1. These auxiliary cuts are used instead of the traditional feasibility cuts shown in (B.7B). The first auxiliary feasibility strategy consists of solving (B.9A) and adding auxiliary cuts (B.9B), while the second strategy solves (B.10A) and adds auxiliary feasibility cuts in (B.10B). The combination of these cut generation strategies is also possible but not discussed for brevity, i.e., using auxiliary cuts in (B.9A)-(B.9B) and (B.10A)-(B.10B) simultaneously. The performance of the proposed GBD approach has been tested using a case study available in the literature.

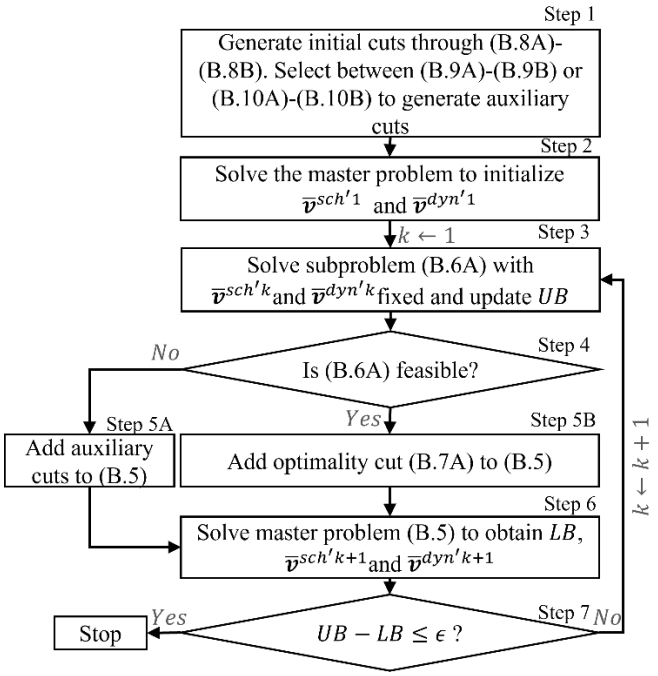


Figure B.1. Proposed GBD strategy

B.3 Results and discussion

The profit maximization of the STN illustrated in Figure B.2 is considered in this section. Colored blocks represent the tasks involved, which are interconnected through storage tanks (i.e., states) represented by grey circles. DAE systems in the form of (B.3A)-(B.3C) are considered for reaction tasks $R1$, $R2$ and $R3$. Seven task modes are used for each reaction task, i.e., 1, 1.5, 2, 2.5, 3, 3.5, 4 hours. These reactions can be performed in a large reactor Rl or a small reactor Rs. These reactions are controlled using bounded heating and cooling utilities that can be adjusted online and used to guarantee that temperature remains within pre-specified limits whereas composition quality requirements are achieved. Thus, the optimal selection of processing times is impacted not only by reaction kinetic parameters, but also by the size of the reactor selected to perform the reactions, the open-loop control actions over cooling and heating utilities, and the costs associated to these control actions. Other tasks in the processes are a mixing task (Mix) and a separation task (Sep), each with a single unit. The mixing task is needed to mix the raw materials required as an input to reaction $R2$, whereas the separation task is needed to purify the output from reaction $R3$. Products $P1$ and $P2$ coming from $R2$ and Sep can be packed in a single unit ($Pack$). The profit objective function includes fixed costs $c_{i,j}^{fix}$ for every task;

variable costs $c_{i,j}^{var}$ for Mix, $Pack1$ and $Pack2$; the costs of heating and cooling utilities for $R1$, $R2$ and $R3$ ($\boldsymbol{c}_{i,j}^{dyn}$); material costs c_m^{mat} of inputs $M1$, $M2$ and $M3$, and the revenue of selling products $P1$ and $P2$ (c_m^{out}). The problem is solved using a scheduling horizon η of 14 hours and a step size $\delta = 0.5$. The parameters, kinetic models, mass and energy balances for this case study are available elsewhere[61]. The computational experiments performed through this section were implemented in Pyomo in a PC Intel® Core™ i7-9700K CPU @ 3.60 GHz with 64 GB of RAM. CPLEX was used to solve the Mixed-Integer Linear Programming (MILP) master problem, while CONOPT4 was used as the NLP subproblem solver. A maximum CPU time of 1000 seconds was allocated for all instances, and a tolerance of $\epsilon = 10^{-5}$ was selected.

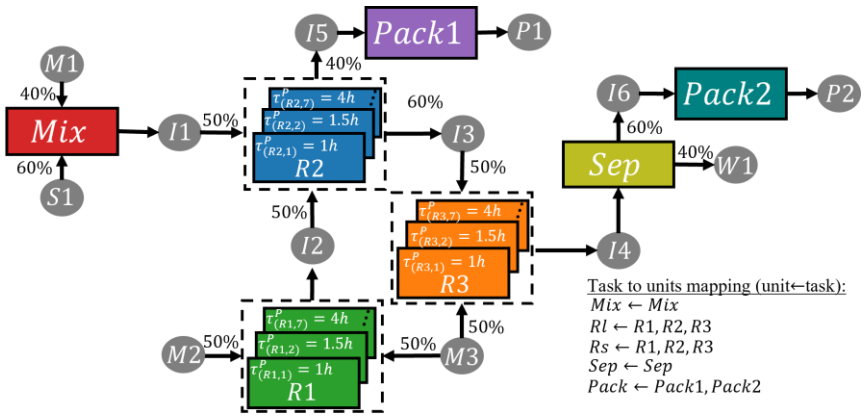


Figure B.2. STN of the problem

B.3.1 Comparison of GBD approaches

This section evaluates the performance of the GBD strategies described above. Three GBD strategies are compared in this section: i) GBD with traditional feasibility cuts and without initial or auxiliary cuts (GBD1); ii) GBD with initial cuts and auxiliary feasibility cuts in the form of (B.9A)-(B.9B) (GBD2), and iii) GBD with initial cuts and auxiliary feasibility cuts in the form of (B.10A)-(B.10B) (GBD3). GBD1 is considered as a benchmark since no changes were made to the original GBD algorithm, other than the addition of feasibility cuts. Figure B.3 depicts the evolution of the best solution known generated by the subproblem (i.e., the upper bound UB) and the lower bound (LB) generated by the master problem for GBD1, GBD2 and GBD3. The lower x-axis shows the cumulative solution time, i.e., the time consumed from the beginning of the execution until the current iteration. Each pair of markers LB and UB obtained for a fixed solution time in Figure B.3 corresponds to a new iteration.

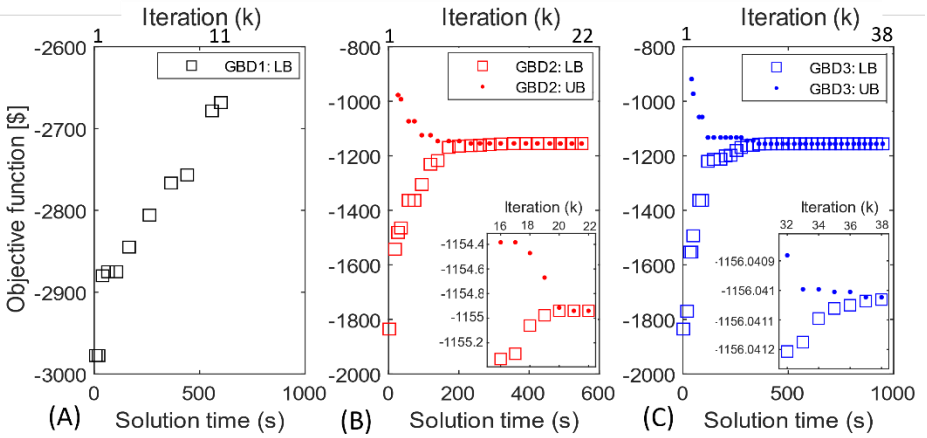


Figure B.3. Proposed GBD approaches. A) GBD1, B) GBD2, C) GBD3

As shown in Figure B.3A, GBD1 was unable to identify a feasible solution to the problem after 11 iterations, i.e., only infeasible subproblems were detected, which generated traditional feasibility cuts. On the other hand, the strategies that implement initial and feasibility cuts (GBD2 and GBD3) were able to close the MINLP gap below 10^{-5}, finding solutions with objective functions of -1154.94$ and -1156.04$ for GBD2 and GBD3, respectively (Figures B.3B and B.3C). GBD2 required 549.45 seconds and 22 iterations, whereas GBD3 needed 948.07 seconds and 38 iterations. Subplots in Figures B.3B and B.3C are shown to illustrate how the gap closes towards the last iterations. GBD2 and GBD3 were able to reduce the size of the search space of variable processing times in the master problem, by generating lower bounds that range from 1.08 hours for $R3$ in Rs to 2.55 hours for $R2$ in Rs. In the case of GBD2, only initial cuts (B.8B) and auxiliary feasibility cuts (B.9B) generated in the first iteration were needed to guarantee feasibility of the subproblems in subsequent iterations, i.e., only the first subproblem (B.6A) was detected as infeasible. Similarly, GBD3 only needed initial cuts (B.8B) and auxiliary feasibility cuts (B.10B) generated in the first two iterations to find a feasible solution and generate feasible subproblems in subsequent iterations, i.e., the first two subproblems (B.6A) were infeasible. Despite the computational advantages of the proposed GBD framework, it is important to note that, due to the non-convex nature of the problem, there are no guarantees that the solutions obtained with GBD are local optima and that the cuts generated will not cut off unexplored feasible regions, as it has been discussed elsewhere[380]. For instance, it was found that the auxiliary cuts generated by GBD2 are cutting off the solution obtained by GBD3, thus leading to a somewhat different solution.

The best scheduling solution found for this problem (GBD3) is shown in Figure B.4A. The schedules found in the first iterations of GBD3 have processing times near their lower bounds (1h) and batch sizes near their maximum capacity (1.5 m^3 for Rl and 1 m^3 for Rs). As the iterations proceed, processing times increase, and batch sizes decrease until a tradeoff is found between the dynamic costs (f^{dyn}) and scheduling costs (f^{sch}), generating the schedule in Figure B.4A. Figure B.4B illustrates the profiles of the composition of the reactants and products involved in reaction $R3$ ($C + E \rightarrow F$) in Rl. In this case an endpoint constraint was imposed over the concentration of F to satisfy product quality requirements. Thus, the control actions implemented on the hot fluid are used at the beginning of the operation to promote the reaction (Figure B.4C). Another endpoint constraint is imposed over the temperature at the end to consider the cool down of the reactor. Thus, the control actions imposed on the flow of cold utility fluid shows a sharp increase after 1.5 hours to satisfy this temperature constraint, as shown in Figure B.4C.

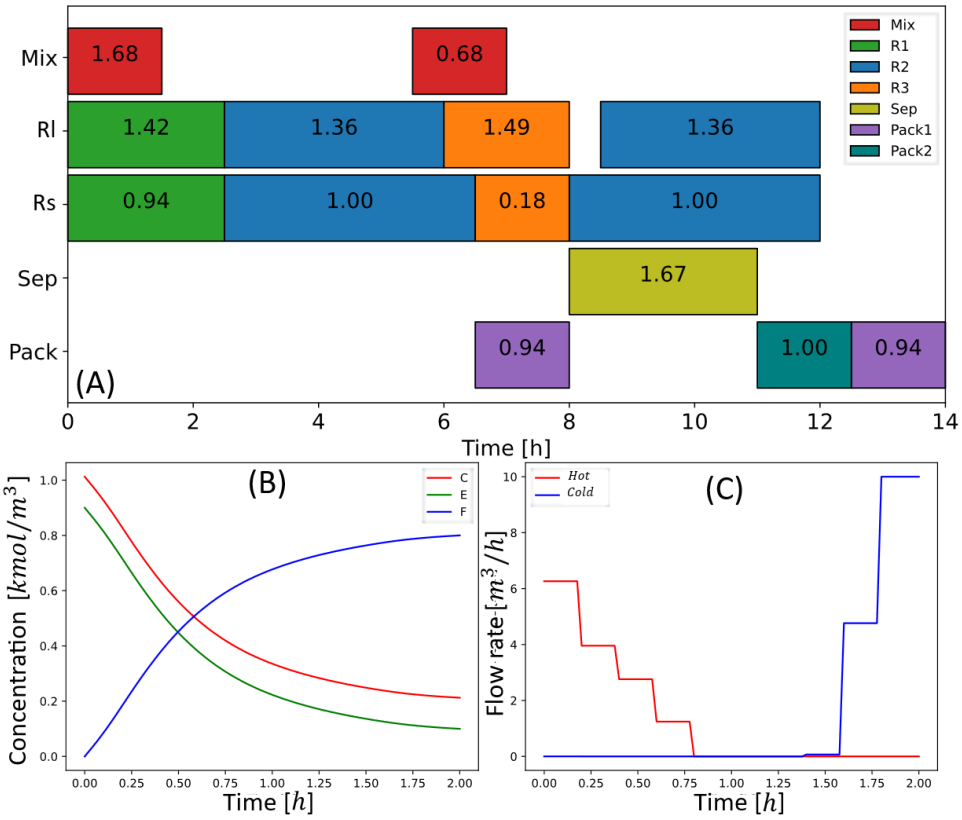


Figure B.4. A) Best scheduling solution; numbers within the colored bloks represent batch sizes (m^3). B) concentration profiles reaction $R3$ in Rl. C) Control profiles for reaction $R3$ in Rl

B.3.2 Comparison of the proposed GBD and solver DICOPT

The present case study was also solved using general purpose MINLP solver DICOPT, which also relies on a master-subproblem decomposition. DICOPT was unable to find a feasible solution to the problem after 2 iterations. The performance of DICOPT improved slightly when initial cuts (B.8B) were added to the formulation, resulting in a faster convergence of the MIP master problem. This allowed to perform 25 iterations within the time limit of 1000 seconds. Despite the improvements in CPU time, DICOPT was still unable to identify a feasible solution. DICOPT implements the outer approximation method (OA), which is generally regarded as an improved version of GBD. In a traditional master-subproblem decomposition where discrete variables are optimized in the master problem, cuts in GBD can be viewed as weaker surrogates of the cuts implemented by OA. Nevertheless, the proposed GBD strategy relies on a customized decomposition strategy and implements an enhanced strategy to handle infeasibility. This allowed the proposed GBD to perform better than the general-purpose solver DICOPT.

B.4 Summary

This study presented a novel and enhanced version of the GBD method as a valuable tool for addressing the discrete-time SSOC of batch processes, using a STN modeling approach with task modes to enable the approximated modeling of variable processing times. The primary contribution of this work is the innovative adaptation of the original GBD algorithm, introducing initial and auxiliary cuts to facilitate convergence. While the traditional feasibility cuts in the GBD method may result in slow convergence, the proposed decomposition technique stands out by substantially improving the CPU time when using customized initial and auxiliary feasibility cuts in the master problem. This innovation not only illustrates the potential of GBD but also highlights the importance of tailored algorithmic modifications in addressing complex integrated problems, which result in improved performance with respect to traditional MINLP solvers. Future work in this research involves consideration of uncertainty in processing times, test the algorithm’s performance under external perturbations in the dynamics of batch units, and under closed-loop operation.

Appendix C
Supplementary Material for Chapter 3

C.1 The D-SDA framework for superstructure optimization

When dealing with multiple discrete decision variables, each set of independent binary variables in Section 3.3.2 is organized in a vector $\boldsymbol{y}$ and expressed in terms of the vector of *external variables* $\boldsymbol{e}$, by using an appropriate *External Variables Reformulation* (EVR). In general, any discrete choice considered adds a new *external variable*, e.g., optimizing the total number of stages and two feeds results in 3 *external variables*. Furthermore, increasing the number of *external variables* results in more possible search directions; the concept of *k-neighborhood* must therefore be implemented, i.e., *k-neighborhoods* are used to explore the feasible region of *external variables*, and to establish a stopping criterion for the D-SDA. Correspondingly, the *k-neighborhood* of $\boldsymbol{e}$ is denoted by $N_k(\boldsymbol{e})$. The most general case is the ∞-*neighborhood*, which comprises every discrete point nearby $\boldsymbol{e}$ within an ℓ_∞-norm (infinite norm) less or equal to 1. The stopping criterion dictated by the ∞-*neighborhood* is: if the objective function takes its minimum value inside $N_\infty(\boldsymbol{e})$, then $\boldsymbol{e}$ is an *i-local* minimizer of the problem. Unfortunately, the ∞-*neighborhood* requires $3^{n_e} - 1$ neighbors to be verified, where n_e is the size of vector $\boldsymbol{e}$. Thus, subsets of this neighborhood have been previously introduced to execute optimization algorithms with shorter computational times [237,381]. An example is the 2-*neighborhood*, defined like the ∞-*neighborhood* but with an ℓ_2-norm (Euclidean norm), which results in $2n_e$ neighbors only. The disadvantage of this neighborhood is that it does not always guarantee *i-local* optimality; however, it can still consider possible design improvements, given an initial design. Both *k-neighborhoods* are exemplified in Figure C.1 for two *external variables*. Keeping these definitions in mind, the D-SDA is described next, following the notation summarized in Table C.1.

The D-SDA framework is summarized in Figure C.2, where the decomposed problem is mathematically defined, the input information that it requires is introduced, and the information exchanged between the IPLC, and the FNLP is sketched. The master IPLC aims to find f_{MINLP}, i.e., a local optimum of the problem. To achieve this goal, the IPLC explores the *external variables'* feasible region $E \subseteq \mathbb{Z}^{n_e}$, while optimizing the objective function $f_{FNLP}(\boldsymbol{e})$; where $f_{FNLP}(\boldsymbol{e})$ is calculated by the subproblem and E oversees the logic of the problem, e.g., logical constraints in Section 3.3.3 reformulated in the domain of *external variables* $\boldsymbol{e}$. The subproblems (FNLPs) fix integer external

variables and optimize continuous variables denoted by $\boldsymbol{x}$, where nonlinear constraints are represented with functions g_j and linear constraints as expressed as a polyhedral set S.

Table C.1. Notation used through this section

Term	Definition
argmin	Argument of the minimum
N_k	*k-neighborhood* for local search
$\boldsymbol{y}$	Vector of independent binary variables
$\boldsymbol{e}$	Vector of external variables
n_e	Number of external variables
E	*External variables'* feasible region
$\boldsymbol{x}$	Vector of continuous variables
$\boldsymbol{g}$	Nonlinear constraints
S	Convex polyhedron of binary and continuous variables
$\boldsymbol{e}^{LS}$	Candidate to local minimizer using the line search (*external variables*)
$\boldsymbol{x}^{LS}$	Candidate to local minimizer using the line search (continuous)
$\boldsymbol{\delta}$	Steepest descent direction

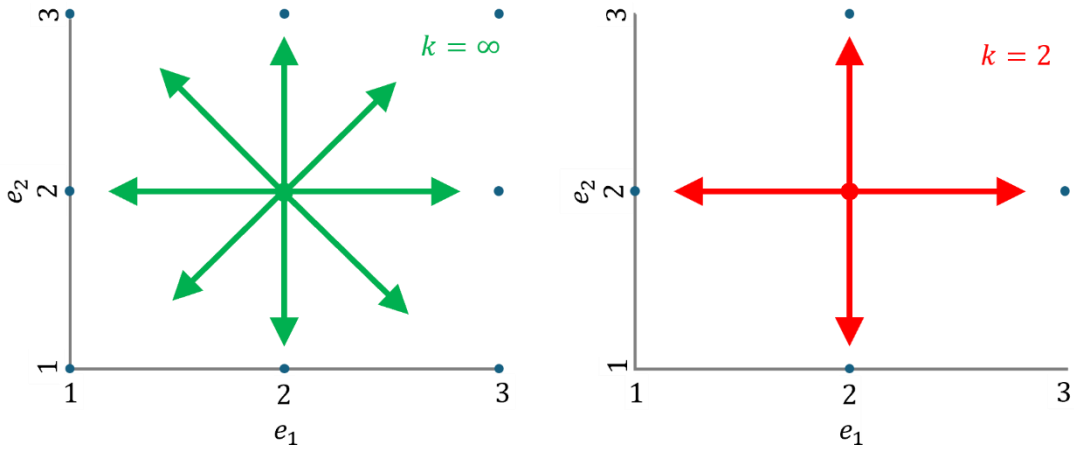


Figure C.1. Example of k-neighborhoods.

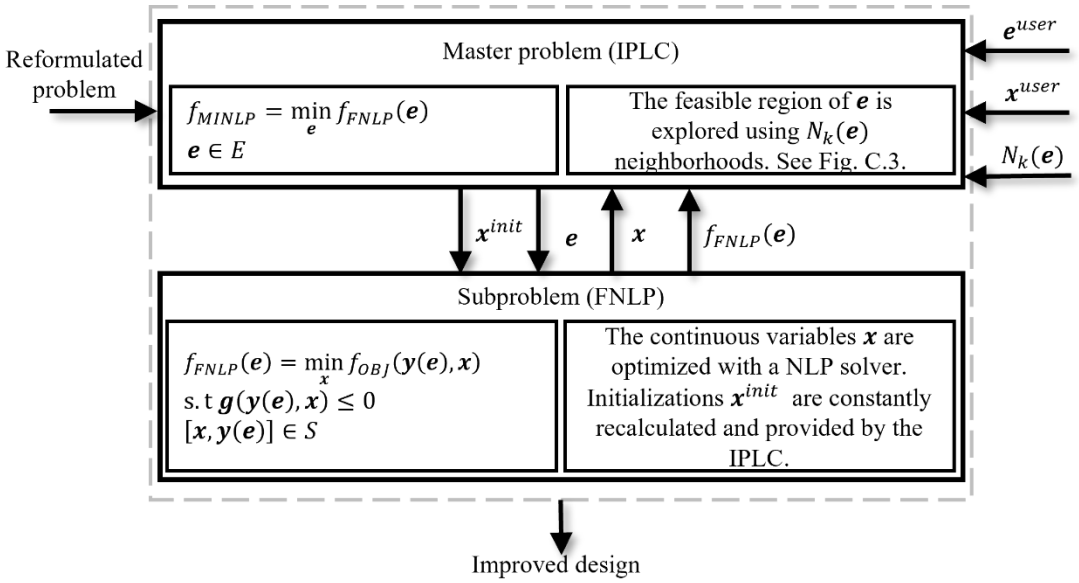


Figure C.2. The D-SDA framework. Black arrows represent the flow of information.

The D-SDA in Figure C.2 requires three inputs from the user. The first input is the neighborhood $N_k(\boldsymbol{e})$ used to perform the local search. If the FNLPs are solved to global optimality, the largest neighborhood with $k = \infty$ allows to find an *i-local* solution. In practice, global optimality for FNLPs may not be guaranteed, and computational times are a constraint; therefore, local solvers can be used instead of global solvers. Also, local searches over subsets of $N_\infty(\boldsymbol{e})$ are suggested if computational resources are limited. The second input is the initialization for the continuous and discrete variables provided by the user, i.e., $\boldsymbol{x}^{user}$ and $\boldsymbol{e}^{user}$. Regarding discrete variables, $\boldsymbol{e}^{user}$ must be feasible, i.e., $\boldsymbol{e}^{user} \in E$. In contrast, $\boldsymbol{x}^{user}$ is not required to be feasible but it must at least allow the NLP solver to find a feasible solution with the values provided. The last input is the reformulated problem using the EVR because there is no unique way to represent binary variables with *external variables*. The only known effect of the EVR occurs when a subset of the ∞-*neighborhood* is used to execute the algorithm because each EVR results in a different subset of the ∞-*neighborhood*. Thus, if an ∞-*neighborhood* verification is not going to be performed, multiple candidate solutions can be obtained by manipulating the EVR. More details about the EVR for superstructure optimization problems are provided in section D.1 of appendix D, while section H.1 of appendix H provides a more general reformulation suitable for a wider variety of problems.

The IPLC of the D-SDA interacts with the subproblems through the exchange of information. Concerning discrete decisions, the IPLC provides values for the *external variables* ($\boldsymbol{e}$), which are fixed by the FNLPs and used to compute objective function values ($f_{FNLP}(\boldsymbol{e})$). Furthermore, the IPLC and the FNLPs also interact by exchanging information related to continuous variables. Specifically, each FNLP requires an initialization for the continuous variables ($\boldsymbol{x}^{init}$), which is recalculated iteratively according to the continuous variables $\boldsymbol{x}$ reported by the subproblems. Both iterative tasks performed by the master level of the D-SDA are summarized in Figure C.3, with particular emphasis on the constantly recalculated parameter $\boldsymbol{x}^{init}$. As it is shown, the initialization provided by the user is used at the first iteration to compute the vector $\boldsymbol{x}^{init}$. This is used to initialize the first search over the *k-neighborhood.* Then, the line search is executed, where improved solutions are calculated one by one along the steepest descent direction whit incremental steps of size $\|\boldsymbol{\delta}\|_{\infty} = 1$. During the line search, initializations are again recalculated each time a new feasible design is found along the steepest-descent direction $\boldsymbol{\delta}$, until the objective function starts to deteriorate. After that, the line search results in a new candidate for local minimizer, with continuous variables $\boldsymbol{x}^{LS}$. These continuous variables are used to update the initializations for the next iteration.

A final concern regarding the D-SDA's implementation is the tradeoff between computational time and *i-local* optimality guarantees. As discussed above, the D-SDA can be executed to i-local optimality when the ∞-neighborhood is used, but subsets of this neighborhood can save computational time. Consequently, the algorithm in Figure C.4 to execute the D-SDA is proposed to decrease the number of ∞-*neighborhood* evaluations and still guarantee *i-local* optimality. This was previously used to design a CD column with an equilibrium model, where only one local search with the 2-*neighborhood* and one verification with the ∞-*neighborhood* was required [12]. Specifically, thirteen iterations of the local search with 2-*neighborhoods* took the same amount of time as one ∞-*neighborhood* verification.

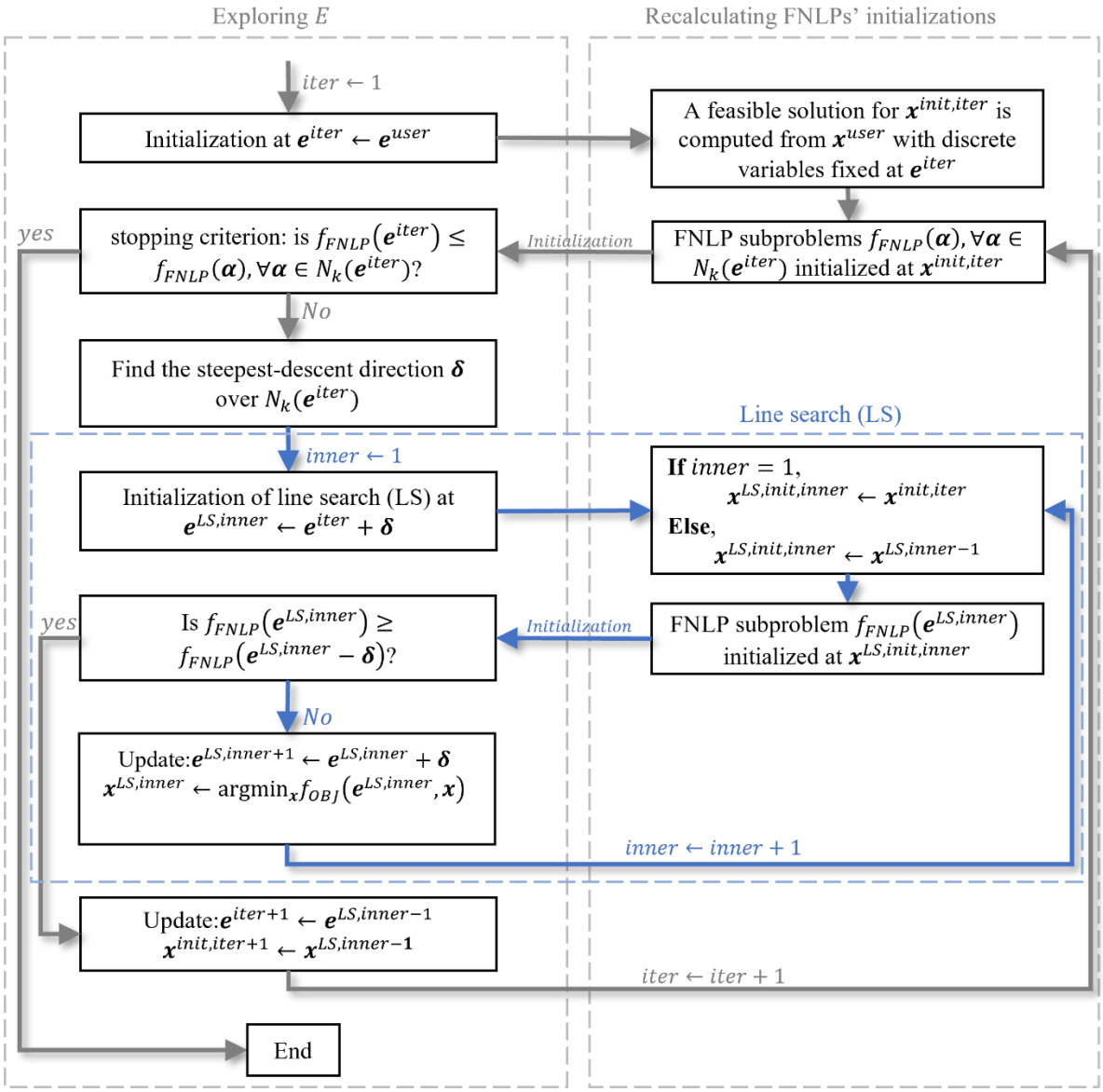


Figure C.3. D-SDA's master problem: exploring external variables' feasible region and recalculating initializations. Grey and blue arrows: algorithm steps. $init$: initialization. $iter$: iteration. $inner$: line search iterations. E is assumed to be bounded to guarantee convergence with a finite number of iterations. Also, $f_{FNLP}(\boldsymbol{e}) = +\infty$ for infeasible NLP problems.

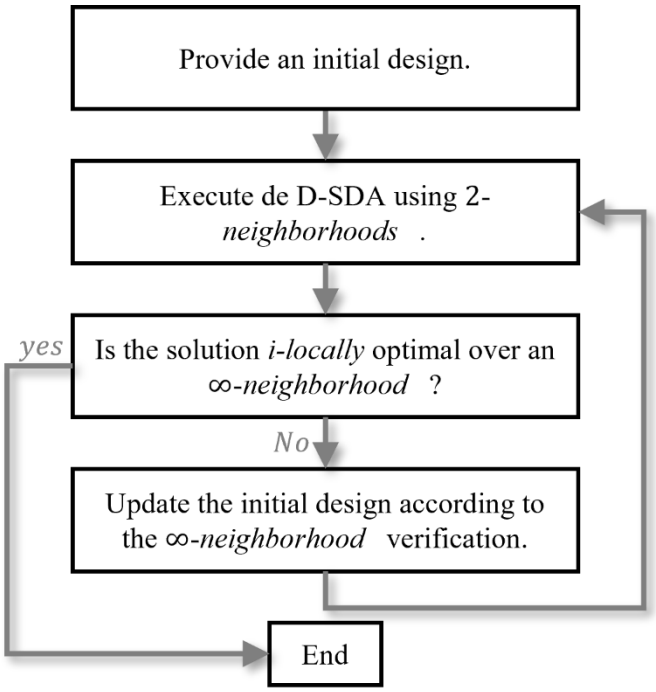


Figure C.4. Time-efficient use of the D-SDA.

Appendix D

Supplementary Material for Chapter 4

D.1 Reformulation with external variables

The reformulation with *external variables* (or EVR) is a strategy that was originally proposed to simplify the solution of challenging Mixed-Integer Nonlinear Programming (MINLP) superstructure optimization problems in chemical engineering[4]. This reformulation was recently generalized to Generalized Disjunctive Programming (GDP) formulations, which offer modeling advantages with respect to traditional MINLP models, e.g., disjunctions and Boolean variables in GDP make the modeling process more intuitive[3]. To introduce the reformulation with *external variables*, consider the superstructure in Figure D.1, where Boolean variables are used to represent six discrete decisions: the number of stages (YB_n), the location of two feeds ($YF_{1,n}$ and $YF_{2,n}$), and the location of three catalytic stages ($YC_{1,n}, YC_{2,n}$ and $YC_{3,n}$). For instance, $YB_{11} = True$ indicates that the final column design has 10 internal stages. In Figure D.1B, a vector of *external variables* $\boldsymbol{e} = [e_1, e_2, \dots, e_{n_e}]$ is used to represent a feasible solution of these discrete choices. This representation is advantageous because the number of discrete decisions can be reduced from 258 Boolean variables to $n_e = 6$ external variables in this example. Besides, the resulting external variables can be efficiently optimized using an adequate solver, e.g., a Logic-Based Benders Decomposition (LD-BD) or Logic-based Discrete-Steepest Descent Algorithm (LD-SDA).

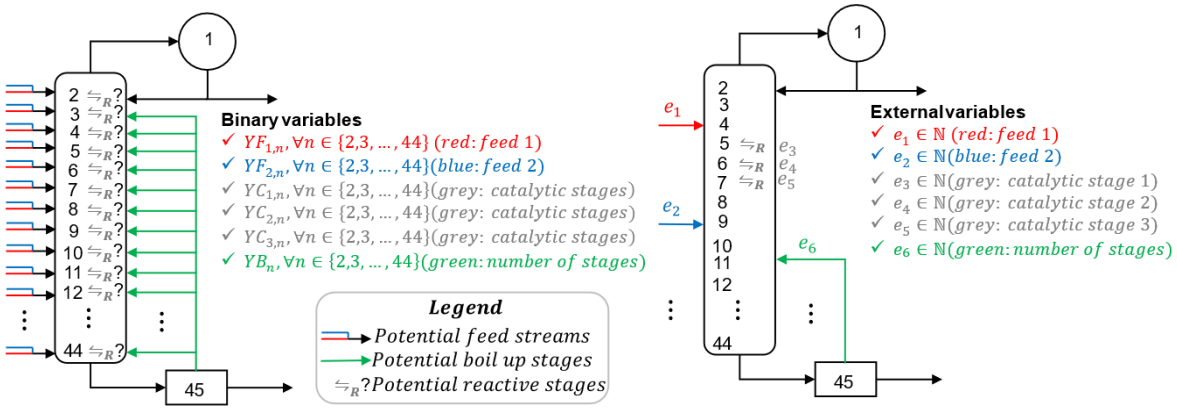


Figure D.1. RD superstructure. A) original formulation, B) reformulation with external variables.

In general, the external variables reformulation can be applied over any vector of Boolean variables $\boldsymbol{Y_R} = [Y_{s(1)}, Y_{s(2)}, \dots, Y_{s(|S_R|)}]$ such that the conditions in Eqs. (D.1) and (D.2) are satisfied. This vector contains elements $Y_{s(a)}, \forall a \in \{1,2,\dots,|S_R|\}$, where a is the position of index $s = s(a)$ in set S_R, i.e., $s(a) \in S_R, \forall a \in \{1,2,\dots,|S_R|\}$. Set S in Eq. (D.1) must be finite and ordered, i.e., it either represents discrete locations, discrete points in time or any other discrete ordered index (see more details in section H.1 of appendix H). S_R is a subset of S and contains those indices of the Boolean variables that satisfy Eq. (D.2), while maintaining the same order defined by S. In turn, Eq. (D.2) indicates that only one variable in vector $\boldsymbol{Y_R}$ takes the value of $True$. For instance, consider the superstructure example in Figure D.1 and Boolean variables $YF_{1,n}, \forall n \in \{2,3,\dots,44\}$. In this case, $s(a) = n$, $Y_{s(a)} = YF_{1,s(a)}$, and the finite ordered set $S = \{1,2,\dots,45\}$ represents ordered stages. This feed can only be allocated within internal stages defined by $S_R = \{2,3,\dots,44\} \subseteq S$ (i.e., Eq. (D.1) holds). Also, only one feed of stream 1 is allowed within internal stages S_R, thus Eq. (D.2) must be imposed. Hence, the reformulation with external variables can be applied over $[YF_{1,2}, YF_{1,3} \dots, YF_{1,44}]$.

$$S_R \subseteq S = finite\ ordered\ set \tag{D.1}$$

$$\veebar_{s(a) \in S_R} Y_{s(a)}, with\ \ Y_{s(a)} \in \{True, False\} \tag{D.2}$$

If the conditions to perform the reformulation are satisfied, variables in $\boldsymbol{Y_R}$ can be expressed in terms of external variable e (Eq. (D.3)), as described in Eq. (D.4). Note that Eqs. (D.3) and (D.4) correspond to the case of a single external variable, i.e., $\boldsymbol{e} = [e]$. Eq. (D.3) constraints external variable e so that constraint in Eq. (D.2) is never violated. The dependence of $\boldsymbol{Y_R}$ on e is denoted with vector function $\boldsymbol{Y_R}(e)$. For example, consider again variables $YF_{1,n}, \forall n \in \{2,3,\dots,44\}$, with $s(a) = n(a)$, $Y_{s(a)} = YF_{1,s(a)}$, and $S_R = \{2,3,\dots,44\}$. According to Eq. (D.3), the external variable that represents $YF_{1,n}$ is e, with $1 \leq e \leq 43$, and $e \in \mathbb{N}$. Variable e indicates the location of this feed within the internal stages of the column S_R (discarding the condenser and the reboiler). According to Eq. (D.4), $YF_{1,s(a)}$ only takes the value of $True$ when $a = e$, or in this case, when the external variable value coincides with index $n - 1$ of the original variable $YF_{1,n}$.

$$1 \leq e \leq |S_R|,\ \ e \in \mathbb{N} \tag{D.3}$$

$$\boldsymbol{Y_R}(e) \coloneqq \; Y_{s(a)} = \begin{cases} True, a = e \\ False, a \neq e \end{cases}, \forall a \in \{1,2,\dots,|S_R|\} \tag{D.4}$$

In general, there will be n_e vectors for which the reformulation can be applied (i.e., $\boldsymbol{Y_R} = [\boldsymbol{Y_{R,1}}, \boldsymbol{Y_{R,2}}, \dots, \boldsymbol{Y_{R,n_e}}]$), which define the vector of *external variables* $\boldsymbol{e} = [e_1, e_2, \dots, e_{n_e}]$, e.g., in Figure D.1 there are $n_e = 6$ vectors, or equivalently, six external variables. In this general case, notation $\boldsymbol{Y_R(e)}$ is used to indicate that all the vectors of Boolean variables in $\boldsymbol{Y_R}$ are expressed as a function of the external variables vector $\boldsymbol{e}$, i.e., each external variable in $\boldsymbol{e} = [e_1, e_2, \dots, e_{n_e}]$ and each vector $\boldsymbol{Y_{R,1}}(e_1)$, $\boldsymbol{Y_{R,2}}(e_2)$, … , $\boldsymbol{Y_{R,n_e}}(e_{n_e})$ is defined following Eqs. (D.3) and (D.4). Note that this reformulation can be extended to problems where exactly q elements of vector $\boldsymbol{Y_R}$ should remain activated[4]. As explained above, the reformulation with external variables can be applied to the optimal design of superstructures involving chemical processes, when Boolean variables follow a special ordered structure.

D.2 Proof of theorems

This section presents the proofs of Theorems 4.1 and 4.2. For the demonstration of Theorem 4.2, first consider Definition D.1 and Lemmas D.1-D.6. This section refers to the equations presented in chapter 4 and the LD-BD algorithm in Figure 4.1.

Theorem 4.1. *Assume that* $f^*: \mathbb{Z}^{n_e} \mapsto \mathbb{R} \cup \{+\infty\}$ *is an integrally convex function. Then, Benders cut* $\langle \boldsymbol{p}^{\hat{k}}, \boldsymbol{e} \rangle + \alpha^{\hat{k}}$ *obtained from* Eq. (4.8) *satisfies* Eqs. (4.5) *and* (4.6).

Proof of Theorem 4.1. According to Equations (4.7) and (4.8), $\langle \boldsymbol{p}^{\hat{k}}, \boldsymbol{e} \rangle + \alpha^{\hat{k}} \leq f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in \mathbb{Z}^{n_e}$. By definition (i.e., Eq. (4.3)), $f^*(\boldsymbol{e}) \leq f\big(\boldsymbol{x}, \boldsymbol{Y_R}(\boldsymbol{e})\big)\ \forall \boldsymbol{x} \in X, \forall \boldsymbol{e} \in E$, hence $\langle \boldsymbol{p}^{\hat{k}}, \boldsymbol{e} \rangle + \alpha^{\hat{k}} \leq f\big(\boldsymbol{x}, \boldsymbol{Y_R}(\boldsymbol{e})\big)\ \forall \boldsymbol{x} \in X, \forall \boldsymbol{e} \in E$, i.e., the inequality in Eq. (4.5) always holds, independently of the convexity properties of f^*. Given that f^* is integrally convex, then the convex envelope of f^* denoted by $\bar{f}^*$ satisfies that $\bar{f}^*(\boldsymbol{e}) = f^*(\boldsymbol{e})\ \forall \boldsymbol{e} \in \mathbb{Z}^{n_e}$[6]; in particular, $\bar{f}^*\big(\boldsymbol{e}^{\hat{k}}\big) = f^*\big(\boldsymbol{e}^{\hat{k}}\big) = \langle \boldsymbol{p}^{\hat{k}}, \boldsymbol{e}^{\hat{k}} \rangle + \alpha^{\hat{k}}$, which completes the proof. ■

Definition D.1. *Consider iteration k. Vector of external variables* $\boldsymbol{\eta} \in D^k$ *is called* D^k*-optimal (i.e., optimal with respect to* D^k*) when* $f^*(\boldsymbol{\eta}) \leq f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in D^k$*, i.e.,* $\boldsymbol{\eta} \in D^k$ *is* D^k*-optimal when* $\boldsymbol{\eta} \in \left\{\boldsymbol{e} \in D^k : f^*(\boldsymbol{e}) = \min_{\boldsymbol{\alpha} \in D^k}\{f^*(\boldsymbol{\alpha})\}\right\}$.

Lemma D.1. *Assume that* $\boldsymbol{e}^{init}$ *returns a feasible solution for the problem. Then, at each iteration k of the algorithm there are two possibilities for the current trial value of external variables* $\boldsymbol{e}^k$*:*

1) $\boldsymbol{e}^k$ *is* D^k*-optimal*
2) $\boldsymbol{e}^k$ *is not* D^k*-optimal, i.e., there exists at least one* $\boldsymbol{e} \in D^k$ *such that* $f^*(\boldsymbol{e}) < f^*(\boldsymbol{e}^k)$.

Proof of Lemma D.1. Since $\boldsymbol{e}^1 = \boldsymbol{e}^{init}$ returns a feasible solution, $f^*(\boldsymbol{e}^1) \in D^k$ is finite. Thus, it is possible to have $f^*(\boldsymbol{e}^1) < f^*(\boldsymbol{e}^k)$, meaning that $\boldsymbol{e}^k$ is not D^k-optimal. On the contrary, when $f^*(\boldsymbol{e}^1) \geq f^*(\boldsymbol{e}^k)$, there are two possibilities in this case: 1) $f^*(\boldsymbol{e}^1)$ and $f^*(\boldsymbol{e}^k)$ are the only members within D^k or 2) there are other elements in D^k. In case 1) $\boldsymbol{e}^k$ is D^k-optimal, and in case 2) $\boldsymbol{e}^k$ can be either be D^k-optimal if the objective functions of the other members are all greater or equal than $f^*(\boldsymbol{e}^k)$, or not D^k-optimal if there is at least one other member in D^k with objective strictly less than $f^*(\boldsymbol{e}^k)$. As explained above, all possibilities lead to either one of two cases. ■

Lemma D.2. *Assume that the feasible region E of external variables is bounded, and that none of the subproblems has an unbounded objective function. If* $\boldsymbol{\eta} \in D^k$ *is not* D^k*-optimal, then there is at least one* $\boldsymbol{\theta}$ *different from* $\boldsymbol{\eta}$ *such that* $\boldsymbol{\theta} \in D^k$ *and* $\boldsymbol{\theta}$ *is* D^k*-optimal.*

Proof of Lemma D.2. $D^k \subseteq E$ and E is bounded, hence D^k has a finite number of elements. Now take $A^k = \{f^*(\boldsymbol{\alpha}) \in \mathbb{R} : \boldsymbol{\alpha} \in D^k\}$. Since D^k is finite, A^k is finite too. Also, $\boldsymbol{\eta}$ is not D^k-optimal, hence there exists at least one $\boldsymbol{e} \in D^k$ such that $f^*(\boldsymbol{e}) < f^*(\boldsymbol{\eta})$. If $f^*(\boldsymbol{\eta}) < +\infty$, then $f^*(\boldsymbol{\eta}) \in \mathbb{R}$, which means that $f^*(\boldsymbol{\eta}) \in A^k$, i.e., A^k is not empty. Otherwise, if $f^*(\boldsymbol{\eta}) = +\infty$, then $f^*(\boldsymbol{e}) < +\infty$, which indicates that $f^*(\boldsymbol{e}) \in A^k$. In any case, A^k is a finite nonempty subset of $\mathbb{R}$, hence A^k has a minimum $f^*(\boldsymbol{\theta})$, with $\boldsymbol{\theta} \in D^k$ [382]. $f^*(\boldsymbol{\theta}) \in \mathbb{R}$ and there are not unbounded subproblems (i.e., subproblems with objective equal to $-\infty$), thus $f^*(\boldsymbol{\theta}) \leq f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in D^k$. This proves that there exists a $\boldsymbol{\theta} \in D^k$ such that $\boldsymbol{\theta}$ is D^k-optimal. Note that $\boldsymbol{\eta} \neq \boldsymbol{\theta}$ can be easily proven by contradiction as follows. If $\boldsymbol{\eta}$ is not D^k-optimal, then

there exists $\boldsymbol{e} \in D^k$ such that $f^*(\boldsymbol{e}) < f^*(\boldsymbol{\eta})$. Having $\min_{\boldsymbol{\alpha}\in D^k}\{f^*(\boldsymbol{\alpha})\} \le f^*(\boldsymbol{e}) < f^*(\boldsymbol{\eta})$ means that any $\boldsymbol{\theta}$ with $f^*(\boldsymbol{\theta}) = \min_{\boldsymbol{\alpha}\in D^k}\{f^*(\boldsymbol{\alpha})\}$ is not equal to $\boldsymbol{\eta}$, i.e., having $\boldsymbol{\eta} = \boldsymbol{\theta}$ would indicate that $\min_{\boldsymbol{\alpha}\in D^k}\{f^*(\boldsymbol{\alpha})\} = f^*(\boldsymbol{\eta})$, which is a contradiction. ■

Lemma D.3. *Consider the optimization problems in* Eqs. (D.5)-(D.7), *with* $f^*(\boldsymbol{e}) = +\infty, \forall \boldsymbol{e} \in \mathbb{Z}^{n_e} \backslash D^{k+1}$.

$$\beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) = \max_{\boldsymbol{p}\in\mathbb{R}^{n_e},\alpha\in\mathbb{R}} \{\langle \boldsymbol{p}, \boldsymbol{e}^{k+1}\rangle + \alpha : \langle \boldsymbol{p}, \boldsymbol{e}\rangle + \alpha \le f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in D^{k+1}\} \tag{D.5}$$

$$\bar{f}^*(\boldsymbol{e}^{k+1}) = \max_{\boldsymbol{p}\in\mathbb{R}^{n_e},\alpha\in\mathbb{R}} \{\langle \boldsymbol{p}, \boldsymbol{e}^{k+1}\rangle + \alpha : \langle \boldsymbol{p}, \boldsymbol{e}\rangle + \alpha \le f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in \mathbb{Z}^{n_e}\} \tag{D.6}$$

$$\tilde{f}^*(\boldsymbol{e}^{k+1}) = \max_{\boldsymbol{p}\in\mathbb{R}^{n_e},\alpha\in\mathbb{R}} \{\langle \boldsymbol{p}, \boldsymbol{e}^{k+1}\rangle + \alpha : \langle \boldsymbol{p}, \boldsymbol{e}\rangle + \alpha \le f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in N(\boldsymbol{e}^{k+1})\} \tag{D.7}$$

Eq. (D.5) *represents the optimization problem required to obtain cut* $\beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e})$ at $\boldsymbol{e}^{k+1}$, Eq. (D.6) *represents the value of the convex envelope* $\bar{f}^*$ *of* f^* *at* $\boldsymbol{e}^{k+1}$, *and* Eq. (D.7) *is the value of the local convex extension* $\tilde{f}^*$ *of* f^* *at* $\boldsymbol{e}^{k+1}$, *with* $N(\boldsymbol{e}^{k+1}) = \left\{\boldsymbol{e} \in \mathbb{Z}^{n_e} : \left\|\boldsymbol{e} - \boldsymbol{e}^{k+1}\right\|_\infty < 1\right\}$. *Then:*

1) $\bar{f}^*(\boldsymbol{e}^{k+1}) \le \beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) \le \tilde{f}^*(\boldsymbol{e}^{k+1})$.

Furthermore, if $\boldsymbol{e}^{k+1}$ *is* D^{k+1}*-optimal, then the following statements are true:*

2) $\bar{f}^*(\boldsymbol{e}^{k+1}) = f^*(\boldsymbol{e}^{k+1})$
3) $\beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) = f^*(\boldsymbol{e}^{k+1})$

Proof of Lemma D.3. Proof of 1)-3) is shown below.

1) By definition, $\tilde{f}^*(\boldsymbol{e}^{k+1}) = f^*(\boldsymbol{e}^{k+1})$[6], and $\beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) \le f^*(\boldsymbol{e}^{k+1})$ because $\boldsymbol{e}^{k+1} \in D^{k+1}$. Hence $\beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) \le \tilde{f}^*(\boldsymbol{e}^{k+1})$. The other inequality (i.e., $\bar{f}^*(\boldsymbol{e}^{k+1}) \le \beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1})$) can be demonstrated by contradiction. First, define the feasible region of problem D.5 as $S_{D5} = \{[\boldsymbol{p}, \alpha] \in \mathbb{R}^{n_e+1} : \langle \boldsymbol{p}, \boldsymbol{e}\rangle + \alpha \le f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in D^{k+1}\}$, and the feasible region of problem D.6 as $S_{D6} = \{[\boldsymbol{p}, \alpha] \in \mathbb{R}^{n_e+1} : \langle \boldsymbol{p}, \boldsymbol{e}\rangle + \alpha \le f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in \mathbb{Z}^{n_e}\}$. $[\boldsymbol{p}^{D6}, \alpha^{D6}]$ is the optimal solution of

D.6, hence it satisfies that $[\boldsymbol{p}^{D6}, \alpha^{D6}] \in S_{S6}$, and $\bar{f}^*(\boldsymbol{e}^{k+1}) = \langle \boldsymbol{p}^{D6}, \boldsymbol{e}^{k+1} \rangle + \alpha^{D6}$. Note that $S_{D6} \subseteq S_{D5}$, hence $[\boldsymbol{p}^{D6}, \alpha^{D6}] \in S_{D5}$. Now, assume that $\beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) < \bar{f}^*(\boldsymbol{e}^{k+1})$. By definition, $\beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) \geq \langle \boldsymbol{p}, \boldsymbol{e}^{k+1} \rangle + \alpha, \forall [\boldsymbol{p}, \alpha] \in S_{D5}$, hence $\bar{f}^*(\boldsymbol{e}^{k+1}) > \langle \boldsymbol{p}, \boldsymbol{e}^{k+1} \rangle + \alpha, \forall [\boldsymbol{p}, \alpha] \in S_{D5}$. The previous statement contradicts the fact that there exists $[\boldsymbol{p}^{D6}, \alpha^{D6}] \in S_{D5}$ such that $\bar{f}^*(\boldsymbol{e}^{k+1}) = \langle \boldsymbol{p}^{D6}, \boldsymbol{e}^{k+1} \rangle + \alpha^{D6}$.

2) This will be demonstrated by showing that $\bar{f}^*(\boldsymbol{e}^{k+1}) \leq f^*(\boldsymbol{e}^{k+1})$ and $\bar{f}^*(\boldsymbol{e}^{k+1}) \geq f^*(\boldsymbol{e}^{k+1})$. $\leq$: By constraints in Eq. D.6, inequality $\bar{f}^*(\boldsymbol{e}^{k+1}) \leq f^*(\boldsymbol{e}^{k+1})$ holds because $\boldsymbol{e}^{k+1} \in \mathbb{Z}^{n_e}$. $\geq$: Since $\boldsymbol{e}^{k+1}$ is D^{k+1}-optimal, $f^*(\boldsymbol{e}^{k+1}) \leq f^*(\boldsymbol{e}), \forall \boldsymbol{e} \in \mathbb{Z}^{n_e}$. This means that constant function $h(\boldsymbol{e}) = f^*(\boldsymbol{e}^{k+1})$ is a convex underestimating function of f^*. By definition of the convex envelope, $\bar{f}^*(\boldsymbol{e}^{k+1}) \geq f^*(\boldsymbol{e}^{k+1}) = h(\boldsymbol{e})$ because $\bar{f}^*$ is greater than or equal to all convex functions that lie under f^*, including $h(\boldsymbol{e})$.

3) Given that $\tilde{f}^*(\boldsymbol{e}^{k+1}) = f^*(\boldsymbol{e}^{k+1})$[6], having $\bar{f}^*(\boldsymbol{e}^{k+1}) = f^*(\boldsymbol{e}^{k+1})$ and $\bar{f}^*(\boldsymbol{e}^{k+1}) \leq \beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) \leq \tilde{f}^*(\boldsymbol{e}^{k+1})$ means that $\beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) = f^*(\boldsymbol{e}^{k+1})$. ■

Lemma D.4. *Assume that the master problem is always solved to optimality and that an absolute optimality gap $\epsilon = 0$ is stablished. If after step 5 of iteration k the LD-BD algorithm has not stopped, $\exists \hat{k}(\boldsymbol{e}^{\hat{k}} = \boldsymbol{e}^{k+1}), \hat{k} \in \{1,2,\dots,k\}$ and $\boldsymbol{e}^{k+1} = \boldsymbol{e}^{\hat{k}}$ is D^k-optimal, then the algorithm is going to stop in iteration $k+1$.*

Proof of Lemma D.4. The requirement to stop in iteration $k+1$ is that $z^{k+2} = f^*(\boldsymbol{e}^{k+1})$. First, it will be shown that $z^{k+2} \geq \beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1})$, which will be used latter to show that $z^{k+2} = f^*(\boldsymbol{e}^{k+1})$. Statement $\exists \hat{k}(\boldsymbol{e}^{\hat{k}} = \boldsymbol{e}^{k+1}), \hat{k} \in \{1,2,\dots,k\}$ means that $\boldsymbol{e}^{k+1}$ is a trial point that was explored before iteration k. This implies that the solution of the master problem at iteration k satisfies that $z^{k+1} \geq \beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1})$, i.e., cut $\beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}) \leq z$ appears in the constraints of the master problem at iteration k, and it must be satisfied by $[z^{k+1}, \boldsymbol{e}^{k+1}]$. Another implication of the statement $\exists \hat{k}(\boldsymbol{e}^{\hat{k}} = \boldsymbol{e}^{k+1}), \hat{k} \in \{1,2,\dots,k\}$ is that $D^{k+1} = D^k$. Consequently, Benders cuts remain the same between iterations k and $k+1$, i.e., the same master problem is solved to optimality in iterations k and $k+1$. Thus, the optimal objective of the master problem remains unchanged, i.e., $z^{k+1} = z^{k+2}$. Equality $z^{k+1} = z^{k+2}$ indicates that $z^{k+2} \geq \beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1})$.

Now, it will be shown that $z^{k+2} = f^*(\boldsymbol{e}^{k+1})$ by contradiction. First of all, note that all cuts at iteration $k+1$ are recalculated to satisfy $\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}^{k+1}) \leq f^*(\boldsymbol{e}^{k+1}), \forall\, \hat{k} \in \{1,2,\dots,k+1\}$. Since $D^{k+1} = D^k$, $\boldsymbol{e}^{k+1}$ is D^{k+1}-optimal and $\beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) = f^*(\boldsymbol{e}^{k+1})$ (see Lemma D.3). This means that $z^{k+2} \geq \beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) \geq \beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}^{k+1}), \forall\, \hat{k} \in \{1,2,\dots,k\}$. Having $z^{k+2} > \beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1})$ would indicate that there is an ϵ such that $z^{k+2} > \epsilon > \beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) \geq \beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}^{k+1}), \forall\, \hat{k} \in \{1,2,\dots,k\}$, i.e., ϵ would be a feasible objective function of the master problem that is better than z^{k+2}. This would contradict the fact that z^{k+2} is the optimal objective function of the master problem. Hence, $z^{k+2} = \beta_{\boldsymbol{e}^{k+1}}(\boldsymbol{e}^{k+1}) = f^*(\boldsymbol{e}^{k+1})$, which is the condition required by the algorithm to stop at iteration $k+1$. ■

Lemma D.5. *Assume that the feasible region E of external variables is bounded, the master problem is always solved to optimality, an absolute optimality gap $\epsilon = 0$ is stablished, and none of the subproblems has an unbounded objective function. If after step 4 of iteration k the algorithm has not stopped, $\exists \hat{k}(\boldsymbol{e}^{\hat{k}} = \boldsymbol{e}^{k+1}), \hat{k} \in \{1,2,\dots,k\}$ and $\boldsymbol{e}^{k+1} = \boldsymbol{e}^{\hat{k}}$ is not D^k-optimal, then the algorithm is going to update $\boldsymbol{e}^{k+1}$ as $\boldsymbol{e}^{k+1} \leftarrow \underset{\boldsymbol{e} \in D^k}{argmin}\{f^*(\boldsymbol{e})\}$ in step 5. Furthermore, the updated $\boldsymbol{e}^{k+1}$ can either be an unexplored trial solution (i.e., $\boldsymbol{e}^{k+1} \neq \boldsymbol{e}^{\hat{k}}, \forall \hat{k} \in \{1,2,\dots,k\}$) or cause the algorithm to stop in iteration $k+1$.*

Proof of Lemma D.5. If after iteration k the algorithm has not stopped, then the algorithm will move to step 5. In this step, the algorithm verifies that $\exists \hat{k}(\boldsymbol{e}^{\hat{k}} = \boldsymbol{e}^{k+1}), \hat{k} \in \{1,2,\dots,k\}$ and $\boldsymbol{e}^{k+1}$ is not D^k-optimal. In this case these conditions are satisfied, thus the algorithm updates $\boldsymbol{e}^{k+1}$ as $\boldsymbol{e}^{k+1} \leftarrow \underset{\boldsymbol{e} \in D^k}{\text{argmin}}\{f^*(\boldsymbol{e})\}$. Note that $\underset{\boldsymbol{e} \in D^k}{\text{argmin}}\{f^*(\boldsymbol{e})\}$ exists, as demonstrated in Lemma D.2. Although the updated $\boldsymbol{e}^{k+1}$ belongs to D^k, $\boldsymbol{e}^{k+1}$ may be a trial solution that was not explored in previous iterations, thus leading to one of the outcomes of the lemma, e.g., $\boldsymbol{e}^{k+1}$ may be a neighbor of a previous trial solution. Alternatively, if the updated $\boldsymbol{e}^{k+1}$ was previously explored, then $\boldsymbol{e}^{k+1}$ will satisfy the requirements in Lemma D.4, which guarantee that the algorithm will stop in iteration $k+1$. ■

Lemma D.6. *Assume that $\boldsymbol{e}^{init}$ returns a feasible solution for the problem, the master problem is always solved to optimality, an absolute optimality gap $\epsilon = 0$ is stablished, the feasible region E of*

external variables is bounded, and that none of the subproblems is unbounded. If at iteration k the algorithm has not stopped, then the algorithm is either going to stop in iteration $k+1$ or $\boldsymbol{e}^{k+1}$ will be an unexplored trial value, i.e., $\boldsymbol{e}^{k+1} \neq \boldsymbol{e}^{\hat{k}}, \forall \hat{k} \in \{1,2,\dots,k\}$.

Proof of Lemma D.6. At each iteration of the algorithm, there are two possibilities for the $\boldsymbol{e}^{k+1}$ obtained in step 3: $\boldsymbol{e}^{k+1}$ is an unexplored trial solution (i.e., $\boldsymbol{e}^{k+1} \neq \boldsymbol{e}^{\hat{k}}, \forall \hat{k} \in \{1,2,\dots,k\}$) and $\boldsymbol{e}^{k+1}$ is an explored trial solution (i.e., $\exists \hat{k}\left(\boldsymbol{e}^{\hat{k}} = \boldsymbol{e}^{k+1}\right), \hat{k} \in \{1,2,\dots,k\}$). The first case trivially leads to one of the conclusions of Lemma D.6. In the second case, there are again two possibilities: 1) $\boldsymbol{e}^{k+1}$ is D^{k+1}-optimal, and 2) $\boldsymbol{e}^{k+1}$ is not D^{k+1}-optimal (see Lemma D.1). In both cases, the fact that $\boldsymbol{e}^{k+1}$ is a previously evaluated trial solution indicates that $D^{k+1} = D^k$. In case 1) $\exists \hat{k}\left(\boldsymbol{e}^{\hat{k}} = \boldsymbol{e}^{k+1}\right), \hat{k} \in \{1,2,\dots,k\}$ and $\boldsymbol{e}^{k+1}$ is D^k-optimal, hence the algorithm will stop in iteration $k+1$ (see Lemma D.4). In case 2), $\exists \hat{k}\left(\boldsymbol{e}^{\hat{k}} = \boldsymbol{e}^{k+1}\right), \hat{k} \in \{1,2,\dots,k\}$ and $\boldsymbol{e}^{k+1}$ is not D^k-optimal, hence the algorithm will replace $\boldsymbol{e}^{k+1}$ by $\underset{\boldsymbol{e} \in D^k}{\operatorname{argmin}}\{f^*(\boldsymbol{e})\}$. This new $\boldsymbol{e}^{k+1}$ can be an unexplored trial value, i.e., $\boldsymbol{e}^{k+1} \neq \boldsymbol{e}^{\hat{k}}, \forall \hat{k} \in \{1,2,\dots,k\}$, or cause the algorithm to stop in iteration $k+1$ (see Lemma D.5). ■

Theorem 4.2. *Assume that an absolute optimality gap $\epsilon = 0$ is stablished, $\boldsymbol{e}^{init}$ returns a feasible solution for the problem, the master problem is always solved to optimality, none of the subproblems has an unbounded objective function, and the feasible region of external variables E is bounded, then the LD-BD algorithm stops in a finite number of iterations, and the solution obtained is feasible and locally optimal.*

Proof of Theorem 4.2. The proof of this theorem is divided in two parts. In the first part, it is shown that the LD-BD algorithm stops in a finite number of iterations. In the second part, it is demonstrated that the solution obtained with LD-BD is feasible and locally optimal.

- *Part 1:* The feasible region of external variables E is bounded, hence it only contains a finite number of feasible trial vectors of external variables $\boldsymbol{e} \in E$. Assume that the algorithm never converges, meaning that the gap does not close after an infinite number of iterations. This suggests that for every $k \in \{1,2,\dots,\infty\}$ the algorithm does not stop. According to Lemma D.6, the only option that prevents termination of the algorithm is to keep obtaining unexplored trial values for the external variables indefinitely, however this contradicts the assumption that the

feasible region of external variables E is bounded. Thus, the algorithm can only explore a finite number of possible external variable values, indicating that it must eventually terminate. ■

- *Part 2*: First, it will be shown that the solution obtained is feasible by contradiction. Assume that the algorithm has stopped (i.e., $z^{k+1} = f^*(\boldsymbol{e}^k)$), and that the solution obtained is infeasible (i.e., $z^{k+1} = f^*(\boldsymbol{e}^k) = +\infty$). Since $\boldsymbol{e}^{init} = \boldsymbol{e}^1$ results in a feasible solution, $f^*(\boldsymbol{e}^1) < +\infty$. By definition $\beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}^1) \leq f^*(\boldsymbol{e}^1), \forall\, \hat{k} \in \{1,2,\dots,k\}$ (see Eqs. (4.11) and (4.12)), hence another feasible objective function for the master problem is $f^*(\boldsymbol{e}^1)$, i.e., there is another solution for the master problem with an objective function better than z^{k+1}. This contradicts the fact that z^{k+1} is an optimal solution of the master problem, thus the solution obtained cannot be infeasible. Furthermore, none of the subproblems is unbounded, hence the solution obtained is feasible.

  Through this paragraph, it will be shown that the solution obtained is locally optimal. The stopping requirement is that $z^{k+1} = f^*(\boldsymbol{e}^k)$. According to Eq. (4.4), $z^{k+1} \geq \beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}^k) = \langle \boldsymbol{p}^{\hat{k}}, \boldsymbol{e}^k \rangle + \alpha^{\hat{k}}, \forall\, \hat{k} \in \{1,2,\dots,k\}$. To guarantee the optimality of the last master problem solved, there must be a plane $\bar{k} \in \{1,2,\dots,k\}$ such that $z^{k+1} = \langle \boldsymbol{p}^{\bar{k}}, \boldsymbol{e}^k \rangle + \alpha^{\bar{k}}$, which indicates that $\langle \boldsymbol{p}^{\bar{k}}, \boldsymbol{e}^k \rangle + \alpha^{\bar{k}} = f^*(\boldsymbol{e}^k)$. Now, assume that the solution obtained is not locally optimal, i.e., assume that the requirement $f^*(\boldsymbol{e}^k) \leq f^*(\boldsymbol{\alpha}), \forall \boldsymbol{\alpha} \in N_\infty(\boldsymbol{e}^k)$ is not satisfied. Then, there is at least one $\boldsymbol{\theta} \in N_\infty(\boldsymbol{e}^k)$ such that $f^*(\boldsymbol{\theta}) < f^*(\boldsymbol{e}^k)$. Having $f^*(\boldsymbol{\theta}) < f^*(\boldsymbol{e}^k)$ is possible thanks to the assumption that solution $f^*(\boldsymbol{e}^k)$ is not unbounded, i.e., $f^*(\boldsymbol{e}^k) > -\infty$ returns finite numerical values. The solution obtained ($\boldsymbol{e}^k$) is feasible, thus its objective function is finite, i.e., $f^*(\boldsymbol{e}^k) < +\infty$, hence $\boldsymbol{\theta} \in D^k$ (see Eq. 4.10) and $\langle \boldsymbol{p}^{\bar{k}}, \boldsymbol{\theta} \rangle + \alpha^{\bar{k}} \leq f^*(\boldsymbol{\theta})$. This means that $\langle \boldsymbol{p}^{\bar{k}}, \boldsymbol{\theta} \rangle + \alpha^{\bar{k}} < f^*(\boldsymbol{e}^k)$. This results in a contradiction with the fact that $\langle \boldsymbol{p}^{\bar{k}}, \boldsymbol{e}^k \rangle + \alpha^{\bar{k}} = f^*(\boldsymbol{e}^k)$, which completes the proof. ■

D.3 Improvements to find a feasible initialization

According to Theorem 4.2, one important requirement for the LD-BD algorithm is that $\boldsymbol{e}^{init}$ returns a feasible solution to the problem. Therefore, an initialization strategy that uses the LD-BD algorithm is proposed in this section. Figure D.2 summarizes the strategy considered to solve an optimization problem when initialized from an infeasible point, i.e., a vector of external variables $\boldsymbol{e}^{init}$ that does not

return a feasible solution. For this general procedure it is being assumed that the user provided initialization of continuous variables ($\boldsymbol{x}^{init}$) does not result in undefined equations, e.g., divisions by zero. The procedure in Figure D.2 accepts any initialization for $\boldsymbol{e}^{init}$, e.g., a vector of ones. Then, it proceeds with initialization phase 1, which finds a combination of external variables that guarantees feasibility for the logical constraints $\Omega(\boldsymbol{Y_R}) = True$ of the problem. Once this solution is found, the algorithm goes to phase 2, where the aim is to attain feasibility for the remaining constraints of the problem, i.e., nonlinear constraints $\boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}, \widehat{\boldsymbol{g}}(\boldsymbol{x}) = \boldsymbol{0}$, $\boldsymbol{h}_{d,q}(\boldsymbol{x}) \leq \boldsymbol{0}$ and $\widehat{\boldsymbol{h}}_{d,q}(\boldsymbol{x}) = \boldsymbol{0}$. A feasible solution is expected after phase 2, and the algorithm can go to the optimization phase.

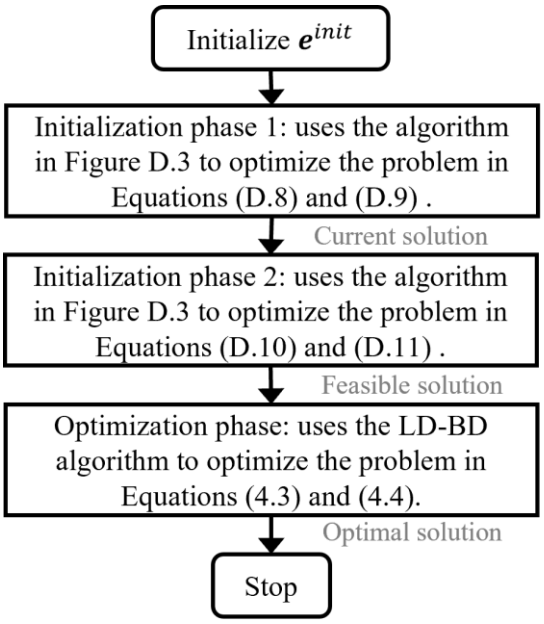


Figure D.2. Steps to optimize a problem when initialized from an infeasible point.

The overall idea of phases 1 and 2 is to attain feasibility by minimizing an infeasibility measure, i.e., I_1 and I_2 for phases 1 and 2 respectively. Some infeasibility measures that are usually considered in the literature are the sum of infeasibilities and the number of infeasibilities[383]. In this work, the sum of infeasibilities is considered for I_1 and I_2. The sum of infeasibilities is defined as the sum over infeasible constraints of the difference between constraint bodies and their upper bounds, e.g., for a constraint of the form $g(x) \leq b$, $g(x)$ is the body and b is the upper bound. Hence, this infeasibility metric is expected to take the value of 0 when infeasibility is minimized.

The LD-BD strategy is used as a basis to perform the minimization of infeasibilities in phases 1 and 2, which means that these optimization problems must be decomposed. The decomposed version of the optimization problems used to minimize infeasibility are shown in Eqs. (D.8)-(D.11). Eqs. (D.8) and (D.9) represent the subproblem and master problem considered in phase 1, while Eqs. (D.10) and (D.11) are the subproblem and master problem in phase 2.

$$Subproblem\ (phase\ 1) = \left\{ f^*(\boldsymbol{e}^k) = I_1\left(\Omega\left(\boldsymbol{Y_R}(\boldsymbol{e}^k)\right)\right) \right. \tag{D.8}$$

$$Master\ problem\ (phase\ 1) = \begin{cases} [z^{k+1}, \boldsymbol{e}^{k+1}] = \arg\min\limits_{z,\boldsymbol{e}} z \\ s.t.\ z \geq \beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}), \forall\, \hat{k} \in \{1,2,\dots,k\} \\ \boldsymbol{e} \in E \subset \mathbb{N}^{n_e}, z \in \mathbb{R} \end{cases} \tag{D.9}$$

$$Subproblem\ (phase\ 2) = \tag{D.10}$$

$$\begin{cases} f^*(\boldsymbol{e}^k) = \min\limits_{\boldsymbol{x}} I_2\big(\boldsymbol{g}(\boldsymbol{x}), \widehat{\boldsymbol{g}}(\boldsymbol{x}), \{[\boldsymbol{h}_{d,q}(\boldsymbol{x}), \widehat{\boldsymbol{h}}_{d,q}(\boldsymbol{x})]: Y_{d,q} = True\}\big) \\ s.t.\ \boldsymbol{x} \in X \subset \mathbb{R}^{n_x}, \Omega\big(\boldsymbol{Y_R}(\boldsymbol{e}^k)\big) = True \end{cases}$$

$$Master\ problem\ (phase\ 2) = \begin{cases} [z^{k+1}, \boldsymbol{e}^{k+1}] = \arg\min\limits_{z,\boldsymbol{e}} z \\ s.t.\ z \geq \beta_{\boldsymbol{e}^{\hat{k}}}(\boldsymbol{e}), \forall\, \hat{k} \in \{1,2,\dots,k\} \\ \boldsymbol{e} \in E \subset \mathbb{N}^{n_e}, z \in \mathbb{R} \end{cases} \tag{D.11}$$

In phase 1, $\boldsymbol{e}^k$ remains fixed in the subproblems, hence Boolean variables $\boldsymbol{Y_R}$ can be fixed accordingly. Therefore, logical constraints $\Omega\left(\boldsymbol{Y_R}(\boldsymbol{e}^k)\right) = True$ remain fixed, and the subproblem in Eq. (D.8) simply requires the evaluation of the infeasibility function I_1. Constraint bodies and upper bounds of logical constraints are computed based on the logical-to-linear interpretation of logical constraints performed automatically by the Feasibility-Based Bound Tightening (FBBT) tool in Pyomo.

In phase 2, logical expressions are moved to the constraints of the subproblem. When vector $\boldsymbol{e}^k$ takes a value that satisfies logical constraints, all logical constraints and disjunctions remain fixed and the infeasibility of constraints that do not depend on disjunctions (i.e., $\boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}$ and $\widehat{\boldsymbol{g}}(\boldsymbol{x}) = \boldsymbol{0}$) and constraints within active disjunctions (i.e., $\{[\boldsymbol{h}_{d,q}(\boldsymbol{x}) \leq \boldsymbol{0}, \widehat{\boldsymbol{h}}_{d,q}(\boldsymbol{x}) = \boldsymbol{0}]: Y_{d,q} = True, \forall d \in D_q, \forall q \in Q\}$) can be minimized using an NLP solver (see Eq. (D.10)).

The solution of optimization problems in phases 1 and 2 is performed using the algorithm presented in Figure D.3. This algorithm starts by executing the LD-BD method in step 1, which is expected to return a solution with an optimal infeasibility measure. Nevertheless, the LD-BD strategy only guarantees local optimality, thus the infeasibility measure obtained after step 1 may not be below the required feasibility tolerance ϵ_f. For this reason, further steps (steps 2-5 in Figure D.3) were added to the procedure to increase the performance of the algorithm at attaining a feasible solution. The algorithm in Figure D.3 keeps iterating between the LD-BD method (step 1) and steps 2-5 until a feasible solution is found. If a solution is not feasible according to step 2, then the algorithm goes to step 3 where an infeasibility measure of $+\infty$ is assigned to the current trial value $\boldsymbol{e}^k$. This assignment avoids convergence to this infeasible solution in subsequent iterations of the algorithm. After that, the algorithm verifies the existence of other solutions with a finite infeasibility measure (i.e., $f^*(\boldsymbol{e}) < +\infty$) within D^k (see step 4). If such solution exists, $\boldsymbol{e}^{init}$ is updated with the best solution within D^k, and D^{init} is updated with D^k in step 5, and the algorithm goes to step 1 to execute the LD-BD algorithm. Since an objective function of $+\infty$ was assigned to $f^*(\boldsymbol{e}^k)$, the LD-BD algorithm is expected to return a different locally optimal solution in a new iteration. This iterative procedure continues until a feasible solution is found or until a value of $+\infty$ is assigned to the infeasibility measure everywhere within D^k. In the second scenario, the algorithm has failed to converge to a feasible solution, thus reinitializing the procedure in Figure D.2 from another point $\boldsymbol{e}^{init}$ is recommended.

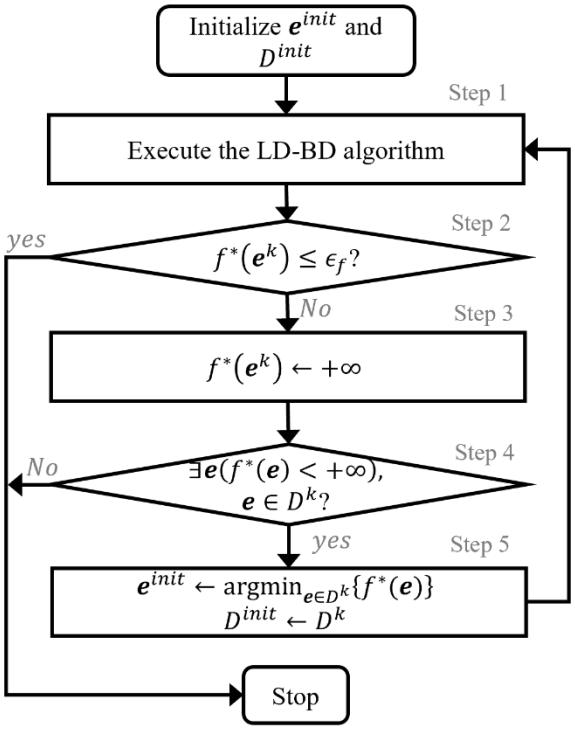


Figure D.3. Algorithm to solve the optimization problems in initialization phases 1 and 2.

D.4 Case study 2: Convexity analysis

An analysis to determine if the first case study is integrally convex is performed in this section. The effective domain of external variables ($\text{dom}_{\mathbb{Z}}(f^*)$) is analyzed first because if this domain is not integrally convex, then objective function $f^*(\boldsymbol{e})$ is not integrally convex. Figure D.4 shows the convex hull of $\text{dom}_{\mathbb{Z}}(f^*)$ (dotted blue line) and compares it with the union of local convex hulls $\text{co}(\text{dom}_{\mathbb{Z}}(f^*) \cap N(\boldsymbol{y}))$ over $\boldsymbol{y} \in \mathbb{R}^2$ (continuous orange line). This test shows that there are points $\boldsymbol{y} \in \mathbb{R}^2$ that belong to $\text{co}\big(\text{dom}_{\mathbb{Z}}(f^*)\big)$, but these points do not belong to $\text{co}(\text{dom}_{\mathbb{Z}}(f^*) \cap N(\boldsymbol{y}))$ (black area in Figure D.4). Hence, $\text{dom}_{\mathbb{Z}}(f^*)$ is not integrally convex. Therefore, $f^*(\boldsymbol{e})$ is not integrally convex, indicating that the problem is not integrally convex.

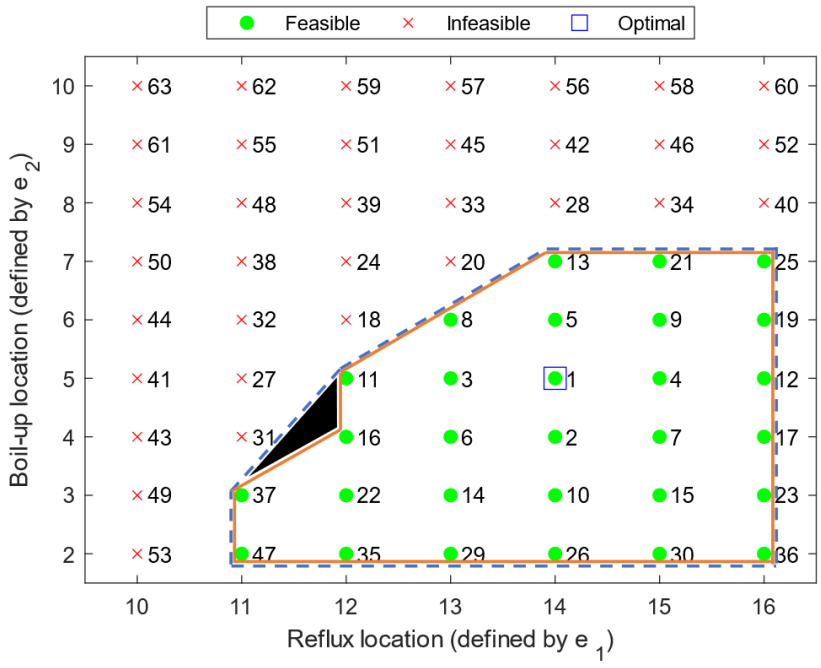


Figure D.4. Representation of 1) $\mathrm{co}\left(\mathrm{dom}_{\mathbb{Z}}(f^*)\right)$ as a dotted blue line, and 2) $\bigcup_{\boldsymbol{y}\in\mathbb{R}^2} \mathrm{co}(\mathrm{dom}_{\mathbb{Z}}(f^*) \cap N(\boldsymbol{y}))$ as a continuous orange line. The black area represents the source of non-convexity.

D.5 Case study 2: Performance of MINLP solvers

This section compares the performance of LD-BD and LD-SDA with local MINLP solvers, using the distillation design problem as a case study. The performance of three MINLP solvers, i.e., SBB, DICOPT and Alpha-ECP, was obtained by solving the optimization problem from different initial guesses of external variables (see details in chapter 4). The distillation GDP problem was transformed into a MINLP using the automatic Big-M transformation within Pyomo. Solver KNITRO was used as the NLP solver and a time limit of 10^3 seconds was stablished for all the optimization trials. The results are presented in Figures D.5A and D.5B.

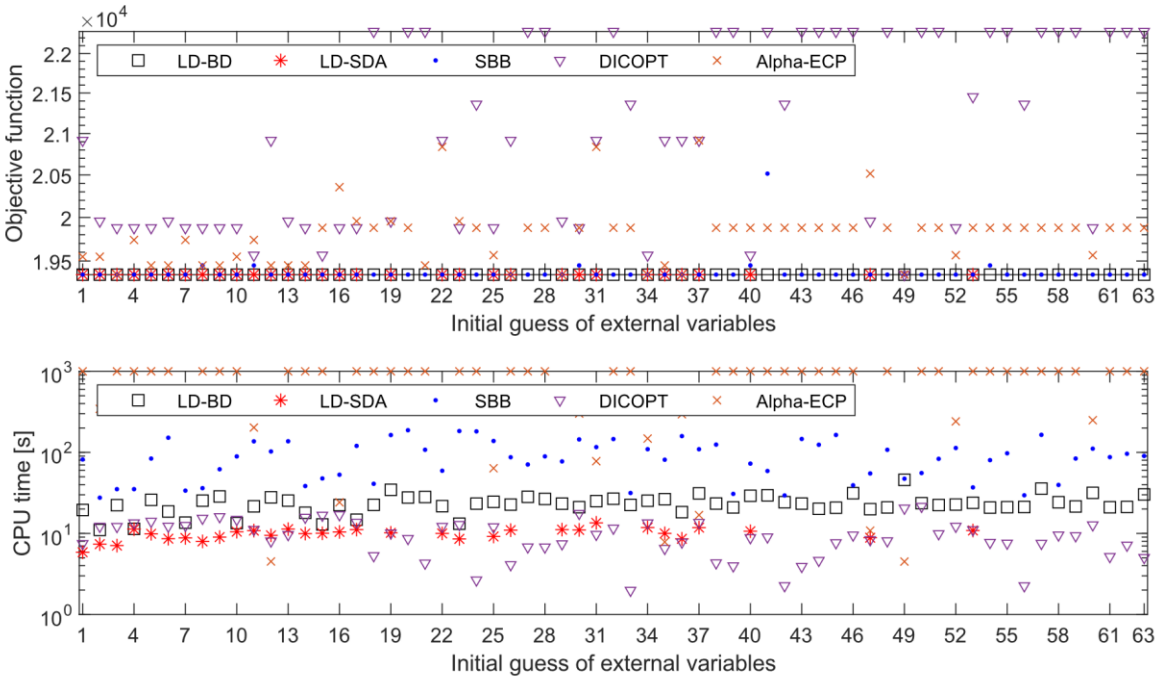


Figure D.5. Performance of MINLP solvers

According to Figure D.5, DICOPT shows the best performance in terms of computational time (it required 9.8 seconds on average), however only 1 out of 63 optimization trials converged to the global optimum. Solver SBB demonstrated to be the best MINLP solver for this problem, converging to the optimal solution in 90.4% of the optimization trials, but requiring 90.9 seconds on average, i.e., almost four times slower than LD-BD. Solver Alpha-ECP shows the worst performance in terms of average computational time (795.3 seconds) and it allowed to find the local optimum in 9.5% of the trials. Hence, none of the local MINLP solvers tested showed a better performance than LD-BD, in terms of their ability to find the global optimum. Note that global MINLP solvers such as BARON and LINDOGLOBAL were also tested for this case study, however, these erroneously claimed infeasibility for every initial guess of external variables.

D.6 Case study 3: Problem statement and reformulation

The aim of this case study is to minimize the total reactor volume of a series of CSTR reactors (Eq. (D.12)), while guaranteeing the desired product specification (Eq. (D.13)) and reactors of equal volume (Eq. (D.14)). Constraints in Eq. (D.15) represent continuity equations and mole balances for components A and B.

$$\min_{\boldsymbol{x},\boldsymbol{Y},\boldsymbol{YF},\boldsymbol{YR}} f = \sum_{n\in\{1,2,\dots,5\}} c_n \tag{D.12}$$

$$s.t.$$

$$C_B^{out} = 0.95 \tag{D.13}$$

$$V_n = V_{n-1}, \forall n \in \{2,3,4,5\} \tag{D.14}$$

$$\widehat{\boldsymbol{g}}(\boldsymbol{x}) = \mathbf{0} \tag{D.15}$$

$$\begin{bmatrix} Y_n \\ \widehat{\boldsymbol{h}}_1(\boldsymbol{x}) = \mathbf{0} \\ c_n = V_n \end{bmatrix} \vee \begin{bmatrix} \neg Y_n \\ \widehat{\boldsymbol{h}}_2(\boldsymbol{x}) = \mathbf{0} \\ c_n = 0 \end{bmatrix}, \forall n \in \{1,2,\dots,5\} \tag{D.16}$$

$$\begin{bmatrix} YR_n \\ \widehat{\boldsymbol{h}}_3(\boldsymbol{x}) = \mathbf{0} \end{bmatrix} \vee \begin{bmatrix} \neg YR_n \\ \widehat{\boldsymbol{h}}_4(\boldsymbol{x}) = \mathbf{0} \end{bmatrix}, \forall n \in \{1,2,\dots,5\} \tag{D.17}$$

$$\Omega(\boldsymbol{Y},\boldsymbol{YF},\boldsymbol{YR}) = True \tag{D.18}$$

This problem has a similar structure than the second case study presented in chapter 4, with a Boolean variable Y_n that activates either reaction constraints ($\widehat{\boldsymbol{h}}_1(\boldsymbol{x}) = \mathbf{0}$) or bypass constraints ($\widehat{\boldsymbol{h}}_2(\boldsymbol{x}) = \mathbf{0}$) to indicate the existence/absence of a CSTR in the series (see Eq. (D.16)). Eq. (D.16) also activates/deactivates reactor volumes (V_n), depending on the value of Y_n. Similarly, Eq. (D.17) indicates if the recycle flow is connected ($\widehat{\boldsymbol{h}}_3(\boldsymbol{x}) = \mathbf{0}$) or excluded ($\widehat{\boldsymbol{h}}_3(\boldsymbol{x}) = \mathbf{0}$) from unit n. Auxiliary Boolean variables YF_n and YR_n are used to represent the feed and recycle location in the reactor series, which allows to reformulate the problem with external variables e_1 and e_2, as depicted in Figures D.6A and D.6B. Eq. (D.18) contains logical constraints, i.e., the unit must be a CSTR to include a recycle at n, there is only one recycle stream YR_n, there is only one feed $YF_n$to the reactor network, and there is an equivalence relationship that links YF_n with Y_n, i.e., $\wedge_{j\in\{1,2,\dots,n\}} \neg YF_j \vee YF_n \Leftrightarrow Y_n$.

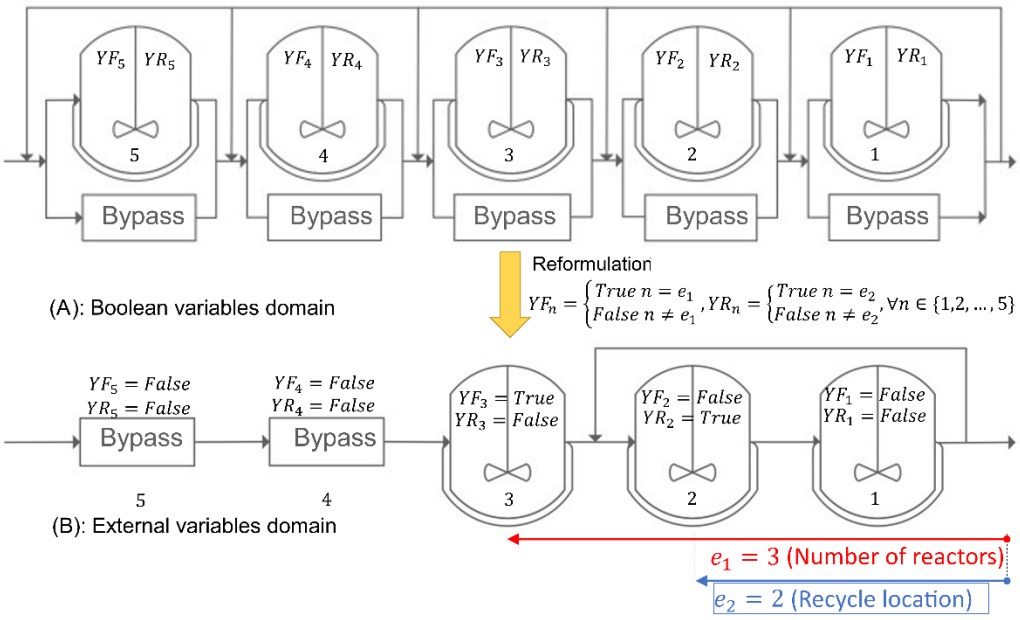


Figure D.6. Representation of the discrete decisions of the reactor network problem in the Boolean variables domain (A) and the external variables domain (B).

Appendix E
Supplementary Material for Chapter 5

E.1 Supplementary algorithms required to execute DSDA-VB

E1.1 Supplementary information for the proposed hybrid algorithm

Figure E.1 provides more details related to the execution of the proposed hybrid algorithm (Figure 5.2). The figure specifies when data exchange occurs, who writes/reads data, how variables and constraints are managed by each algorithm (if applicable), and how the different steps interact. Different SMs that can be used include evolution strategies, genetic algorithms, simulated annealing, among others. Different simulation packages that can be coupled to the proposed optimization approach include (but not limited to) Aspen Plus, UniSim, Pro/II, gProms, etc. The key advantages unique to each algorithm involved can be summarized as follows:

- Steps performed in processor 1 (Steps S1-S5): these iterative steps decide when the SM is executed or stopped. Each time the SM is executed, it uses a new set of bounds for the DOFs involved. These updated search bounds come from the deterministic steps D1-D7. One attractive feature of each execution of the SM is that, in contrast to the DSDA, it aims to perform a global search every time it is executed, e.g., it performs an exploration of discrete decisions over larger neighborhoods than those considered by the DSDA. It is also responsible for identifying the first set of feasible solutions to the problem.
- Steps performed in processor 2 (Steps D1-D7): These iterative steps refine the best solutions obtained by the SM, using the DSDA as a local optimization algorithm. A VB strategy is also implemented to refine the bounds of continuous variables. The key strength of the DSDA relies on its ability to optimize both discrete and continuous DOF when interacting with a simulation software, allowing it to identify local optima. The key roles of the VB strategy are to facilitate the convergence of NLP subproblems, and refine the search bounds considered by the SM.

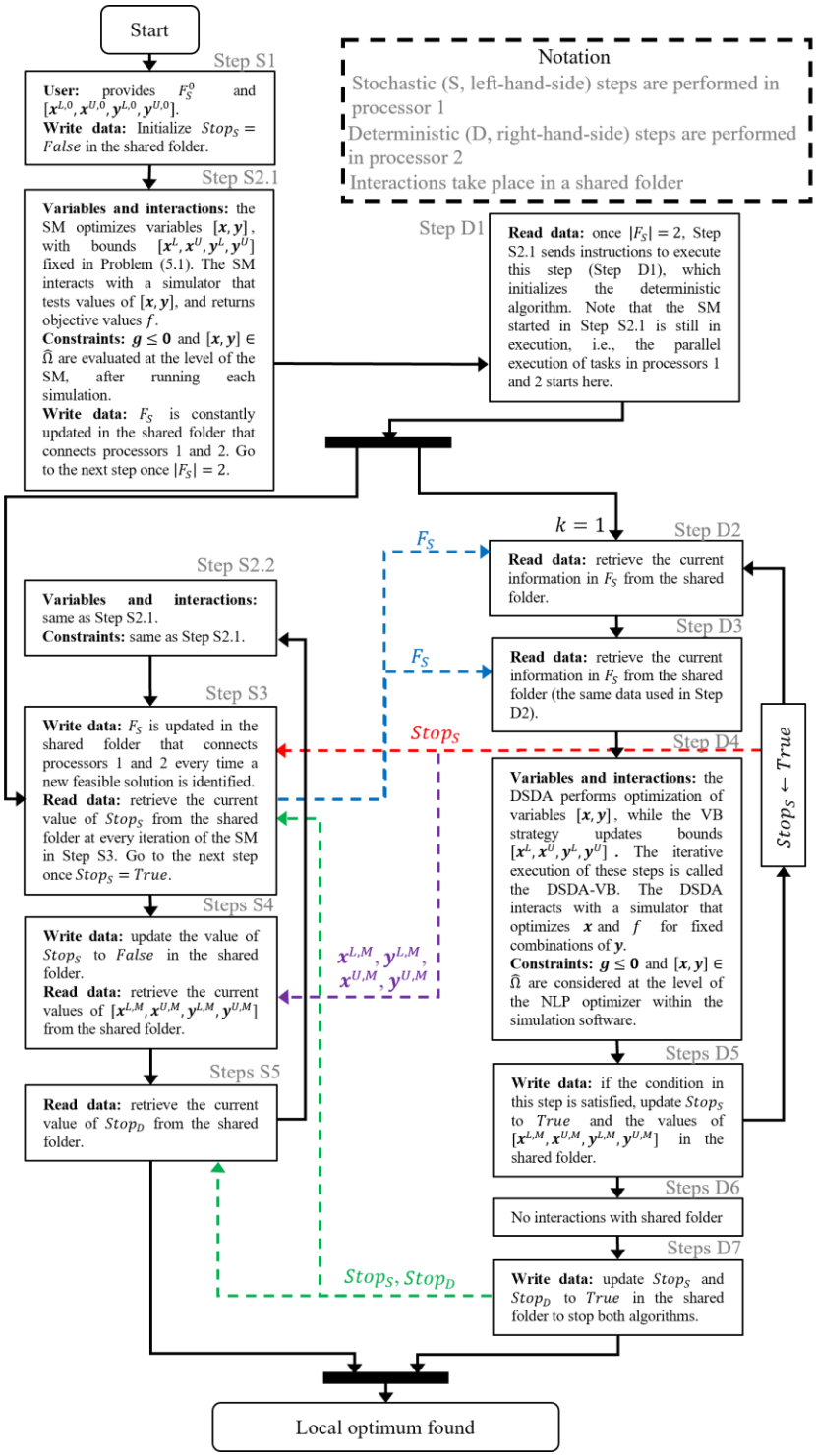


Figure E.1. Additional details of the steps in the proposed hybrid algorithm (Figure 5.2). A shared folder allows processors 1 and 2 to communicate with each other.

E1.2 The DSDA-VB algorithm

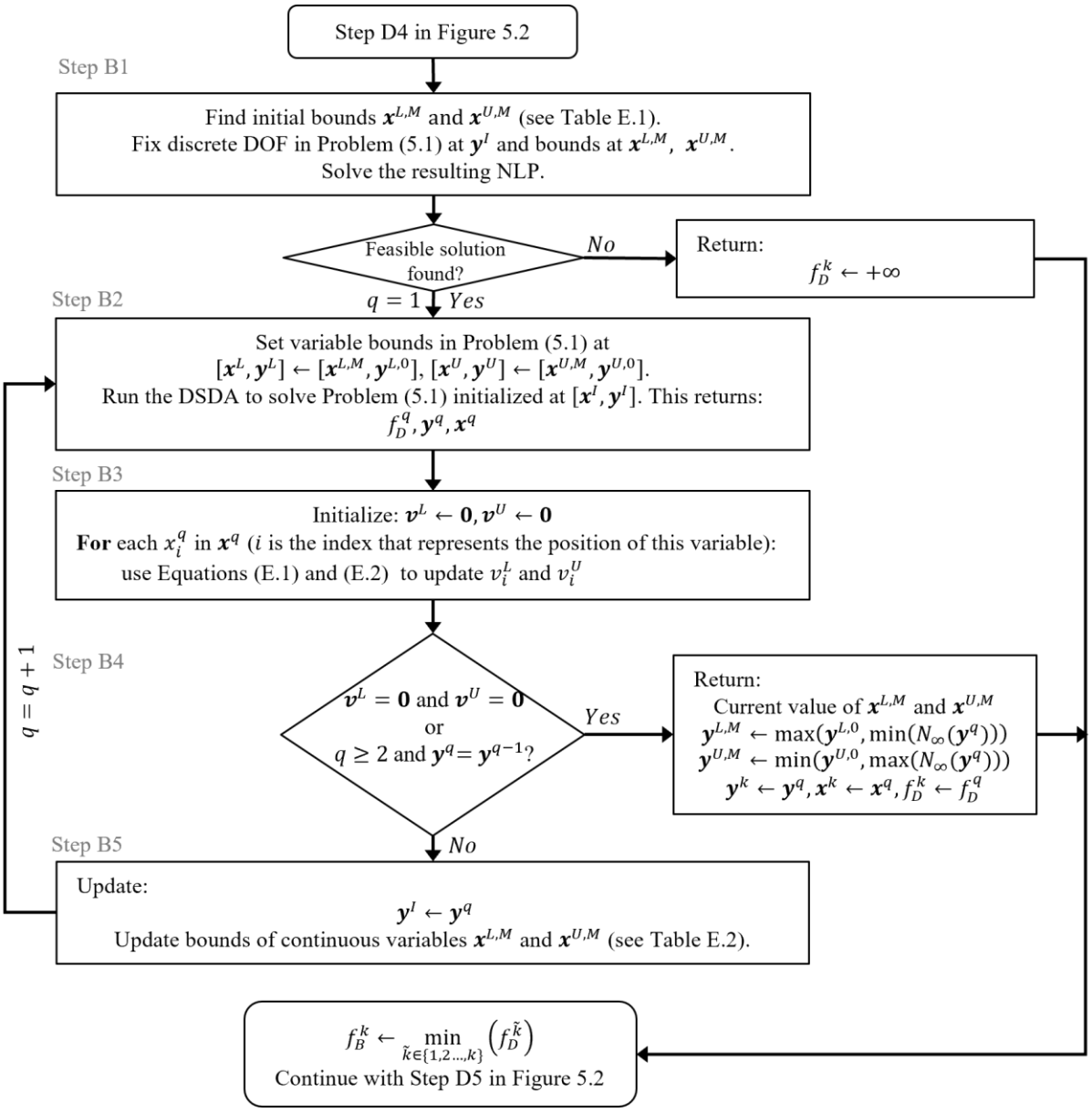


Figure E.2. Step D4 of Figure 5.2: The Discrete-Steepest Descent Algorithm with Variable Bounding (DSDA-VB). Superscript q is the iteration counter used within DSDA-VB.

In *Step B1* (Figure E.2), the DSDA-VB algorithm starts by finding variable bounds $\boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^{U,M}$ that result in a feasible solution when DSDA is executed. The feasibility of the DSDA is ensured when its first NLP subproblem defined by fixing $\boldsymbol{y}$ at $\boldsymbol{y}^I$ converges to a feasible solution. The NLP solver

initially has access to the feasible initialization point $\boldsymbol{x}^I$ and bound estimates $\boldsymbol{x}^{L,S}$ and $\boldsymbol{x}^{U,S}$ provided by the stochastic algorithm (see *Step D3* in Figure 5.1). If this initial information results in convergence of the first NLP solved, then $\boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^{U,M}$ can be initialized at $\boldsymbol{x}^{L,S}$ and $\boldsymbol{x}^{U,S}$, respectively in *Step B1* of DSDA-VB. Otherwise, point $\boldsymbol{x}^I$ and bound estimates $\boldsymbol{x}^{L,S}$ and $\boldsymbol{x}^{U,S}$ can be used as initial guess to perform a search that results in convergence of the first NLP subproblem. In this work, variable bounds are sequentially tightened from $\boldsymbol{x}^{L,S}$ and $\boldsymbol{x}^{U,S}$ towards $\boldsymbol{x}^I$ until NLP convergence is attained (see Table E.1). If the NLP solver does not recognize point $[\boldsymbol{x}^I, \boldsymbol{y}^I]$ as feasible, DSDA-VB returns $f_D^k = +\infty$, indicating that no feasible solution was attained. Otherwise, the iterative execution of DSDA-VB begins with $q = 1$. In *Step B2*, the DSDA is executed considering the bounds obtained from the previous step. The execution of DSDA returns solution $[\boldsymbol{x}^q, \boldsymbol{y}^q]$, which is analyzed in *Step B3* to decide if there is any variable x_i^q in $\boldsymbol{x}^q$ at an algorithmic bound $x_i^{L,M}$ or $x_i^{U,M}$ using Eqs. (E.1) and (E.2), where ϵ_2 is a small tolerance. Vectors $\boldsymbol{v}^L$ and $\boldsymbol{v}^U$ with elements v_i^L and v_i^U are used in Eqs. (E.1) and (E.2) to keep track of variables at their lower (L) or upper (U) algorithmic bounds, where a value of 1 is assigned to v_i^L or v_i^U if variable x_i^q is at its lower or upper algorithmic bound and this algorithmic bound ($x_i^{L,M}$ or $x_i^{U,M}$) is different to the user-defined bound ($x_i^{L,0}$ or $x_i^{U,0}$). A value of 0 is assigned to v_i^L or v_i^U when x_i^q is not at its lower ($x_i^{L,M}$) or upper ($x_i^{U,M}$) algorithmic bound, or when a variable x_i^q is equal to the user-defined upper ($x_i^{U,0}$) or lower bound ($x_i^{L,0}$). This assignment guarantees that user-defined bounds are always satisfied and left unmodified by the DSDA-VB, which accepts a variable at a user-defined bound in a locally optimal solution.

$$v_i^L = \begin{cases} 1, \left|\frac{x_i^q - x_i^{L,M}}{x_i^{U,M} - x_i^{L,M}}\right| \leq \epsilon_2 \text{ and } \left|\frac{x_i^{L,M} - x_i^{L,0}}{x_i^{U,0} - x_i^{L,0}}\right| \geq \epsilon_2, \forall i \in \{1,2,\dots,n_x\} \\ 0, otherwise \end{cases} \tag{E.1}$$

$$v_i^U = \begin{cases} 1, \left|\frac{x_i^q - x_i^{U,M}}{x_i^{U,M} - x_i^{L,M}}\right| \leq \epsilon_2 \text{ and } \left|\frac{x_i^{U,M} - x_i^{U,0}}{x_i^{U,0} - x_i^{L,0}}\right| \geq \epsilon_2 \ , \ \forall i \in \{1,2,\dots,n_x\} \\ 0, otherwise \end{cases} \tag{E.2}$$

The stopping criteria of the DSDA-VB algorithm are tested in *Step B4*. According to this step, if there no variables at their algorithmic bounds (i.e., $\boldsymbol{v}^L = \boldsymbol{0}$ and $\boldsymbol{v}^U = \boldsymbol{0}$) or discrete decisions have not changed for 2 consecutive iterations (i.e., $q \geq 2$ and $\boldsymbol{y}^q = \boldsymbol{y}^{q-1}$), then DSDA-VB stops. When DSDA-VB stops, it returns the optimal objective f_D^q and solution $[\boldsymbol{x}^q, \boldsymbol{y}^q]$, as well as updated continuous ($\boldsymbol{x}^{L,M}$, $\boldsymbol{x}^{U,M}$) and discrete ($\boldsymbol{y}^{L,M}, \boldsymbol{y}^{U,M}$) bounds. Otherwise, the algorithm continues to the next iteration, e.g.,

when at least one algorithmic bound is restraining a variable. Then, *Step B5* is executed to relax the bounds of continuous variables, which may improve the DSDA-VB solution in subsequent iterations. In this work, bounds of variables that are at their bounds are relaxed based on the solution of a series of NLP subproblems that test different bounds of continuous DOF, with discrete DOF fixed at $\boldsymbol{y} = \boldsymbol{y}^I$ (see Table E.2). For instance, consider variable x_i, and assume that its current value x_i^q is at its lower algorithmic bound $x^{L,M}$. Then, $x^{L,M}$ is iteratively relaxed from its current value and a new NLP returning an updated x_i^q is solved until either x_i^q is not changing significatively between iterations, or until further modifying $x^{L,M}$ within a vicinity of its current value results in NLP solver failure. The strategy considered to update $x^{L,M}$ is a bisection strategy performed over $[x^{L,0}, x^{L,M}]$, based on a function that takes values of 1 when the NLP converges and -1 when the NLP diverges, i.e., the bisection strategy is seeking the most relaxed value of $x^{L,M}$ that does not result in NLP solver failure. The above procedure is repeated iteratively for every continuous variable that is at an algorithmic bound. This relaxation procedure continuous until either variables are no longer at their bounds, or there are still some variables at their bounds, but further modifying $\boldsymbol{x}^{L,M}$ or $\boldsymbol{x}^{U,M}$ would result in NLP solver failure, e.g., x_i^q is at its bound ($x_i^q = x^{L,M}$) and the current value of $x^{L,M}$ is keeping x_i^q away from a singularity in a nonlinear function. Figures E.3 and E.4 show an illustrative example of this relaxation procedure performed in two iterations for a problem involving two continuous DOF. The updated bounds obtained after *Step B5* are sent back to the DSDA, and a new iteration of the DSDA-VB algorithm begins.

E1.3 Supplementary information for step B1 of the DSDA-VB algorithm

The algorithm in Table E.1 shows how variable bounds are shrunk from $\boldsymbol{x}^{L,S}$ and $\boldsymbol{x}^{U,S}$ to $\boldsymbol{x}^I$ until feasibility of the first NLP subproblem solved by the DSDA-VB algorithm is attained. The tolerance considering in this work when executing this algorithm is $\epsilon_1 = 1 * 10^{-6}$.

Table E.1. Iterative strategy to initialize continuous variables bounds $\boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^{U,M}$ from $\boldsymbol{x}^{L,S}$ and $\boldsymbol{x}^{U,S}$

Step 1. Initialize

Initialize $\boldsymbol{x}^{L,M} \leftarrow \boldsymbol{x}^{L,S}$ and $\boldsymbol{x}^{U,M} \leftarrow \boldsymbol{x}^{U,S}$. Then, go to Step 2.

Step 2. Solve NLP subproblem with the current bounds

Solve the NLP problem that results from fixing $\boldsymbol{y}$ at $\boldsymbol{y}^I$, $\boldsymbol{x}^L$ at $\boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^U$ at $\boldsymbol{x}^{U,M}$ in Problem (5.1). Then, go to Step 3.

Step 3. Verification: Feasible solution found?

If a feasible solution was found in Step 2, then stop: $\boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^{U,M}$ can be used to initialize DSDA-VB.

Else if $\boldsymbol{x}^{L,M} = \boldsymbol{x}^I(1-\epsilon_1)$ and $\boldsymbol{x}^{U,M} = \boldsymbol{x}^I(1+\epsilon_1)$, then stop: suitable bounds for $\boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^{U,M}$ could not be found to initialize DSDA-VB.

Else, continue with step 4.

Step 4. Update

Update continuous variables bounds as shown below, where $n \in \mathbb{Z}_{>0}$ and ϵ_1 is a tolerance that satisfies $\boldsymbol{x}^I(1+\epsilon_1) < \boldsymbol{x}^{U,S}$ and $\boldsymbol{x}^I(1-\epsilon_1) > \boldsymbol{x}^{L,S}$. Then, go to Step 2.

$$\boldsymbol{x}^{L,M} \leftarrow \min\left(\boldsymbol{x}^I(1-\epsilon_1), \boldsymbol{x}^{L,M} + \frac{\boldsymbol{x}^I(1-\epsilon_1)-\boldsymbol{x}^{L,S}}{n}\right), \boldsymbol{x}^{U,M} \leftarrow \max\left(\boldsymbol{x}^I(1+\epsilon_1), \boldsymbol{x}^{U,M} - \frac{\boldsymbol{x}^{U,S}-\boldsymbol{x}^I(1+\epsilon_1)}{n}\right)$$

E1.4 Supplementary information for step B5 of the DSDA-VB algorithm

The algorithm in Table E.2 shows the detailed steps considered in this work to perform the relaxation of bounds in Step B5 of the DSDA-VB method. The values for tolerances ϵ_2, ϵ_3 and ϵ_4 considered in this work is $1 * 10^{-2}$.

Table E. 2. Bound relaxation using a bisection strategy to update upper and lower bounds of continuous variables.

Initialize

Initialize $\boldsymbol{v}^{L,0} \leftarrow \boldsymbol{v}^{L}, \boldsymbol{v}^{U,0} \leftarrow \boldsymbol{v}^{U}$

For $iter = 1$ **to** max_{iter}:

For each continuous variable i such that $v_i^{L,iter-1} = 1$ or $v_i^{U,iter-1} = 1^*$:

Step 1. Define NLP(c_i) **and** $f_b(c_i)$**, using** $\boldsymbol{x}^{L,M}$, $\boldsymbol{x}^{U,M}$ **and** $\boldsymbol{y}^{I}$ **as input**

Take c_i as the lower/(upper) bound of variable i that will change iteratively within this algorithm. Define NLP(c_i), by using the following configuration in Problem (5.1):

- Discrete variables fixed at $\boldsymbol{y}^{I}$.
- Lower/(upper) bounds of variable i set at $x_i^L = c_i$ /($x_i^U = c_i$).
- Variable bounds $x_j^L = x_j^{L,M}$ and $x_j^U = x_j^{U,M}$ for $j \neq i$.

NLP(c_i) returns optimal objective function f_i^* and continuous variables x_i^*. Then, define function $f_b(c_i)$: $\mathbb{R} \mapsto \{-1,1\}$, where $f_b(c_i)$ takes the value of 1 when NLP(c_i) converges to a feasible solution (i.e., $f_i^* < +\infty$) and -1 otherwise (i.e., $f_i^* = +\infty$). Then, go to Step 2.

Step 2. Initialization using f_D^q **and** $\boldsymbol{x}^q$ **as input**

Initialize $a_i \leftarrow x_i^{L,0}/(a_i \leftarrow x_i^{U,M})$, $b_i \leftarrow x_i^{L,M}/(b_i \leftarrow x_i^{U,0})$, $f_{i,previous}^* \leftarrow f_D^q$ and $x_{i,previous}^* \leftarrow x_i^q$. Then, go to Step 3.

Step 3. Update the current bound estimate

Let $c_i \leftarrow \frac{a_i+b_i}{2}$. Obtain f_i^* and x_i^* from NLP(c_i), update $f_{i,current}^* \leftarrow f_i^*$ and $x_{i,current}^* \leftarrow x_i^*$, and compute $f_b(c_i)$. Then, go to step 4.

Step 4. Verification

If $f_{i,current}^* < +\infty, f_{i,previous}^* < +\infty$, and $\frac{\left|x_{i,current}^* - x_{i,previous}^*\right|}{1+|x_{i,current}^*|} \leq \epsilon_3$, then stop: the updated lower/(upper) bound of variable i is $x_i^{L,M} = c_i/(x_i^{L,U} = c_i)$. Else if $\frac{|b_i-a_i|}{1+|b_i|} \leq \epsilon_4$,

then stop: If $f_b(a_i) = 1$, then the updated lower/(upper) bound of variable i is $x_i^{L,M} = a_i/(x_i^{L,U} = a_i)$; else $x_i^{L,M} = b_i/(x_i^{L,U} = b_i)$. Otherwise, continue with step 5.

Step 5. Update

If $f_b(a_i) = f_b(b_i) = 1$, update $b_i \leftarrow c_i/(a_i \leftarrow c_i)$. Ese if $f_b(c_i)f_b(a_i) > 0$, update $a_i \leftarrow c_i$, else $b_i \leftarrow c_i$. Also, update $f^*_{i,previous} \leftarrow f^*_{i,current}$ and $x^*_{i,previous} \leftarrow x^*_{i,current}$. Then, go to Step 3.

Update $\left[\boldsymbol{v}^{L,iter}, \boldsymbol{v}^{U,iter}\right]$

Let $\boldsymbol{x}^{iter}$ be the solution of the NLP with $\boldsymbol{y} = \boldsymbol{y}^I$, $\boldsymbol{x}^L = \boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^U = \boldsymbol{x}^{U,M}$ in Problem (5.1).

For each $x_i^{iter}\ in\ \boldsymbol{x}^{iter}$ (i represents the position of variable x_i^{iter} in $\boldsymbol{x}^{iter}$):

$$v_i^{L,iter} = \begin{cases} 1, \left|\frac{x_i^{iter} - x_i^{L,M}}{x_i^{U,M} - x_i^{L,M}}\right| \leq \epsilon_2 \text{ and } \left|\frac{x_i^{L,M} - x_i^{L,0}}{x_i^{U,0} - x_i^{L,0}}\right| \geq \epsilon_2 \\ 0, otherwise \end{cases}$$

$$v_i^{U,iter} = \begin{cases} 1, \left|\frac{x_i^{iter} - x_i^{U,M}}{x_i^{U,M} - x_i^{L,M}}\right| \leq \epsilon_2 \text{ and } \left|\frac{x_i^{U,M} - x_i^{U,0}}{x_i^{U,0} - x_i^{L,0}}\right| \geq \epsilon_2 \\ 0, otherwise \end{cases}$$

If $\boldsymbol{v}^{L,iter} = \boldsymbol{0}$ and $\boldsymbol{v}^{U,iter} = \boldsymbol{0}$:

Stop: there are no variables at their bounds.

Return the current value of $\boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^{U,M}$

Else if $\left[\boldsymbol{v}^{L,iter}, \boldsymbol{v}^{U,iter}\right] = \left[\boldsymbol{v}^{L,iter-1}, \boldsymbol{v}^{U,iter-1}\right]$:

Stop: the same variables remained at their bounds for two consecutive iterations.

Return the current value of $\boldsymbol{x}^{L,M}$ and $\boldsymbol{x}^{U,M}$

*If $v_i^{L,iter-1}$, the inner bisection algorithm (Steps 1-5) is used to update the lower bound of variable i. Conversely, if $v_i^{U,iter-1}$, this algorithm is used to update the upper bound of variable i . Notation "lower/(upper)" is used to make this distinction through the algorithm.

Figures E.3 and E.4 illustrate two iterations of the relaxation procedure shown in Table E.2 for a problem involving two continuous DOF. The algorithm in Table E.2 has two nested for-loops. The first

loop represents the iterations of the algorithm (e.g., Figure E.3 illustrates iteration 1), whereas the second loop iterates over the continuous DOF in the formulation (e.g., Figure E.3A illustrates the relaxation of the first continuous DOF, whereas Figure E.3B shows the relaxation of the second continuous DOF). The grey shaded areas in Figures E.3 and E.4 represent the feasible region whereas the yellow lines indicate the level curves of the objective function. The dashed lines represent the current algorithmic bounds, and the red point represents the current NLP solution. In the first iteration, variables x_1^q and x_2^q are at their algorithmic bounds $x_1^{U,M}$ and $x_2^{L,M}$ (Figure E.3A); the bisection strategy is then applied to both $x_1^{U,M}$ and $x_2^{L,M}$ as illustrated in Figures E.3A and E.3B, respectively. In the second iteration, the optimized value of x_1^q is $x_1^{U,M}$; thus, the bisection strategy is re-applied to this variable (see Figure E.4A). At the end of the second iteration, the NLP becomes infeasible if $x_1^{U,M}$ is further relaxed, e.g., there is a singularity; similarly, x_2^q is not at any bound. This indicates that the relaxation procedure can be terminated since both variables (x_1^q and x_2^q) are no longer at their algorithmic bounds, as shown in Figure E.4B.

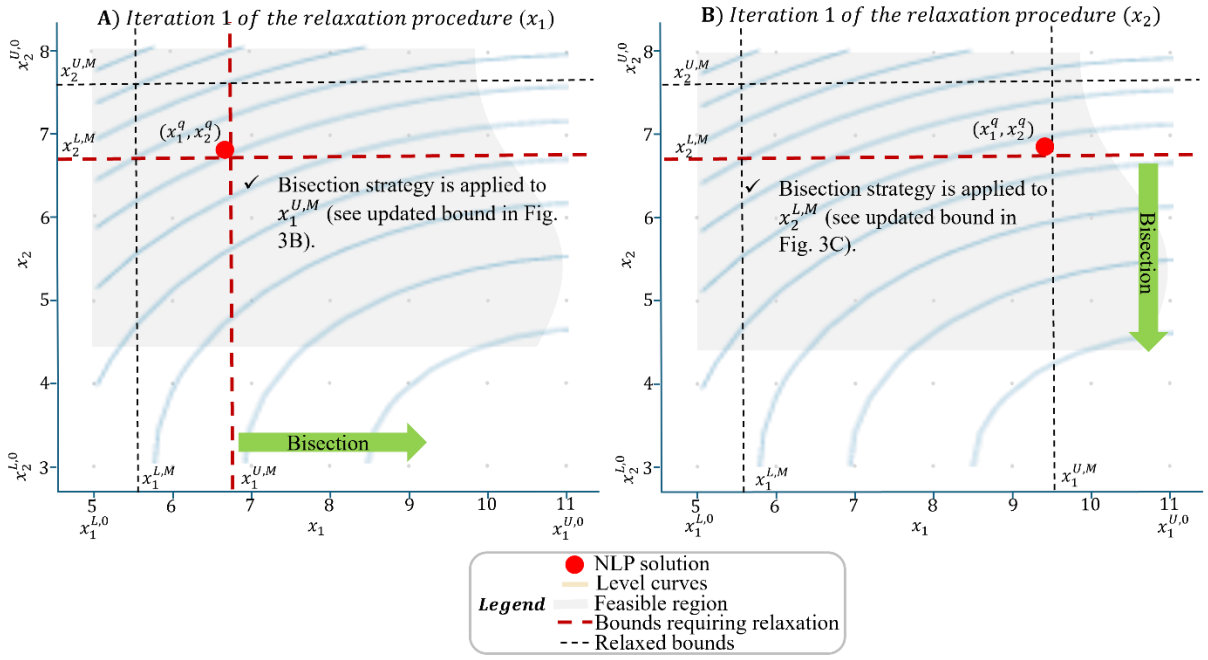


Figure E.3. Illustrative example of the strategy of the algorithm in Table E.2 (i.e., Step B5 in Figure 5.2). A) iteration 1, bound 1, B) iteration 1, bound 2

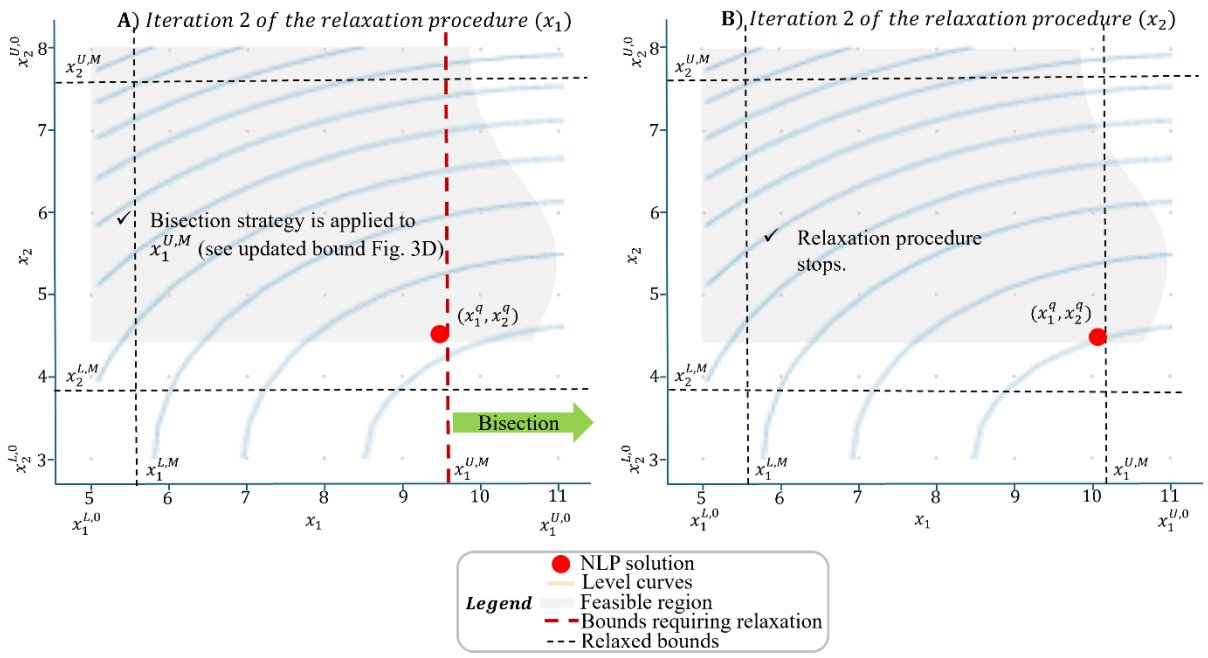


Figure E.4. Illustrative example of the strategy of the algorithm in Table E.2 (i.e., Step B5 in Figure 2). A) iteration 2, bound 1, B) iteration 2, bound 2

E.2 Proof of theorems

Theorem 5.1. *Assume that: 1) the first run of the SM in Step S1 (iteration $k = 0$) can identify at least two different feasible solutions which are added to F_S; 2) the DSDA-VB in Step D4 can eventually find at least one locally optimal objective function (f_D^k) using feasible information in F_S, and 3) the objective function in optimization problem (5.1) is bounded, i.e., the objective function cannot be improved indefinitely without violating the constraints, and the SM in Steps S2.2 and S3 eventually converges towards the global optimum, subject to the current selection of bounds ($\boldsymbol{x}^{L,M}, \boldsymbol{x}^{U,M}, \boldsymbol{y}^{L,M}, \boldsymbol{y}^{U,M}$). Then, the proposed hybrid stochastic-deterministic algorithm will eventually terminate in a local optimum for any user-defined ϵ-tolerance greater or equal to* 0.

Proof of Theorem 5.1. According to assumption 1, the deterministic algorithm does not have to wait indefinitely for the SM to find two feasible solutions in *Step S2.1*, and once two feasible solutions are detected it can proceed with *Steps D1-D7*. Also, having at least one feasible solution in F_S means that f_S^k will have a finite value for every iteration $k = 1,2,\dots,n_k$. According to assumption 2, DSDA-VB detects one local optimum with objective function f_D^k at some iteration $k = k'$, thus $f_B^k = \min_{\tilde{k}\in\{1,\dots,k\}} \left(f_D^{\tilde{k}}\right)$

will have a finite value for any iteration $k \geq k'$. Based on these two assumptions, the difference $f_B^{k-1} - f_B^k$ (*Step D5*) and the gap $\epsilon_k = (f_S^k - f_B^k)/|f_B^k|$ (*Step D6*) can be calculated for $k \geq k'$.

According to assumption 3, the optimization problem is not unbounded, i.e., there exists a constant ϑ such that $f_B^k \geq \vartheta$ for every iteration k. Consequently, sequence $\{f_B^k\}_{k=k'}^{\infty}$ is non-increasing ($f_B^{k-1} \geq f_B^k$) and bounded; hence, it converges to f_B^c. Since the sequence converges, there exists an iteration k'' such that $|f_B^{k-1} - f_B^c| < \delta''$, for $k - 1 > k''$ and any $\delta'' > 0$. Assume $k - 1 > k''$ and consider term $|f_B^{k-1} - f_B^k|$, which can be rewritten as $|f_B^{k-1} - f_B^c + f_B^c - f_B^k| \leq |f_B^{k-1} - f_B^c| + |f_B^k - f_B^c| < 2\delta''$. Take $2\delta'' = \delta$, then $|f_B^{k-1} - f_B^k| = f_B^{k-1} - f_B^k < \delta$ for $k - 1 > k''$ and any $\delta > 0$, i.e., the proposed hybrid algorithm will not stagnate at *Step D5* after iteration k''. This indicates that the hybrid algorithm will stop updating bounds in the SM after iteration k''. Lastly, assume that the algorithm will iterate indefinitely, i.e., $\epsilon_k > \epsilon$ as $k \to \infty$ in *Step D6*. This means that $(f_S^k - f_B^c)/|f_B^c| > \epsilon$ as $k \to \infty$. This statement contradicts assumption 3, which indicates that the SM eventually converges to the global optimum when bounds ($\boldsymbol{x}^{L,M}, \boldsymbol{x}^{U,M}, \boldsymbol{y}^{L,M}, \boldsymbol{y}^{U,M}$) remain fixed. Therefore, the proposed hybrid algorithm will eventually terminate and return local optimum f_B^k. ■

E.3 Illustrative iterations performed by the hybrid algorithm

Consider the results presented in Figure 5.5A of chapter 5. The steps followed in the first iterations ($k = 0$ and $k = 1$) of the proposed hybrid algorithm are explained below, which refers to *Steps S1-S5* and *Steps D1-D7* of Figure 5.2. Note that detailed descriptions of the Discrete-Steepest Descent Algorithm (DSDA) and the Differential Evolution with Tabu List (DETL) are available in [4,384], respectively.

Iteration $k = 0$: The execution of the hybrid algorithm starts by initializing the stochastic algorithm in one processor (i.e., *Steps S1-S5*) and a deterministic algorithm in another (i.e., *Steps D1-D7*). For the problems considered in this work, each processor is a separate computer, and these exchange information through a shared folder in a network drive. The first task performed by each processor is the execution of *Steps S1*, *S2.1* and *D1*. *Step S1* initializes the stochastic method (SM), which in this case is the DETL. The set of known feasible solutions (F_S) is initialized as $F_S \leftarrow \emptyset$, and the flow control parameter that decides when execution of a SM run must stop (i.e., $stop_S$) is initialized as $stop_S \leftarrow False$. Bounds of discrete and continuous DOF are initialized as shown in Table E.3. This information is then used in *Step S2.1* to initialize and execute the SM.

Table E.3. Initialization of bounds to begin the execution of the SM (iteration $k = 0$).

Continuous DOF ($\boldsymbol{x}$)				
Variable	Units	User-defined lower bounds ($\boldsymbol{x}^{L,0}$)	User-defined upper bounds ($\boldsymbol{x}^{U,0}$)	Initialization
x_1: Reflux ratio (C1)	-	$x_1^{L,0} = 0.5$	$x_1^{U,0} = 10$	$x_1^L \leftarrow x_1^{L,0}, x_1^U \leftarrow x_1^{U,0}$
x_2: Distillate flow (C1)	kmol/h	$x_2^{L,0} = 17.29$	$x_2^{U,0} = 19.11$	$x_2^L \leftarrow x_2^{L,0}, x_2^U \leftarrow x_2^{U,0}$
x_3: Distillate flow (C2)	kmol/h	$x_3^{L,0} = 8.69$	$x_3^{U,0} = 9.6$	$x_3^L \leftarrow x_3^{L,0}, x_3^U \leftarrow x_3^{U,0}$
x_4: Vapor flow (from C1 to C2)	kmol/h	$x_4^{L,0} = 11.34$	$x_4^{U,0} = 34.02$	$x_4^L \leftarrow x_4^{L,0}, x_4^U \leftarrow x_4^{U,0}$
x_5: Diameter (C1)	m	$x_5^{L,0} = 0.1$	$x_5^{U,0} = 6$	$x_5^L \leftarrow x_5^{L,0}, x_5^U \leftarrow x_5^{U,0}$
x_6: Diameter (C2)	m	$x_6^{L,0} = 0.1$	$x_6^{U,0} = 6$	$x_6^L \leftarrow x_6^{L,0}, x_6^U \leftarrow x_6^{U,0}$
Discrete DOF ($\boldsymbol{y}$)				
Variable	Units	User-defined lower bounds ($\boldsymbol{y}^{L,0}$)	User-defined upper bounds ($\boldsymbol{y}^{U,0}$)	Initialization
y_1: Number of stages (C1)	-	$y_1^{L,0} = 5$	$y_1^{U,0} = 70$	$y_1^L \leftarrow y_1^{L,0}, y_1^U \leftarrow y_1^{U,0}$
y_2: Number of stages (C2)	-	$y_2^{L,0} = 5$	$y_2^{U,0} = 70$	$y_2^L \leftarrow y_2^{L,0}, y_2^U \leftarrow y_2^{U,0}$
y_3: Feed stage (C1)	-	$y_3^{L,0} = 2$	$y_3^{U,0} = 69$	$y_3^L \leftarrow y_3^{L,0}, y_3^U \leftarrow y_3^{U,0}$
y_4: Liquid feed stage (from C2 to C1)	-	$y_4^{L,0} = 2$	$y_4^{U,0} = 69$	$y_4^L \leftarrow y_4^{L,0}, y_4^U \leftarrow y_4^{U,0}$

According to *Step S2.1*, the SM is kept running until it finds two feasible solutions. After that, *Step D1* is executed to initialize the deterministic algorithm by assigning a value of $+\infty$ to f_B^0, indicating that no local solution has been detected yet. In this particular case study, the algorithm stays at *Step S2.1* for 150.35 minutes, which is the time the SM takes to identify two feasible solutions. Then, the parallel execution of the deterministic and stochastic steps begins, and iterations start counting at $k = 1$. Note that the SM initialized in *Step S2.1* should keep running, so that new feasible information is available in F_S for subsequent iterations.

Iteration $k = 1$: While the SM is running (*Step S3*), the deterministic algorithm moves to *Step D2*, which assigns the best objective function value in F_S to f_S^1, i.e., $f_S^1 \leftarrow 645529.01\ \$/year$. Then, *Step D3* is executed to initialize DSDA-VB using information in F_S, as shown in Table E.4.

Table E. 4. Information required to initialize DSDA-VB (iteration $k = 1$).

Continuous DOF ($\boldsymbol{x}$)				
Variable	Units	Initial guess: variable values	Initial guess: lower bounds	Initial guess: upper bounds
x_1: Reflux ratio (C1)	-	$x_1^I \leftarrow 9.33$	$x_1^{L,S} \leftarrow 9.33$	$x_1^{U,S} \leftarrow 9.40$
x_2: Distillate flow (C1)	kmol/h	$x_2^I \leftarrow 18.27$	$x_2^{L,S} \leftarrow 18.27$	$x_2^{U,S} \leftarrow 18.41$
x_3: Distillate flow (C2)	kmol/h	$x_3^I \leftarrow 8.96$	$x_3^{L,S} \leftarrow 8.79$	$x_3^{U,S} \leftarrow 8.96$
x_4: Vapor flow (from C1 to C2)	kmol/h	$x_4^I \leftarrow 14.04$	$x_4^{L,S} \leftarrow 14.04$	$x_4^{U,S} \leftarrow 15.99$
x_5: Diameter (C1)	m	$x_5^I \leftarrow 5.60$	$x_5^{L,S} \leftarrow 5.60$	$x_5^{U,S} \leftarrow 5.73$
x_6: Diameter (C2)	m	$x_6^I \leftarrow 3.14$	$x_6^{L,S} \leftarrow 2.67$	$x_6^{U,S} \leftarrow 3.15$
Discrete DOF ($\boldsymbol{y}$)				
Variable	Units	Initial guess: variable values	Initial guess: lower bounds	Initial guess: upper bounds
y_1: Number of stages (C1)	-	$y_1^I \leftarrow 63$	$y_1^{L,S} \leftarrow 63$	$y_1^{U,S} \leftarrow 66$
y_2: Number of stages (C2)	-	$y_2^I \leftarrow 10$	$y_2^{L,S} \leftarrow 10$	$y_2^{U,S} \leftarrow 66$
y_3: Feed stage (C1)	-	$y_3^I \leftarrow 33$	$y_3^{L,S} \leftarrow 21$	$y_3^{U,S} \leftarrow 33$
y_4: Liquid feed stage (from C2 to C1)	-	$y_4^I \leftarrow 48$	$y_4^{L,S} \leftarrow 48$	$y_4^{U,S} \leftarrow 55$

The algorithm then goes to *Step D4*, which executes de DSDA-VB, shown in *Steps B1-B5* of Figure E.2. DSDA-VB aims to return a locally optimal solution $[\boldsymbol{x}^1, \boldsymbol{y}^1]$, and modified bounds $[\boldsymbol{x}^{L,M}, \boldsymbol{y}^{L,M}]$ and $[\boldsymbol{x}^{U,M}, \boldsymbol{y}^{U,M}]$. The algorithm presented in Table E.1 is executed in *Step B1*, which evaluates different combinations of bounds, until a combination that guarantees feasibility of the problem with discrete DOF fixed at $\boldsymbol{y}^I$ is found. In this case, the initialization of bounds presented in Table S4 guarantees these requirement and returns an objective function value of $638954.67\ \$/year$, hence bounds are initialized as $[\boldsymbol{x}^L, \boldsymbol{y}^L] \leftarrow [\boldsymbol{x}^{L,S}, \boldsymbol{y}^{L,0}]$ and $[\boldsymbol{x}^L, \boldsymbol{y}^L] \leftarrow [\boldsymbol{x}^{L,S}, \boldsymbol{y}^{L,0}]$ to start the execution of the

DSDA in *Step B2* from feasible solution $[\boldsymbol{x}^I, \boldsymbol{y}^I]$. In this case, DSDA uses the SQP solver available in Aspen Plus to solve NLP subproblems. The first execution of DSDA returns an objective function of $580590.436\ \$/year$. *Steps B3* and *B4* are then executed to verify if there is any continuous variable at its algorithmic bound in the solution reported by DSDA. Then, the relaxation algorithm in Table S2 is executed to relax the algorithmic bounds of those variables at their upper or lower bounds, and the DSDA in *Step B2* is executed again. The execution of DSDA and the relaxation algorithm are repeated iteratively until no variables are at their bounds, or the optimal value of discrete decisions is not changing in consecutive iterations. DSDA-VB ultimately returns a solution with objective function $f_D^1 = 275351.02\ \$/year$ after 58.2 minutes of execution, i.e., the current execution time of the hybrid algorithm is 208.55 minutes. Given that this is the first locally optimal solution identified by the hybrid algorithm, the best locally optimal solution at iteration $k = 1$ is $f_B^1 = f_D^1 = 275351.02\ \$/year$. The optimal solution and bounds obtained by DSDA-VB are shown in Table E.5.

The proposed hybrid algorithm continues with *Step D5*, which verifies if there has been an improvement in the best local optimum found (f_B^1) with respect to the previous iteration (f_B^0). Given that $f_B^0 = +\infty$, $\lim_{f_B^0 \to +\infty} f_B^0 - f_B^1 = +\infty$, indicating that there has been an improvement in the best local solution known. This means that the current execution of the SM can be re-started, by assigning a value of $True$ to $stop_S$, which stops the execution of the SM in *Step S4* of the stochastic algorithm. The problem solved by the SM is then updated using the information presented in Table E.5. Thus, a new execution of the SM with updated bounds is performed and a new iteration begins. A total of 126 iterations (i.e., 1825 minutes) were required for this case study to converge. Table E.6 shows the values of f_S^k and f_B^k, as well as the number of feasible solutions in set F_s for each iteration.

Table E.5. Solution obtained by DSDA-VB (iteration $k = 1$).

Continuous DOF ($\boldsymbol{x}$)				
Variable	Units	Locally optimal variable values	Modified lower bounds	Modified upper bounds
x_1: Reflux ratio (C1)	-	$x_1^1 = 8.28$	$x_1^{L,M} = 0.5$	$x_1^{U,M} = 9.40$
x_2: Distillate flow (C1)	kmol/h	$x_2^1 = 18.21$	$x_2^{L,M} = 17.29$	$x_2^{U,M} = 18.41$
x_3: Distillate flow (C2)	kmol/h	$x_3^1 = 8.92$	$x_3^{L,M} = 8.79$	$x_3^{U,M} = 8.96$
x_4: Vapor flow (from C1 to C2)	kmol/h	$x_4^1 = 19.20$	$x_4^{L,M} = 14.04$	$x_4^{U,M} = 20.78$
x_5: Diameter (C1)	m	$x_5^1 = 1.65$	$x_5^{L,M} = 0.1$	$x_5^{U,M} = 5.73$
x_6: Diameter (C2)	m	$x_6^1 = 0.24$	$x_6^{L,M} = 0.1$	$x_6^{U,M} = 3.15$
Discrete DOF ($\boldsymbol{y}$)				
Variable	Units	Locally optimal variable values	Modified lower bounds	Modified upper bounds
y_1: Number of stages (C1)	-	$y_1^1 = 54$	$y_1^{L,M} = 53$	$y_1^{U,M} = 55$
y_2: Number of stages (C2)	-	$y_2^1 = 8$	$y_2^{L,M} = 7$	$y_2^{U,M} = 9$
y_3: Feed stage (C1)	-	$y_3^1 = 23$	$y_3^{L,M} = 22$	$y_3^{U,M} = 24$
y_4: Liquid feed stage (from C2 to C1)	-	$y_4^1 = 44$	$y_4^{L,M} = 43$	$y_4^{U,M} = 45$

E.4 Comparison of the parallel-hybrid and nested-hybrid strategies

This section presents additional results for case study 1, showing how the proposed parallel-hybrid strategy performs against a nested-hybrid algorithm. The nested-hybrid algorithm implements an iterative strategy where discrete variables are optimized by the DETL, and continuous variables are directly optimized in Aspen Plus, using the SQP method. Thus, the key difference between a naïve execution of the DETL and the nested-hybrid approach is that the naïve DETL relies on black-box simulations with all DOF fixed that return objective function values, while the nested-hybrid approach relies on black-box optimizations with discrete DOF fixed that return optimized objective function values and continuous DOF. This helps to reduce the computational effort at the DETL level, which optimizes a reduced number of variables in the nested-hybrid case. Four runs of the nested-hybrid

method were performed for the first case study, using the same tunning parameters considered for the execution of the naïve DETL. The results of these tests are shown in Figures E.5A and E.5B, and compared against the parallel-hybrid strategy proposed in this thesis.

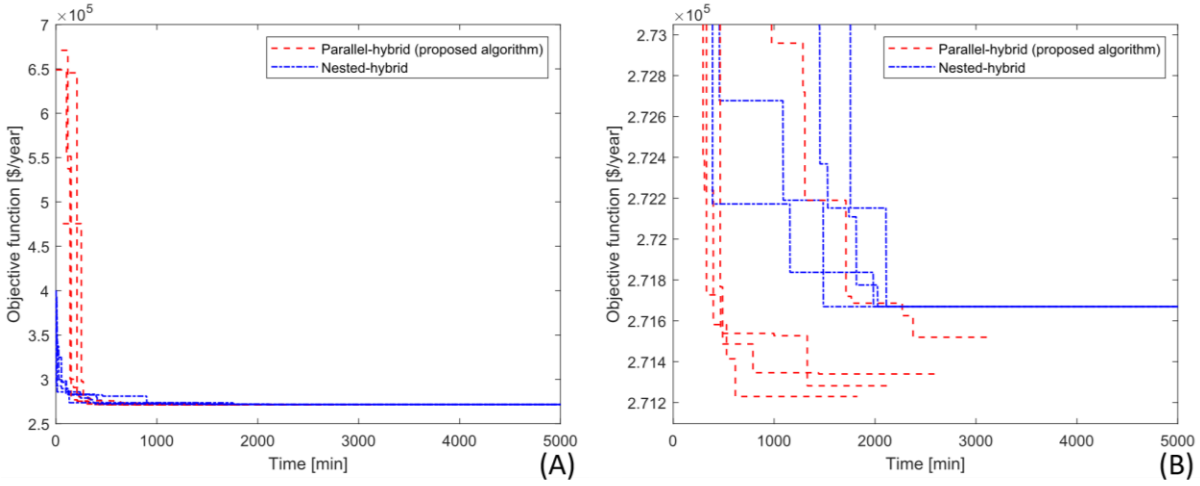


Figure E.5. Performance of the proposed parallel-hybrid strategy and a nested-hybrid strategy. A: Comparison using a time limit of 5000 seconds. B: augmented version of A.

The nested-hybrid strategy shows improvements in terms of computational speed with respect to the naïve DETL, converging to a solution with objective value of 271669 $/year before 2500 minutes of execution for all the trials performed. Despite this, the nested-hybrid algorithm is still slightly slower than the parallel-hybrid algorithm. As shown in Figure E.5B, the runs of the parallel-hybrid algorithm found their best solution after 1442 minutes on average, while the nested-hybrid runs required 1902 minutes on average, i.e., the parallel-hybrid algorithm shows an improvement of 24% in terms of computational speed when compared to the nested-hybrid algorithm for this case study. In this case, the best nested-hybrid solution was 0.2% worse than the best solution identified by the proposed parallel-hybrid framework. Also, two key limitations were identified from the nested-hybrid runs when compared to the parallel-hybrid runs. First, the best parallel-hybrid combination of discrete DOF was erroneously identified as infeasible by the nested-hybrid solution. This is attributed to the fact that the nested-hybrid algorithm always keeps the bounds of continuous DOF at the user-defined values, which may not necessarily be the best combination of bounds to guarantee convergence by the Aspen's SQP optimizer. In contrast, the proposed parallel-hybrid algorithm implements the VB strategy, which

provides a computational framework to iteratively refine continuous DOF bounds until feasibility is attained. This demonstrates the advantages of the proposed parallel-hybrid methodology to facilitate the convergence of NLP subproblems. Second, the solutions obtained by the nested-hybrid solution cannot be guaranteed that is a local optimum, e.g., there are no guarantees that the search performed by the DETL will evaluate neighboring points of a solution. In contrast, the parallel-hybrid solutions satisfy the local optimality requirement in Definition 5.1. Based on the above, the proposed hybrid strategy also offers optimality and convergence improvements when compared to a nested-hybrid methodology.

E.5 Iteration details

This section presents tables summarizing the evolution of the proposed parallel hybrid algorithm with time for case studies 1 (Tables E.6-E.9) and 2 (Table E.10). Each row presents, the computational time in minutes, the current iteration (k), the number of feasible solutions that have been identified by the stochastic algorithm ($|F_S|$), the objective function of best local optimum detected by DSDA-VB at the current iteration k (f_B^k), and the best objective function identified by the SM at the current iteration k (f_S^k). At iteration k, f_B^k is the objective function obtained after running the DSDA-VB, when initialized from the best solution in F_S. The best solution in F_S at iteration k has objective function f_S^k. Rows highlighted in blue represent those iterations when the variable bounds of the SM are updated based on the solution of DSDA-VB.

Table E. 6. Iterations for case study 1 (Figure 5.5A in chapter 5).

Time [min]	k	$\|F_S\|$	f_B^k	f_S^k
208.55	1	2	275351	645529
266.75	2	7	275351	645529
295.85	3	27	272400.8	456880.5
310.4	4	28	272239.6	303830.4
320.1	5	33	272239.6	303830.4
329.8	6	47	272239.6	303830.4
339.5	7	75	272239.6	303830.4
344.35	8	92	272239.6	303830.4
354.05	9	146	272239.6	303830.4
358.9	10	172	272239.6	303830.4
368.6	11	237	272239.6	298930
373.45	12	269	272239.6	298930
378.3	13	296	272239.6	298930

383.15	14	322	272239.6	298930
397.7	15	418	271581.4	297821.8
407.4	16	443	271581.4	293337.3
412.25	17	451	271581.4	293337.3
421.95	18	474	271581.4	293337.3
431.65	19	498	271581.4	293337.3
441.35	20	525	271581.4	293337.3
451.05	21	563	271581.4	293337.3
465.6	22	651	271581.4	293337.3
475.3	23	712	271581.4	293337.3
489.85	24	791	271581.4	293337.3
509.25	25	914	271581.4	291813.9
528.65	26	1037	271413.9	291813.9
552.9	27	1164	271413.9	291813.9
572.3	28	1310	271413.9	289042.4
577.15	29	1353	271413.9	289042.4
586.85	30	1436	271413.9	289042.4
596.55	31	1516	271413.9	289042.4
606.25	32	1590	271413.9	289042.4
615.95	33	1663	271229.4	287693.8
1825.15	126	8444	271229.4	272583.5

Table E.7. Iterations for case study 1 (Figure 5.5B in chapter 5).

Time [min]	k	$\lvert F_S \rvert$	f_B^k	f_S^k
140.65	1	2	300204.4	537369.4
155.2	2	10	300204.4	537369.4
174.6	3	63	290767.6	343978.6
227.95	4	213	280322.5	343978.6
247.35	5	243	280322.5	317875.9
257.05	6	259	280322.5	317875.9
261.9	7	265	280322.5	317875.9
281.3	8	300	280322.5	317875.9
286.15	9	319	280322.5	312468.4
295.85	10	365	280322.5	312468.4
300.7	11	394	279991.3	312468.4
310.4	12	428	279991.3	312468.4
320.1	13	438	277932.5	305387.3
324.95	14	447	277932.5	305387.3
334.65	15	478	277932.5	305387.3

344.35	16	525	277932.5	305387.3
354.05	17	593	277932.5	305387.3
363.75	18	657	277932.5	305387.3
373.45	19	745	277932.5	305387.3
383.15	20	810	277932.5	305387.3
397.7	21	921	275933.4	304412.2
407.4	22	934	275933.4	304155.7
417.1	23	953	275933.4	304155.7
426.8	24	989	275933.4	304155.7
446.2	25	1091	275933.4	304155.7
465.6	26	1224	275933.4	304155.7
475.3	27	1287	275933.4	301452.1
485	28	1355	275933.4	301452.1
494.7	29	1422	275933.4	301452.1
504.4	30	1477	275933.4	299077.4
514.1	31	1531	275933.4	299077.4
523.8	32	1583	275933.4	299077.4
538.35	33	1674	275665.4	298710.2
557.75	34	1771	275665.4	296912.6
567.45	35	1804	275665.4	288728.8
577.15	36	1855	275665.4	288728.8
586.85	37	1902	275665.4	288728.8
596.55	38	1967	275665.4	288728.8
606.25	39	2022	275665.4	288728.8
620.8	40	2115	275665.4	288728.8
640.2	41	2257	273494.4	288263.9
664.45	42	2316	273494.4	288263.9
683.85	43	2371	273349.1	288263.9
703.25	44	2424	273349.1	288263.9
727.5	45	2541	273349.1	288263.9
746.9	46	2654	273349.1	288263.9
771.15	47	2796	273349.1	288263.9
790.55	48	2918	273349.1	288263.9
814.8	49	3082	273349.1	288263.9
834.2	50	3217	273349.1	288263.9
858.45	51	3392	273349.1	288263.9
882.7	52	3559	273349.1	288263.9
897.25	53	3654	273349.1	288263.9
916.65	54	3778	273349.1	288263.9
931.2	55	3875	273349.1	288263.9
955.45	56	4030	273349.1	285959.4

974.85	57	4133	272958.8	283130.4
989.4	58	4163	272958.8	283130.4
1008.8	59	4207	272958.8	283130.4
1028.2	60	4263	272958.8	283130.4
1047.6	61	4326	272958.8	283130.4
1062.15	62	4377	272958.8	283130.4
1081.55	63	4469	272958.8	283130.4
1100.95	64	4564	272958.8	283130.4
1115.5	65	4635	272958.8	283130.4
1134.9	66	4754	272958.8	283130.4
1154.3	67	4862	272958.8	283130.4
1173.7	68	4987	272958.8	283130.4
1193.1	69	5092	272958.8	283130.4
1207.65	70	5188	272958.8	283130.4
1227.05	71	5306	272958.8	283130.4
1246.45	72	5414	272958.8	283130.4
1265.85	73	5527	272958.8	283130.4
1285.25	74	5625	272719.8	282740.2
1304.65	75	5665	272189.8	281939.7
1328.9	76	5733	272189.8	281939.7
1348.3	77	5846	272189.8	281939.7
1372.55	78	6005	272189.8	281939.7
1396.8	79	6156	272189.8	281939.7
1416.2	80	6292	272189.8	281939.7
1440.45	81	6457	272189.8	281939.7
1464.7	82	6612	272189.8	281939.7
1484.1	83	6734	272189.8	281939.7
1508.35	84	6883	272189.8	281939.7
1532.6	85	7026	272189.8	281939.7
1547.15	86	7111	272189.8	281939.7
1561.7	87	7193	272189.8	281939.7
1576.25	88	7287	272189.8	281939.7
1590.8	89	7380	272189.8	281939.7
1605.35	90	7445	272189.8	281939.7
1619.9	91	7520	272189.8	281939.7
1634.45	92	7598	272189.8	281939.7
1653.85	93	7688	272189.8	281939.7
1668.4	94	7763	272189.8	281939.7
1682.95	95	7832	272189.8	281939.7
1712.05	96	7971	271719.6	279853.9
1736.3	97	7994	271719.6	279853.9

1765.4	98	8028	271686.4	279853.9
1794.5	99	8063	271686.2	279853.9
1813.9	100	8140	271686.2	279853.9
1833.3	101	8245	271686.2	279853.9
1857.55	102	8393	271686.2	279853.9
1876.95	103	8502	271686.2	279853.9
1896.35	104	8599	271686.2	279853.9
1915.75	105	8704	271686.2	279853.9
1940	106	8849	271686.2	279853.9
1959.4	107	8958	271686.2	279853.9
1983.65	108	9095	271686.2	279853.9
2003.05	109	9210	271686.2	279853.9
2012.75	110	9267	271686.2	279853.9
2027.3	111	9324	271686.2	279853.9
2037	112	9363	271686.2	279853.9
2051.55	113	9435	271686.2	279853.9
2061.25	114	9489	271686.2	279853.9
2075.8	115	9552	271686.2	279853.9
2085.5	116	9597	271686.2	279853.9
2100.05	117	9682	271686.2	279853.9
2109.75	118	9735	271686.2	279853.9
2124.3	119	9804	271686.2	279853.9
2134	120	9853	271686.2	279853.9
2148.55	121	9923	271686.2	276942.8
2158.25	122	9962	271686.2	276942.8
2172.8	123	10021	271686.2	276942.8
2182.5	124	10058	271686.2	276942.8
2197.05	125	10116	271686.2	276942.8
2206.75	126	10159	271686.2	276942.8
2221.3	127	10209	271686.2	276942.8
2231	128	10257	271686.2	276942.8
2240.7	129	10297	271686.2	276942.8
2255.25	130	10352	271686.2	276942.8
2269.8	131	10415	271625.2	275657.8
2289.2	132	10456	271625.2	275657.8
2303.75	133	10487	271625.2	275657.8
2318.3	134	10575	271625.2	275657.8
2328	135	10645	271625.2	275657.8
2347.4	136	10794	271625.2	275657.8
2361.95	137	10901	271625.2	275657.8
2376.5	138	11012	271519.5	275657.8

3165.79	189	15056	271519.5	272761.3

Table E.8. Iterations for case study 1 (Figure 5.5C in chapter 5).

Time [min]	k	$\lvert F_S \rvert$	f_B^k	f_S^k
150.35	1	2	276903.6	551489.4
198.85	2	75	276903.6	551489.4
213.4	3	135	275018.8	314247
223.1	4	149	275018.7	314247
237.65	5	188	275018.7	314247
252.2	6	239	275018.7	314247
266.75	7	354	275018.7	314247
281.3	8	470	275018.7	314247
291	9	544	275018.7	314247
300.7	10	606	275018.7	309493.2
310.4	11	667	275018.7	309493.2
329.8	12	808	271726.7	304345.4
344.35	13	839	271726.7	297685
354.05	14	854	271726.7	297685
373.45	15	898	271726.7	297685
388	16	950	271726.7	297685
397.7	17	999	271726.7	297685
412.25	18	1098	271726.7	297685
421.95	19	1173	271726.7	297685
431.65	20	1236	271726.7	297685
441.35	21	1299	271726.7	297685
455.9	22	1408	271726.7	297685
470.45	23	1513	271564.8	297140
489.85	24	1574	271486.3	296504.3
509.25	25	1615	271486.3	296504.3
514.1	26	1621	271486.3	296504.3
518.95	27	1631	271486.3	296504.3
523.8	28	1646	271486.3	296504.3
528.65	29	1670	271486.3	296504.3
533.5	30	1694	271486.3	296504.3
611.1	31	2222	271486.3	296504.3
630.5	32	2348	271486.3	295172.3
649.9	33	2490	271486.3	295172.3
659.6	34	2565	271486.3	292524.6
664.45	35	2600	271486.3	291227.5

669.3	36	2639	271486.3	291227.5
674.15	37	2671	271486.3	291227.5
679	38	2703	271486.3	291227.5
688.7	39	2767	271486.3	291227.5
693.55	40	2809	271486.3	286786.9
722.65	41	2998	271486.3	286026.6
746.9	42	3154	271486.3	286026.6
776	43	3326	271486.3	286026.6
790.55	44	3411	271345.8	282521.7
805.1	45	3443	271345.8	278418.5
824.5	46	3518	271345.8	278418.5
843.9	47	3636	271345.8	278418.5
858.45	48	3726	271345.8	278418.5
877.85	49	3865	271345.8	278418.5
897.25	50	3985	271345.8	278418.5
916.65	51	4101	271345.8	278418.5
936.05	52	4225	271345.8	278418.5
955.45	53	4343	271345.8	278418.5
965.15	54	4399	271345.8	278418.5
979.7	55	4500	271345.8	278418.5
989.4	56	4559	271345.8	278418.5
1003.95	57	4655	271345.8	278418.5
1013.65	58	4712	271345.8	278418.5
1023.35	59	4772	271345.8	278418.5
1037.9	60	4854	271345.8	278418.5
1047.6	61	4905	271345.8	278418.5
1057.3	62	4963	271345.8	278418.5
1071.85	63	5049	271345.8	278418.5
1081.55	64	5102	271345.8	278418.5
1096.1	65	5189	271345.8	278418.5
1105.8	66	5246	271345.8	278418.5
1120.35	67	5340	271345.8	278418.5
1130.05	68	5388	271345.8	278418.5
1144.6	69	5482	271345.8	278418.5
1154.3	70	5529	271345.8	278418.5
1164	71	5576	271345.8	278418.5
1178.55	72	5663	271345.8	278418.5
1188.25	73	5712	271345.8	278418.5
1202.8	74	5790	271345.8	278418.5
1212.5	75	5833	271345.8	278418.5
1227.05	76	5909	271345.8	278418.5

1236.75	77	5956	271345.8	278418.5
1251.3	78	6016	271345.8	278418.5
1265.85	79	6077	271345.8	278418.5
1275.55	80	6124	271345.8	278418.5
1285.25	81	6179	271345.8	277859.4
1294.95	82	6227	271345.8	277245.4
1309.5	83	6297	271345.8	277245.4
1319.2	84	6339	271345.8	277245.4
1333.75	85	6412	271345.8	277245.4
1343.45	86	6461	271345.8	277245.4
1358	87	6530	271345.8	277245.4
1372.55	88	6602	271345.8	277245.4
1382.25	89	6657	271345.8	277245.4
1396.8	90	6707	271345.8	277245.4
1406.5	91	6743	271345.8	277245.4
1421.05	92	6793	271345.8	277245.4
1440.45	93	6870	271339.7	275915.4
2624.04	184	12534	271339.7	272695.1

Table E.9. Iterations for case study 1 (Figure 5.5D in chapter 5).

Time [min]	k	$\lvert F_S \rvert$	f_B^k	f_S^k
252.2	1	4	297621.2	475507.7
271.6	2	14	275622.3	457088.4
315.25	3	70	275622.3	367688.8
320.1	4	72	273258.9	305358.5
354.05	5	198	273258.9	305358.5
465.6	6	897	271765.7	285794.3
489.85	7	943	271538.3	285794.3
514.1	8	966	271538.3	285794.3
533.5	9	1013	271538.3	285794.3
557.75	10	1126	271538.3	285794.3
582	11	1281	271538.3	285794.3
606.25	12	1442	271538.3	285794.3
625.65	13	1551	271538.3	285794.3
649.9	14	1682	271538.3	285794.3
679	15	1857	271538.3	285794.3
688.7	16	1903	271538.3	284416.6
703.25	17	1993	271538.3	284416.6
712.95	18	2051	271538.3	284416.6

717.8	19	2071	271538.3	282402.9
722.65	20	2092	271538.3	282402.9
727.5	21	2120	271538.3	282402.9
732.35	22	2144	271538.3	282402.9
742.05	23	2188	271538.3	282402.9
746.9	24	2219	271538.3	282402.9
751.75	25	2238	271538.3	282402.9
771.15	26	2321	271538.3	279689.5
795.4	27	2435	271538.3	279689.5
824.5	28	2590	271538.3	279689.5
839.05	29	2666	271538.3	279287
853.6	30	2731	271538.3	279287
868.15	31	2795	271538.3	279287
897.25	32	2914	271538.3	278701.7
921.5	33	3024	271538.3	278701.7
926.35	34	3043	271538.3	278165.8
936.05	35	3087	271538.3	278165.8
945.75	36	3129	271538.3	278165.8
950.6	37	3148	271538.3	278165.8
960.3	38	3181	271538.3	278165.8
970	39	3214	271538.3	278165.8
999.1	40	3325	271527	278083.9
1023.35	41	3391	271527	277671.2
1042.75	42	3453	271527	277477.5
1062.15	43	3524	271527	277477.5
1076.7	44	3593	271527	277477.5
1096.1	45	3680	271527	277477.5
1115.5	46	3768	271527	277477.5
1134.9	47	3874	271527	277477.5
1154.3	48	3991	271527	277477.5
1173.7	49	4111	271527	277477.5
1193.1	50	4215	271527	277477.5
1202.8	51	4269	271527	277477.5
1217.35	52	4355	271527	277477.5
1227.05	53	4409	271527	277477.5
1236.75	54	4471	271527	277477.5
1251.3	55	4564	271527	277477.5
1261	56	4624	271527	277477.5
1270.7	57	4672	271527	277477.5
1285.25	58	4755	271527	277477.5
1294.95	59	4799	271527	277477.5

1309.5	60	4863	271527	277477.5
1328.9	61	4962	271281.8	276501.6
2125.27	103	8019	271281.8	272504.6

Table E.10. Iterations for case study 2 (Figure 5.8 in chapter 5).

Time [min]	k	$\|F_S\|$	f_B^k	f_S^k
1833	1	2	878540.4	1134177
2275	2	7	878540.4	1134177
2288	3	8	878540.4	1000910
2301	4	8	878540.4	1000910
2314	5	8	878540.4	1000910
2327	6	9	878540.4	1000910
2340	7	9	878540.4	1000910
2353	8	9	878540.4	1000910
2379	9	9	878540.4	1000910
2392	10	10	878540.4	1000910
2405	11	10	878540.4	1000910
2418	12	11	878540.4	1000910
2431	13	13	878540.4	1000910
2457	14	13	878540.4	1000910
2470	15	13	878540.4	1000910
2496	16	17	878540.4	1000910
2509	17	18	878540.4	987397
2522	18	19	878540.4	987397
2548	19	19	878540.4	987397
2561	20	21	878540.4	987397
2574	21	21	878540.4	987397
2600	22	21	878540.4	987397
2613	23	21	878540.4	987397
2626	24	21	878540.4	987397
2639	25	22	878540.4	987397
2665	26	23	878540.4	987397
2678	27	24	878540.4	987397
2691	28	24	878540.4	987397
2717	29	27	878540.4	987397
2730	30	28	878540.4	987397
2743	31	29	878540.4	987397
2769	32	38	878540.4	987397
2782	33	38	878540.4	987397
2795	34	40	878540.4	987397
2821	35	43	878540.4	987397

2834	36	45	878540.4	987397
2860	37	48	878540.4	987397
2873	38	48	878540.4	987397
2886	39	49	878540.4	987397
2899	40	50	878540.4	987397
2925	41	52	878540.4	987397
2938	42	56	878540.4	987397
2964	43	60	878540.4	987397
2977	44	62	878540.4	987397
2990	45	68	878540.4	987397
3016	46	73	878540.4	987397
3029	47	75	878540.4	987397
3055	48	81	878540.4	987397
3692	49	218	878356.7	987397
3705	50	219	878356.7	959781.5
3718	51	219	878356.7	959781.5
3744	52	221	878356.7	959781.5
3757	53	222	878356.7	959781.5
3783	54	224	878356.7	959781.5
4498	55	560	878356.7	959781.5
5473	56	1318	874157.8	959781.5
5499	57	1324	874157.8	949343.7
5512	58	1324	874157.8	949343.7
5538	59	1324	874157.8	949343.7
5564	60	1324	874157.8	949343.7
5590	61	1324	874157.8	949343.7
5616	62	1324	874157.8	949343.7
5642	63	1324	874157.8	949343.7
5668	64	1324	874157.8	949343.7
5681	65	1324	874157.8	949343.7
5707	66	1325	874157.8	949343.7
5733	67	1325	874157.8	949343.7
5746	68	1326	874157.8	949343.7
5772	69	1326	874157.8	949343.7
5785	70	1326	874157.8	949343.7
5798	71	1327	874157.8	949343.7
5824	72	1336	874157.8	949343.7
5850	73	1342	874157.8	949343.7
5863	74	1344	874157.8	949343.7
5889	75	1347	874157.8	949343.7
5915	76	1347	874157.8	949343.7

5941	77	1350	874157.8	949343.7
5954	78	1353	874157.8	949343.7
5980	79	1359	874157.8	949343.7
6006	80	1370	874157.8	949343.7
6019	81	1373	874157.8	949343.7
6045	82	1377	874157.8	949343.7
6071	83	1383	874157.8	949343.7
6084	84	1389	874157.8	949343.7
6110	85	1399	874157.8	949343.7
6123	86	1405	874157.8	949343.7
6149	87	1424	874157.8	949343.7
6175	88	1440	874157.8	949343.7
6188	89	1446	874157.8	949343.7
6214	90	1457	874157.8	949343.7
6240	91	1467	874157.8	949343.7
6253	92	1475	874157.8	949343.7
6279	93	1489	874157.8	949343.7
6292	94	1495	874157.8	949343.7
6318	95	1500	874157.8	949343.7
6344	96	1508	874157.8	949343.7
6357	97	1510	874157.8	949343.7
6383	98	1522	874157.8	949343.7
6409	99	1535	874157.8	949343.7
6422	100	1541	874157.8	949343.7
6448	101	1558	874157.8	949343.7
6474	102	1567	874157.8	949343.7
6487	103	1575	874157.8	949343.7
6513	104	1591	874157.8	949343.7
6526	105	1595	874157.8	949343.7
6552	106	1606	874157.8	949343.7
6565	107	1609	874157.8	949343.7
6591	108	1630	874157.8	949343.7
6617	109	1647	874157.8	949343.7
6630	110	1653	874157.8	949343.7
6656	111	1674	874157.8	949343.7
6669	112	1681	874157.8	949343.7
6695	113	1698	874157.8	949343.7
6708	114	1701	874157.8	949343.7
6734	115	1714	874157.8	949343.7
6747	116	1723	874157.8	949343.7
6773	117	1734	874157.8	949343.7

6786	118	1738	874157.8	949343.7
6812	119	1748	874157.8	949343.7
6838	120	1766	874157.8	949343.7
6851	121	1776	874157.8	949343.7
6877	122	1792	874157.8	949343.7
6890	123	1803	874157.8	949343.7
6916	124	1817	874157.8	949343.7
6929	125	1827	874157.8	949343.7
6955	126	1848	874157.8	949343.7
6968	127	1853	874157.8	949343.7
6994	128	1866	874157.8	949343.7
7007	129	1869	874157.8	949343.7
7020	130	1874	874157.8	949343.7
7033	131	1878	874157.8	949343.7
7059	132	1896	874157.8	949343.7
7072	133	1898	874157.8	949343.7
7085	134	1903	874157.8	949343.7
7098	135	1913	874157.8	949343.7
7111	136	1920	874157.8	949343.7
7124	137	1926	874157.8	949343.7
7137	138	1934	874157.8	949343.7
7150	139	1938	874157.8	949343.7
7163	140	1948	874157.8	949343.7
7176	141	1953	874157.8	949343.7
7202	142	1967	874157.8	949343.7
7215	143	1974	874157.8	949343.7
7228	144	1974	874157.8	949343.7
7241	145	1974	874157.8	949343.7
7254	146	1974	874157.8	949343.7
7306	147	1975	874157.8	949343.7
7345	148	1975	874157.8	949343.7
7358	149	1976	874157.8	949343.7
7371	150	1977	874157.8	949343.7
7397	151	1977	874157.8	949343.7
7410	152	1977	874157.8	949343.7
7436	153	1979	874157.8	949343.7
7462	154	1980	874157.8	949343.7
7488	155	1980	874157.8	949343.7
7501	156	1980	874157.8	949343.7
7527	157	1982	874157.8	949343.7
7553	158	1986	874157.8	949343.7

7566	159	1988	874157.8	949343.7
7592	160	1994	874157.8	949343.7
7826	161	2032	868193.2	946637
7917	162	2037	868193.2	946637
8671	163	2154	868193.2	946637
9009	164	2276	868193.2	946637
9334	165	2417	868193.2	946637
9737	166	2560	868193.2	946637
10244	167	2769	868193.2	946637
10452	168	2775	867731.8	946637
10465	169	2775	867731.8	946637
11245	170	2785	867731.8	946637
11583	171	2807	867731.8	946637
12077	172	2890	867731.8	946637
12649	173	3057	867731.8	946637
15288	174	4183	867731.8	946637
15405	175	4227	867731.8	936945.6
15522	176	4276	867731.8	936945.6
17043	177	4965	867731.8	936945.6
17641	178	5241	867731.8	912210.9
17979	179	5406	867731.8	912210.9
18330	180	5582	867731.8	912210.9
18824	181	5800	867731.8	912210.9
18837	182	5804	867731.8	908139.6
18863	183	5809	867731.8	908139.6
18889	184	5823	867731.8	908139.6
18902	185	5829	867731.8	908139.6
18915	186	5835	867731.8	908139.6
18941	187	5844	867731.8	908139.6
18954	188	5852	867731.8	908139.6
18967	189	5857	867731.8	908139.6
18993	190	5867	867731.8	908139.6
19006	191	5872	867731.8	908139.6
19019	192	5879	867731.8	908139.6
19045	193	5892	867731.8	908139.6
19656	194	6151	867731.8	898612.7
19669	195	6153	867731.8	898514.7
19695	196	6160	867731.8	898514.7
19721	197	6173	867731.8	898514.7
19734	198	6180	867731.8	898514.7
19760	199	6195	867731.8	898514.7

19773	200	6198	867731.8	898514.7
19799	201	6211	867731.8	898514.7
19825	202	6220	867731.8	898514.7
19838	203	6227	867731.8	898514.7
19864	204	6240	867731.8	898514.7
19877	205	6243	867731.8	898514.7
19903	206	6257	867731.8	898514.7
19929	207	6271	867731.8	898514.7
19942	208	6279	867731.8	898514.7
19968	209	6288	867731.8	898514.7
19981	210	6293	867731.8	898514.7
20007	211	6306	867731.8	898514.7
20033	212	6322	867731.8	898514.7
20059	213	6333	867731.8	898514.7
20085	214	6346	867731.8	898514.7
20111	215	6357	867731.8	898514.7
20137	216	6372	867731.8	898514.7
20150	217	6380	867731.8	898514.7
20176	218	6389	867731.8	898514.7
20189	219	6392	867731.8	898514.7
20215	220	6398	867731.8	898514.7
20241	221	6408	867731.8	898514.7
20267	222	6421	867731.8	898514.7
20293	223	6431	867731.8	898514.7
20319	224	6438	867731.8	898514.7
20332	225	6447	867731.8	898514.7
20358	226	6461	867731.8	898514.7
20384	227	6473	867731.8	898514.7
20410	228	6482	867731.8	898514.7
20436	229	6489	867731.8	898514.7
20462	230	6500	867731.8	898514.7
20488	231	6513	867731.8	898514.7
20501	232	6517	867731.8	898514.7
20527	233	6530	867731.8	898514.7
20553	234	6535	867731.8	898514.7
20579	235	6542	867731.8	898514.7
20605	236	6553	867731.8	898514.7
20631	237	6566	867731.8	898514.7
20644	238	6571	867731.8	898514.7
20670	239	6581	867731.8	898514.7
20696	240	6586	867731.8	898514.7

20722	241	6596	867731.8	898514.7
20748	242	6603	867731.8	898514.7
20761	243	6606	867731.8	898514.7
20787	244	6615	867731.8	898514.7
20813	245	6627	867731.8	898514.7
20839	246	6635	867731.8	898514.7
20865	247	6647	867731.8	898514.7
20891	248	6658	867731.8	898514.7
20917	249	6667	867731.8	898514.7
20930	250	6671	867731.8	898514.7
20956	251	6683	867731.8	898514.7
20982	252	6694	867731.8	898514.7
21008	253	6706	867731.8	898514.7
21034	254	6714	867731.8	898514.7
21060	255	6730	867731.8	898514.7
21086	256	6742	867731.8	898514.7
21112	257	6758	867731.8	898514.7
21138	258	6771	867731.8	898514.7
21151	259	6773	867731.8	898514.7
21177	260	6780	867731.8	898514.7
21203	261	6788	867731.8	898514.7
21996	262	7105	867731.8	897704.3
22672	263	7398	867731.8	894645.2
23322	264	7668	867731.8	894645.2
23712	265	7834	867731.8	893780.9
24401	266	8084	867731.8	891738.7
24726	267	8201	867731.8	885693
25077	268	8334	867731.8	885693
25363	269	8439	867731.8	885693
25909	270	8642	867731.8	885693
26390	271	8827	867731.8	884129.2
27378	272	9183	863423.2	882641.1
49999.95	442	18418	863423.2	875273.2

Appendix F
Supplementary Material for Chapter 6

F.1 Theorem proof

To proof theorem 6.1, first consider the following lemma.

Lemma F.1. *The optimal solution* $f_{SUB}^{SS}(\boldsymbol{e})$ *for a fixed value of* $\boldsymbol{e}$ *in* Eq. (6.11) *provides a lower bound for the optimization problem in* Eq. (F.1).

$$\min_{\boldsymbol{v}^{SS},\boldsymbol{\varphi}} f_{OBJ}^{SS}(\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})) \tag{F.1}$$
$$s.t.$$
$$[\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{SS}, [\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{TR}$$

Proof of Lemma F.1. From Eq. (6.11), $f_{SUB}^{SS}(\boldsymbol{e}) = f_{OBJ}^{SS}(\boldsymbol{v}^{SS*},\boldsymbol{\varphi}^{*},\boldsymbol{y}(\boldsymbol{e})) \leq f_{OBJ}^{SS}(\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e}))$ for every $[\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})]$ such that $[\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$.

Now, it is possible to prove the lemma by contradiction. Suppose the opposite, i.e., $f_{SUB}^{SS}(\boldsymbol{e})$ is not a lower bound for the problem in Eq. (F.1). Then, $f_{SUB}^{SS}(\boldsymbol{e}) = f_{OBJ}^{SS}(\boldsymbol{v}^{SS*},\boldsymbol{\varphi}^{*},\boldsymbol{y}(\boldsymbol{e})) > f_{OBJ}^{SS}(\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e}))$ for at least one value $[\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})]$ such that $[\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$ and $[\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{TR}$. In other words, there exist a $[\tilde{\boldsymbol{v}}^{SS},\tilde{\boldsymbol{v}}^{TR},\tilde{\boldsymbol{\varphi}},\boldsymbol{y}(\boldsymbol{e})]$ such $[\tilde{\boldsymbol{v}}^{SS},\tilde{\boldsymbol{\varphi}},\boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$, $[\tilde{\boldsymbol{v}}^{SS},\tilde{\boldsymbol{v}}^{TR},\tilde{\boldsymbol{\varphi}},\boldsymbol{y}(\boldsymbol{e})] \in S^{TR}$ and $f_{OBJ}^{SS}(\boldsymbol{v}^{SS*},\boldsymbol{\varphi}^{*},\boldsymbol{y}(\boldsymbol{e})) > f_{OBJ}^{SS}(\tilde{\boldsymbol{v}}^{SS},\tilde{\boldsymbol{\varphi}},\boldsymbol{y}(\boldsymbol{e}))$. This contradicts the fact that $f_{OBJ}^{SS}(\boldsymbol{v}^{SS*},\boldsymbol{\varphi}^{*},\boldsymbol{y}(\boldsymbol{e})) \leq f_{OBJ}^{SS}(\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e}))$ for every $[\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})]$ such that $[\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$■

Proof of Theorem 6.1. By Lemma F.1, $f_{SUB}^{SS}(\boldsymbol{e}) \leq f_{OBJ}^{SS}(\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e}))$ for every $[\boldsymbol{v},\boldsymbol{y}(\boldsymbol{e})]$ such that $[\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$ and $[\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{TR}$. Since $f_{OBJ}^{TR}(\boldsymbol{v}^{SS},\boldsymbol{v}^{TR}) \geq 0$ for every $[\boldsymbol{v},\boldsymbol{y}(\boldsymbol{e})]$ such that $[\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$and $[\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{TR}$. Consequently, $f_{SUB}^{SS}(\boldsymbol{e}) \leq f_{OBJ}^{SS}\big(\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})\big) \leq f_{OBJ}^{SS}\big(\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})\big) + f_{OBJ}^{TR}(\boldsymbol{v}^{SS},\boldsymbol{v}^{TR}) = f_{OBJ}\big(\boldsymbol{v},\boldsymbol{y}(\boldsymbol{e})\big)$ for any $[\boldsymbol{v},\boldsymbol{y}(\boldsymbol{e})]$ such that $[\boldsymbol{v}^{SS},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{SS}$ and $[\boldsymbol{v}^{SS},\boldsymbol{v}^{TR},\boldsymbol{\varphi},\boldsymbol{y}(\boldsymbol{e})] \in S^{TR}$.■

F.2 Initialization subroutine

In this section, the methodology to initialize and solve the NLP dynamic subproblems that appear throughout module 3 (Algorithm 6.1) of the proposed solution strategy are discussed, based on our computational experience with CONOPT4 and the orthogonal collocation technique to solve these problems.

According to Algorithm 6.1, the optimization of an NLP dynamic subproblem with $\boldsymbol{e}$ fixed requires the optimization of the problem in Eq. (6.10B), which results in an objective function value of $f_{SUB}(\boldsymbol{e}) = f_{OBJ}(\boldsymbol{v}^{SSo}(\boldsymbol{e}), \boldsymbol{v}^{TRo}(\boldsymbol{e}), \boldsymbol{\varphi}^{o}(\boldsymbol{e}))$. To avoid the misleading detection of infeasibility by the solver due to the nonlinearities and size of the problem, a rigorous initialization strategy was used to improve convergence. At each iteration, a steady state *multi-scenario* subproblem ($f_{SUB}^{SS}(\boldsymbol{e})$) is solved first to compute lower bounds. These steady state subproblems can be initialized using standard steady state initialization routines created for the standard D-SDA [4,7]. After that, the solution of steady state subproblems $[\boldsymbol{v}^{SS*}(\boldsymbol{e}), \boldsymbol{\varphi}^{*}(\boldsymbol{e})]$ are used to initialize dynamic subproblems as described in Figure F.1. This procedure was used to optimize dynamic subproblems for both sequential and simultaneous approaches, except that step S6 in Figure F.1 (the step where dynamics are integrated with steady state variables) was not included for the sequential approach.

According to Figure F.1, the first stage consists of initializing the dynamic variables $\boldsymbol{v}^{TR}(\boldsymbol{e})$ with steady state values $[[\boldsymbol{v}^{SS*}(\boldsymbol{e}), \boldsymbol{\varphi}^{*}(\boldsymbol{e})]$ obtained from module 2, while keeping design variables fixed. This stage comprises three steps (S1, S2 and S3) which are executed iteratively over finite elements n_{FE} until the desired number of finite elements n_{FE}^{total} is attained. After a new finite element is added, feasibility is tested with CONOPT4, which guarantees good initializations throughout the iterative procedure. The result from this loop is used at S4, where a simulation of the process with step inputs in $\boldsymbol{\mu}(t)$ is performed. Note that the model used through S1-S4 should be a simplification of the equations described in S that does not consider product specification constraints and Eqs. (6.7A)-(6.7F), i.e., continuous differentiability between scenarios is not enforced at these stages. Also, the values of $\Delta \boldsymbol{t}_k, \forall k \in K$ are used as parameters from S1 to S4, although their width must satisfy the constraints in Eq. (6.6C).

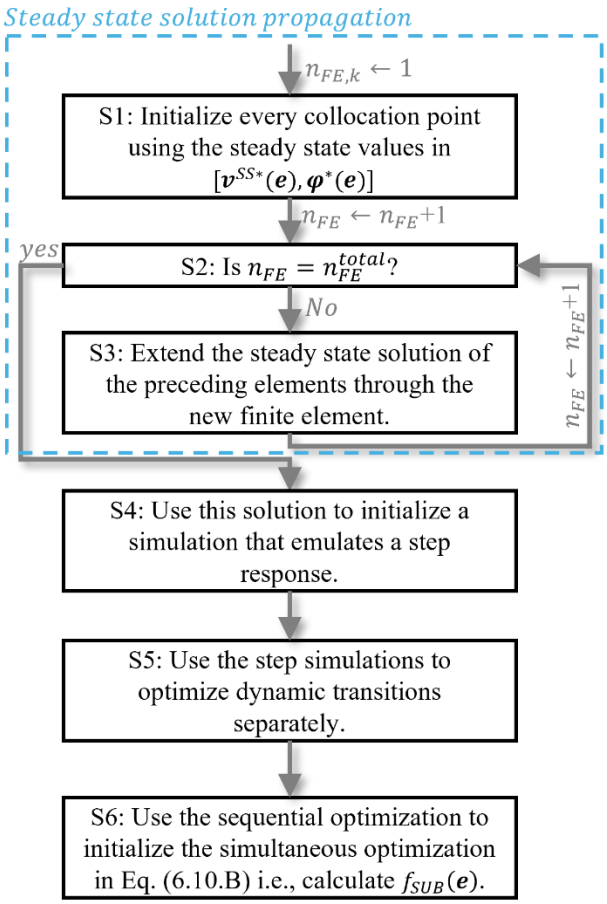


Figure F.1. Solution of NLP dynamic multi-scenario subproblems with $\boldsymbol{e}$ fixed.

The author emphasizes that there are two options to extend steady state variables through time in steps S1 and S3. Every time slot comprises a transition period $(t_{k-1}, t_k^{TR}]$ and a point in time t_k after t_k^{TR} when steady state is attained. Thus, there are two possibilities: extending the steady state information of t_{k-1} through $(t_{k-1}, t_k^{TR}]$ (forward approximation) or extending the steady state information at t_k through $(t_{k-1}, t_k^{TR}]$ (backward approximation). In this study, I found the backward approximation more efficient, because it guaranteed a good initialization for step S4. This is because the backward approximation enforces feasibility over Eqs. (6.7C)-(6.7F) and the desired values of the dynamic variables at t_k^{TR}, while the forward approximation does not. Therefore, the forward approximation results in an initialization of S4 that presents a larger measure of infeasibility, which is not desirable for a feasible path solver such as CONOPT4.

The initialization procedure described above was used to initialize the optimization steps S5 and S6, for which every constraint S was considered. The intention of S5 is to optimize every scenario as if they were independent, by keeping continuous design variables $\boldsymbol{\varphi}$ and steady state variables fixed $\boldsymbol{v}^{SS}(\boldsymbol{e})$. This offers the advantage that the resulting problem in S5 is easier than the simultaneous optimization in S6. If feasibility is not obtained at S5, it is possible to skip this step and continue directly to S6, where the NLP subproblem is solved while considering every term, $[\boldsymbol{v}^{SS}(\boldsymbol{e}), \boldsymbol{v}^{TR}(\boldsymbol{e}), \boldsymbol{\varphi}(\boldsymbol{e})]$, as variable. The addition of this step was included after experimentation with CONOPT4, which found better solutions using S5 and S6 than using S6 alone, and sometimes failed to prove feasibility when skipping S5. For the case study, multithread and parallel computing strategies were used to improve the solution of dynamic subproblems. In particular, each of the following steps were solved in parallel for each scenario k: the steady state propagation steps (S1, S2 and S3, Figure F.1); the initialization with step inputs (S4, Figure F.1), and the independent NLP dynamic optimizations used to initialize the NLP dynamic subproblems (S5, Figure F.1). After that, the *multi-scenario* NLP dynamic subproblem (S5, Figure F.1) was solved using 28 threads, for every fixed value of $\boldsymbol{e}$. These strategies allowed to reduce the total computational time to half on average.

F.3 Reformulation with external variables (case study)

The reformulation with external variables considered for this study is presented in this section. First, the independent binary variables $\boldsymbol{y}_I$ must be calculated in terms of the external variables $\boldsymbol{e} = [e_1, e_2, e_{3,1}, e_{3,2}, e_{3,3}, e_6]$ as follows.

$$YF_{n,1} = \begin{cases} 1, e_1 = n \\ 0, otherwise \end{cases}, \forall n \in \{2,3,\dots,44\} \tag{F.2}$$

$$YF_{n,2} = \begin{cases} 1, e_4 - e_2 = n \\ 0, otherwise \end{cases}, \forall n \in \{2,3,\dots,44\} \tag{F.3}$$

$$YC_{1,n} = \begin{cases} 1, e_1 + e_{3,1} = n \\ 0, otherwise \end{cases}, \forall n \in \{2,3,\dots,44\} \tag{F.4}$$

$$YC_{2,n} = \begin{cases} 1, e_1 + e_{3,1} + e_{3,2} = n \\ 0, otherwise \end{cases}, \forall n \in \{2,3,\dots,44\} \tag{F.5}$$

$$YC_{3,n} = \begin{cases} 1, e_4 - e_2 - e_{3,3} = n \\ 0, otherwise \end{cases}, \forall n \in \{2,3,\dots,44\} \tag{F.6}$$

$$YC_n = \sum_{p=1}^{3} yc_{p,n}\,, \forall n \in \{2,3,\dots,44\} \tag{F.7}$$

$$YB_n = \begin{cases} 1, e_4 = n \\ 0, otherwise \end{cases}, \forall n \in \{2,3,\dots,44\} \tag{F.8}$$

Furthermore, to keep three alternating reactive trays within two feeds, to keep the feeds and the boil-up locations within the column, and to have one boil up and one feed stage per feed stream only, constraints in Eq. (F.9) are imposed over the *external variable's* domain. Finally, dependent binary variables yp_n and ye_n can be calculated in terms of independent binary variables. Thus, *external variables* are enough to represent the discrete nature of the distillation superstructure, i.e., $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{e})$.

$$E = \left\{ \begin{array}{c} 2 - e_1 \leq 0 \\ -e_2 \leq 0 \\ 1 - e_{3,2} \leq 0 \\ \boldsymbol{e} \in \mathbb{Z}^6 \colon e_1 + e_{3,1} + e_{3,2} + e_{3,3} - e_4 + e_2 + 1 \leq 0 \\ -e_{3,1} \leq 0 \\ -e_{3,3} \leq 0 \\ e_4 - 44 \leq 0 \end{array} \right\} \tag{F.9}$$

F.4 Mathematical formulation (case study)

This section presents the main equations used for the mathematical formulation of the Ethyl Tert Butyl Ether (ETBE) case study presented in chapter 6. Tables F.1-F.5 summarize the symbols used to define these equations. The reader is referred to previous works where this model mas proposed for further details [12,343].

Table F.1. Sets and superstructure size parameters.

Symbol	Definition
I	Set of components
N	Set of stages
F	Set of feeds
$COND$	Condenser stage set
REB	Reboiler stage set
INT	Internal stages set
FT	Number of feed stages of the superstructure
KT	Number of catalytic stages of the superstructure
NT	Total number of stages of the superstructure
CT	Total number of components

Table F.2. Binary variables ($\boldsymbol{y}$) and parameters

Symbol	Definition
YC_n	Existence of catalyst at stage *n*
YR_n	Existence of reflux at stage *n*
YB_n	Existence of boil-up at stage *n*
$YF_{n,f}$	Existence of feed f at stage *n*
YP_n	Binary term: Real or leftover tray
YE_n	Binary term: Interacting or non-interacting phases

Table F.3. Real variables ($[\boldsymbol{\varphi}, \boldsymbol{\chi}(\boldsymbol{t}), \boldsymbol{\zeta}(\boldsymbol{t})]$) and parameters

Symbol	Definition
A_h	Hole area $[m^2]$
A_T	Total area $[m^2]$
A_{DC}	Downcomer area $[m^2]$
D_c	Column diameter $[m]$
d_h	Hole diameter $[m]$

$F_{factor,n}$	Vapor Flow factor $[Pa^{0.5}]$
h_s	Stage height $[m]$
h_w	weir height $[m]$
h_{pack}	Packing height $[m]$
H_T	Total column height $[m]$
H_n^L	Liquid enthalpy of stage *n* $[kJ/mol]$
H_n^V	Vapor enthalpy of stage *n* $[kJ/mol]$
H_f^{IN}	Enthalpy of feed *f* $[kJ/mol]$
IN_f	Feed flow of component *f* $[mol/min]$
$k_{rate,n}$	Rate constant at stage *n* $[mol/kg]$
K_{Eq}	ETBE reaction equilibrium constant
k_A	Ethanol adsorption equilibrium constant
$K_{n,i}$	Equilibrium constant of component *i* at stage *n*
$L_{load,n}$	Liquid load through the reactive section n $[m/s]$
L_n	Liquid flow of stage *n* $[mol/min]$
LR_n	Reflux feed flow at stage *n* $[mol/min]$
$m_{cat,n}$	Mass of catalyst per stage *n* $[kg]$
M_n	Total accumulation at stage n $[mol]$
$M_{n,i}$	Partial accumulation of component i at stage n $[mol]$
N_{int}	Number of internal stages
P_n	Pressure at stage *n* $[bar]$
P_{op}	Operating pressure at the top $[bar]$
ΔP_n	Pressure drop at stage *n* $[bar]$
$\Delta P_{C,n}$	Pressure drop due to the catalyst packing $[bar]$
$\Delta P_{\sigma,n}$	Pressure drop due to superficial tension at stage *n* $[bar]$
$\Delta P_{D,n}$	Pressure drop due to dry air at stage *n* $[bar]$
$\Delta P_{L,n}$	Pressure drop due to the liquid $[bar]$
$P_{ow,n}$	Pressure drop of the liquid over the tray $[bar]$
Q_{Cond}	Condenser duty $[kJ/mol]$

Q_{Reb}	Reboiler duty $[kJ/mol]$
$\mathcal{R}_n$	Reaction rate at stage *n* $[mol/kg]$
RB	Boil-up ratio
RR	Reflux ratio
$u_{L,n}$	Liquid velocity $[m/s]$
$u_{V,n}$	Vapor velocity $[m/s]$
U_n	Internal energy at stage n $[kJ]$
V_n	Vapor flow of stage *n* $[mol/min]$
VB_n	Boil-up feed flow at stage *n* $[mol/min]$
$x_{n,i}$	Liquid molar composition at stage *n* for component *i*
$y_{n,i}$	Vapor molar composition at stag *n* for component *i*
$z_{i,f}$	Molar composition of component *i* in feed f
β	Angle for downcomer size determination
$\gamma_{n,i}$	Activity coefficient of component *i* at stage *n*
v_i	Stoichiometric coefficient of component *i*
ρ_{cat}	Catalyst density $[kg/m^3]$
$\varphi_{n,i}$	Fugacity coefficient of component *i* at stage *n*
ϕ_c	Catalyst volume fraction

Table F.4. Set subindices

Symbol	Definition
i	Index of set I
f	Index of set F
n	Index of sets $N, COND, REB, INT$

Table F.5. Superscripts

Symbol	Definition
IN	Inlet
L	Liquid
V	Vapor
sat	Referring to saturation

F.4.1 Sets

The sets used in the formulation are:

Set of components (index i)

$$I = \{1, \dots, CT\} = \{Isobutene, Ethanol, nbutene, ETBE\} \quad \text{(F.10)}$$

Set of units in the superstructure (index n)

$$N = \{1, \dots, NT\} = \{1,2, \dots ,45\} \quad \text{(F.11)}$$

Set of feeds (index f)

$$F = \{1, \dots, FT\} = \{1: ethanol, 2: butenes\} \quad \text{(F.12)}$$

Condenser stage

$$COND = \{1\} \subseteq N \quad \text{(F.13)}$$

Reboiler stage

$$REB = \{NT\} \subseteq N \quad \text{(F.14)}$$

Internal stages of the superstructure

$$INT = N \backslash REB \backslash COND \quad \text{(F.15)}$$

F.4.2 Structural decisions

Independent binary variables used to represent structural decisions are shown below. YR_n is a parameter fixed at $YR_2 = 1$ and $YR_n = 0, \forall n \neq 2$.

Existence of catalyst at unit n.

$$YC_n \in \{0,1\}, \forall n \in INT \quad \text{(F.16)}$$

If $YC_n = 1 \Longrightarrow$ *Unit* n *is catalytic.*

Existence of reflux at unit n

$$YR_n \in \{0,1\}, \forall n \in INT \quad \text{(F.17)}$$

If $YR_n = 1 \Longrightarrow$ *There is reflux at unit* n.

Existence of boil-up at unit n

$$YB_n \in \{0,1\}, \forall n \in INT \quad \text{(F.18)}$$

If $YB_n = 1 \Longrightarrow$ *There is boil-up at unit* n.

Existence of feed f at unit n

$$YF_{n,f} \in \{0,1\}, \forall n \in INT, \forall f \in F \quad \text{(F.19)}$$

If $YF_{n,f} = 1 \Longrightarrow$ *There is feed at unit* n.

To define if a unit will be part of the final design and will be in equilibrium, the following constraints are enforced to define dependent binary variables:

$$YP_n = 1 - \left(1 - \sum_{j=2}^{n} YR_j\right) - \left(\sum_{j=2}^{n} YB_j - YB_n\right), \forall n \in INT \quad \text{(F.20)}$$

If $YP_n = 1 \Longrightarrow$ *Unit* n *is a real tray.*

$$YE_n = YP_n, \forall n \in INT \quad \text{(F.21)}$$

If $YE_n = 1 \Longrightarrow$ *There is vapor-liquid equilibrium in tray* n.

F.4.3 Logical constraints over discrete design variables

These constraints represent the feasible region Y. The following logical constraints are required to obtain a realistic design that distinguishes between rectification, reaction and stripping zones:

The boil-up stage is below the reflux stage

$$\sum_{j=2}^{n} YR_j \geq YB_n, \forall n \in INT \quad \text{(F.22)}$$

There is one reflux stage

$$\sum_{j \in INT} YR_j = 1 \quad \text{(F.23)}$$

There is one boil-up stage

$$\sum_{j \in INT} YB_j = 1 \quad \text{(F.24)}$$

There is one feed stage per feed stream

$$\sum_{j \in INT} YF_{j,f} = 1, \forall f \in F \quad \text{(F.25)}$$

The feed stages are below the reflux stage

$$\sum_{j=2}^{n} YR_j \geq YF_{n,f}, \forall f \in F, \forall n \in INT \quad \text{(F.26)}$$

The boil-up stage is below the feed stages

$$\sum_{j=2}^{n} YF_{j,f} \geq YB_n, \forall f \in F, \forall n \in INT \quad \text{(F.27)}$$

The catalytic stages are below the reflux stage

$$\sum_{j=2}^{n} YR_j \geq YC_n, \forall n \in INT \quad \text{(F.28)}$$

The catalytic stages are above the boil-up stage

$$\sum_{j=2}^{n} YB_j - YB_n \leq 1 - YC_n, \forall n \in INT \quad \text{(F.29)}$$

There is a fixed number of catalytic stages

$$\sum_{j \in INT} YC_j = KT \quad \text{(F.30)}$$

The ethanol feed is above the butenes feed

$$\sum_{j=2}^{n} YF_{j,1} \geq YF_{j,2}, \forall n \in INT \quad \text{(F.31)}$$

The catalytic stages are below the ethanol feed stage

$$\sum_{j=2}^{n} YF_{j,1} \geq YC_n, \forall n \in INT \quad \text{(F.32)}$$

The catalytic stages are above the butenes feed stage

$$\sum_{j=2}^{n} YF_{j,2} - YF_{n,2} \leq 1 - YC_n, \forall n \in INT \quad \text{(F.33)}$$

F.4.4 Constraints over discrete and continuous variables

These constraints represent vector $\boldsymbol{g}^D$. These include:

Definition of active area

$$A_a = A_T - 2A_{DC} \tag{F.34}$$

Definition of downcomer area

$$A_{DC} = 0.5\left(\frac{D_c}{2}\right)^2 (\beta - \sin(\beta)) \tag{F.35}$$

Definition of weir length

$$L_W = 0.8D_C \tag{F.36}$$

Definition of hole area

$$A_h = 0.907A_a\left[\frac{d_h}{pitch}\right]^2 \tag{F.37}$$

Total column height

$$H_T = 1.15\sum_{n\in N} YP_n h_S \tag{F.38}$$

Inequalities between weir height and stage height

$$\frac{h_S}{20} \leq h_w \leq \frac{h_S}{3} \tag{F.39}$$

Available catalyst space

$$\frac{m_{cat,n}}{\rho_{cat}} \leq YC_n\phi_c\left(\frac{\pi}{4}\right)D_c^2 h_{pack}, \forall n \in INT \tag{F.40}$$

Length to diameter ratio

$$H_T \leq 20D_c \tag{F.41}$$

F.4.5 Constraints at steady state

Constraints at steady state in $\boldsymbol{h}^{SS}$ that were not defined in chapter 6 are defined in this section. These include mass and energy balances and tray capacity limits at steady state. The reader is referred to previous works to see how enthalpies were calculated[343]. The constraints described in this section are valid at $t_k, \forall k \in K$. Condenser equations are defined as:

Total mole balance (F.42)

$$0 = V_2 - V_1(1 + RR)$$

Partial mole balance

$$0 = V_2 y_{2,i} - V_1 x_{1,i}(RR + 1), \forall i \in I \quad \text{(F.43)}$$

Summation equation

$$\sum_{i \in I}\left(y_{1,i} - x_{1,i}\right) = 0 \quad \text{(F.44)}$$

Thermodynamic equilibrium

$$\left(y_{1,i} - K_{1,i} x_{1,i}\right) = 0, \forall i \in I \quad \text{(F.45)}$$

Energy balance

$$0 = V_2 H_2^V - V_1(RR + 1)H_1^L - Q_{Cond} \quad \text{(F.46)}$$

Definition of reflux flow

$$LR_n = (RR)V_1 YR_n\,, \forall n \in INT \quad \text{(F.47)}$$

$$L_1 = 0 \quad \text{(F.48)}$$

Reboiler equations are defined as:

Total mole balance

$$0 = L_{NT-1} - L_{NT}(1 + RB) \quad \text{(F.49)}$$

Partial mole balance

$$0 = L_{NT-1} x_{NT-1,i} - L_{NT}\left(x_{NT,i} + RB y_{NT,i}\right), \forall i \in I \quad \text{(F.50)}$$

Summation equation

$$\sum_{i \in I}\left(y_{NT,i} - x_{NT,i}\right) = 0 \quad \text{(F.51)}$$

Thermodynamic equilibrium

$$\left(y_{NT,i} - K_{NT,i} x_{NT,i}\right) = 0, \forall i \in I \quad \text{(F.52)}$$

Energy balance

$$0 = Q_{Reb} + L_{NT-1}H^{L}_{NT-1} - L_{NT}H^{L}_{NT} - (RB)L_{NT}H^{V}_{NT} \quad \text{(F.53)}$$

Definition of boil-up flow

$$VB_n = (RB)L_{NT}YB_n, \forall n \in INT \quad \text{(F.54)}$$

$$V_{NT} = 0 \quad \text{(F.55)}$$

Equations for internal trays are defined as:

Total mole balance

$$0 = \sum_{f\in F} YF_{n,f}IN_f + LR_n + VB_n + L_{n-1} + V_{n+1} - L_n - V_n + \left(m_{cat,n}\mathcal{R}_n \sum_{i\in C} v_i\right)YC_n, \forall n \in INT \quad \text{(F.56)}$$

Partial mole balance

$$0 = \sum_{f\in F} YF_{n,f}IN_f z_{i,f} + LR_n x_{1,i} + VB_n y_{NT,i} + L_{n-1}x_{n-1,i} + V_{n+1}y_{n+1,i} - L_n x_{n,i} - V_n y_{n,i} + \left(m_{cat,n}\mathcal{R}_n v_i\right)YC_n, \forall n \in INT, \forall i \in I \quad \text{(F.57)}$$

Summation equation

$$\sum_{i\in I}\left(y_{n,i} - x_{n,i}\right) = 0, \forall n \in INT \quad \text{(F.58)}$$

Thermodynamic equilibrium

$$YE_n\left(y_{n,i} - K_{n,i}x_{n,i}\right) = 0, \forall i \in I, \forall n \in INT \quad \text{(F.59)}$$

Input-output operation

$$\left(\sum_{j=2}^{n} YR_j\right)(1 - YE_n)\left(y_{n,i} - y_{n+1,i}\right) = 0, \forall i \in I\backslash\{1\}, \forall n \in INT \quad \text{(F.60)}$$

$$\left(1 - \sum_{j=2}^{n} YR_j\right)(1 - YE_n)\left(x_{n,i} - x_{n-1,i}\right) = 0, \forall i \in I\backslash\{1\}, \forall n \in INT \quad \text{(F.61)}$$

$$(1 - YE_n)(V_n - V_{n+1} - VB_n) = 0, \forall n \in INT \quad \text{(F.62)}$$

Energy balance

$$0 = \sum_{f \in F} YF_{n,f} IN_f H_f^{IN} + LR_n H_1^L + VB_n H_{NT}^V + L_{n-1} H_{n-1}^L + V_{n+1} H_{n+1}^V - L_n H_n^L - V_n H_n^V, \forall n \in INT \quad \text{(F.63)}$$

Hydrodynamic inequality constraints are defined as follows (the reader is referred to previous works to see how velocities and pressure drops were calculated [343]):

Maximum vapor velocity

$$YP_n\left(u_{V,n} - u_{V,n}^{max}\right) \leq 0, \forall n \in INT \quad \text{(F.64)}$$

Maximum liquid velocity

$$YP_n\left(u_{L,n} - u_{L,n}^{max}\right) \leq 0, \forall n \in INT \quad \text{(F.65)}$$

Pressure drop of the liquid

$$YP_n\left(P_{ow,n} - P_{ow,n}^{max}\right) \leq 0, \forall n \in INT \quad \text{(F.66)}$$

Minimum vapor velocity through the holes

$$YP_n(u_{h,n}^{min} - u_{h,n}) \leq 0, \forall n \in INT \quad \text{(F.67)}$$

Maximum pressure drop of the catalyst

$$YC_n\left(\Delta P_{C,n} - \Delta P_{C,n}^{max}\right) \leq 0, \forall n \in INT \quad \text{(F.68)}$$

F.4.6 Constraints during dynamic transitions

Constraints at transient state in $\boldsymbol{h}^{TR}$ that were not defined in chapter 6 are defined in this section. These include mass balances and energy balances. The reader is referred to previous works to see how enthalpies were calculated, and to see the definitional equations of accumulation terms [343]. The constraints described in this section are valid over $\left(t_{k-1}, t_k^{TR}\right], \forall k \in K$.

Condenser equations are defined as:

Total mole balance

$$0 = V_2 - V_1(1 + RR) \quad \text{(F.69)}$$

Partial mole balance

$$\frac{dM_{1,i}}{dt} = V_2 y_{2,i} - V_1 x_{1,i}(RR+1), \forall i \in I \quad \text{(F.70)}$$

Summation equation

$$\sum_{i \in I}\left(y_{1,i} - x_{1,i}\right) = 0 \quad \text{(F.71)}$$

Thermodynamic equilibrium

$$\left(y_{1,i} - K_{1,i} x_{1,i}\right) = 0, \forall i \in I \quad \text{(F.72)}$$

Energy balance

$$\frac{dU_1}{dt} = V_2 H_2^V - V_1(RR+1)H_1^L - Q_{Cond} \quad \text{(F.73)}$$

Definition of reflux flow

$$LR_n = (RR)V_1 YR_n\,, \forall n \in INT \quad \text{(F.74)}$$

$$L_1 = 0 \quad \text{(F.75)}$$

Reboiler equations are defined as:

Total mole balance

$$0 = L_{NT-1} - L_{NT}(1+RB) \quad \text{(F.76)}$$

Partial mole balance

$$\frac{dM_{NT,i}}{dt} = L_{NT-1} x_{NT-1,i} - L_{NT}\left(x_{NT,i} + RB y_{NT,i}\right), \forall i \in I \quad \text{(F.77)}$$

Summation equation

$$\sum_{i \in I}\left(y_{NT,i} - x_{NT,i}\right) = 0 \quad \text{(F.78)}$$

Thermodynamic equilibrium

$$\left(y_{NT,i} - K_{NT,i} x_{NT,i}\right) = 0, \forall i \in I \quad \text{(F.79)}$$

Energy balance

$$\frac{dU_{NT}}{dt} = Q_{Reb} + L_{NT-1} H_{NT-1}^L - L_{NT} H_{NT}^L - (RB) L_{NT} H_{NT}^V \quad \text{(F.80)}$$

Definition of boil-up flow

$$VB_n = (RB)L_{NT}YB_n, \forall n \in INT \quad \text{(F.81)}$$

$$V_{NT} = 0 \quad \text{(F.82)}$$

Equations for internal trays are defined as:

Total mole balance

$$YP_n \frac{dM_n}{dt} = \sum_{f\in F} YF_{n,f} IN_f + LR_n + VB_n + L_{n-1} + V_{n+1} - L_n - V_n + \left(m_{cat,n}\mathcal{R}_n \sum_{i\in C} v_i\right)YC_n, \forall n \in INT \quad \text{(F.83)}$$

Partial mole balance

$$YP_n \frac{dM_{n,i}}{dt} = \sum_{f\in F} YF_{n,f} IN_f z_{i,f} + LR_n x_{1,i} + VB_n y_{NT,i} + L_{n-1}x_{n-1,i} + V_{n+1}y_{n+1,i} - L_n x_{n,i} - V_n y_{n,i} + \left(m_{cat,n}\mathcal{R}_n v_i\right)YC_n, \forall n \in INT, \forall i \in I \quad \text{(F.84)}$$

Summation equation

$$\sum_{i\in I}\left(y_{n,i} - x_{n,i}\right) = 0, \forall n \in INT \quad \text{(F.85)}$$

Thermodynamic equilibrium

$$YE_n\left(y_{n,i} - K_{n,i}x_{n,i}\right) = 0, \forall i \in I, \forall n \in INT \quad \text{(F.86)}$$

Input-output operation

$$\left(\sum_{j=2}^{n} YR_j\right)(1 - YE_n)\left(y_{n,i} - y_{n+1,i}\right) = 0, \forall i \in I\backslash\{1\}, \forall n \in INT \quad \text{(F.87)}$$

$$\left(1 - \sum_{j=2}^{n} YR_j\right)(1 - YE_n)\left(x_{n,i} - x_{n-1,i}\right) = 0, \forall i \in I\backslash\{1\}, \forall n \in INT \quad \text{(F.88)}$$

$$(1 - YE_n)(V_n - V_{n+1} - VB_n) = 0, \forall n \in INT \quad \text{(F.89)}$$

Energy balance

$$YP_n \frac{dU_n}{dt} = \sum_{f\in F} YF_{n,f} IN_f H_f^{IN} + LR_n H_1^L + VB_n H_{NT}^V + L_{n-1}H_{n-1}^L + V_{n+1}H_{n+1}^V - L_n H_n^L - V_n H_n^V, \forall n \in INT \quad \text{(F.90)}$$

F.4.7 Intermediate expressions

This section contains information about the main definitional equations used in the formulation of the problem. The reader is referred to previous works for more detailed descriptions [343]. These expressions appear both in $\boldsymbol{h}^{SS}$ and $\boldsymbol{h}^{TR}$, because they are valid over the $\left(t_{k-1}, t_k^{TR}\right] \cup \{t_k\}, \forall \in K$.

Pressure drop.

$$P_1 = P_{op} \tag{F.91}$$

$$P_{NT} = P_{NT-1} \tag{F.92}$$

$$P_n = \ (P_{n-1} + YP_n \Delta P_n), \forall n \ \in INT \tag{F.93}$$

$$\Delta P_n = \Delta P_{\sigma,n} + \Delta P_{D,n} + \Delta P_{L,n} + YC_n \Delta P_{C,n}, \forall n \ \in INT \tag{F.94}$$

K-value with the gamma-phi formulation (F.95)

$$K_{n,i} = \frac{P_i^{sat} \gamma_{n,i}}{P_n \varphi_{n,i}}, \forall i \in I, \forall n \in N$$

LHHW model

$$\mathcal{R}_n = \frac{(Kinetic\ Factor_n)(Driving\ Force_n)}{(Adsorption\ group_n)^3}, \forall n \ \in INT \tag{F.96}$$

$$Kinetic\ Factor_n = k_{rate,n}\left(\gamma_{n,Ethanol} x_{n,Ethanol}\right)^2, \forall n \ \in INT \tag{F.97}$$

$$Driving\ Force_n = \left(\gamma_{n,Isobutene} x_{n,Isobutene} - \frac{\gamma_{n,ETBE} x_{n,ETBE}}{K_{Eq}\left(\gamma_{n,Ethanol} x_{n,Ethanol}\right)}\right), \forall n \ \in INT \tag{F.98}$$

$$Adsorption\ group_n = 1 + k_A\left(\gamma_{n,Ethanol} x_{n,Ethanol}\right), \forall n \ \in INT \tag{F.99}$$

F.5 Additional formulation details (case study)

In this section, constraints that only apply for this case study are presented. The first requirement is that the ETBE composition ($v_{product}(t) = x_{ETBE}(t)$) must satisfy its requirements at steady state, as

described by Eq. (F.100). In addition, the case study is constrained such that the steady state attained by the system when operating at $x_{ETBE} = 0.83$ is always the same (Eqs. (F.102) and (F.103)).

$$x_{ETBE}(t) = sp_{product}(t), \forall t \in \{t_1, t_2, t_3, t_4\},\ sp_{product}(t) = \begin{cases} 83\%, t \in (t_0, t_1] \\ 95\%, t \in (t_1, t_2] \\ 83\%, t \in (t_3, t_3] \\ 63\%, t = (t_4, t_4] \end{cases} \quad \text{(F.100)}$$

$$\boldsymbol{\mu}(t_1) = \boldsymbol{\mu}(t_3), with\ \boldsymbol{\mu}(t) = [Q_r(t), RR(t)] \quad \text{(F.102)}$$

$$[\boldsymbol{\chi}(t_1), \boldsymbol{\zeta}(t_1)] = [\boldsymbol{\chi}(t_3), \boldsymbol{\zeta}(t_3)] \quad \text{(F.103)}$$

Another requirement to set up this case study is to select appropriate upper and lower bounds for the input variables. In this case study, these inequality constraints are defined by the feasible hydrodynamic operation region, which enforces steady states to avoid undesired flow regimes that generate weeping, entrainment flooding, downflow flooding and catalyst flooding. During real operation hydrodynamic constraints are never completely absent and the limits imposed only avoid excessive undesired conditions [385]. Moreover, these operating conditions are predominantly accepted during dynamic operation, e.g., the weeping phenomenon is a normal situation during the start-up of a column [386–388]. Consequently, hydrodynamic constraints were only imposed at steady state operation. Nevertheless, in the work by Medina-Herrera et al. [28] it was demonstrated that under simple inputs to the process, e.g. ramp responses, the desired steady states calculated from a steady-state *multi-scenario* approach can be reached, although there may be constraint violations in the inequality constraints that try to keep the response within a desired operation range during transitions. Thus, to minimize the effect of these abnormal operating conditions during transitions, the bounds on the operating variables (reboiler duty and reflux ratio) were selected as the maximum and minimum values that maintain the column within satisfactory hydrodynamic operating limits at steady state, for each fixed combination of discrete variables. The optimization problems used to compute the bounds that help to avoid excessive hydrodynamic violations during operation were embedded into the problem's constraints. An upper bound for the reflux ratio was not always dictated by the hydrodynamic constraints, thus, the maximum reflux ratio was selected as 10, to avoid abrupt changes in this manipulated variable (Eqs. (F.104) and (F.105)).

$$Q_r^{min}(\boldsymbol{e}, \boldsymbol{\varphi}) \leq Q_r(t) \leq Q_r^{max}(\boldsymbol{e}, \boldsymbol{\varphi}), \forall t \in \left(t_{k-1}, t_k^{TR}\right], \forall\ k \in K \quad \text{(F.104)}$$

$$RR^{min}(\boldsymbol{e}, \boldsymbol{\varphi}) \leq RR(t) \leq \min\{10, RR^{max}(\boldsymbol{e}, \boldsymbol{\varphi})\}, \forall t \in \left(t_{k-1}, t_k^{TR}\right], \forall\ k \in K \quad \text{(F.105)}$$

Appendix G
Supplementary Material for Chapter 7

G.1 Initialization strategies

Two approaches that implement sequential scheduling and dynamic optimization strategies are considered in this work to initialize the D-SDA execution. The first initialization approach assumes that the batch size $b_{i,j,t}$ processed by task-unit pairs (i,j) with process dynamics remains fixed at its maximum value. The underlying logic of this assumption is that if task-unit pair (i,j) can operate at its maximum capacity, then lower capacities may also be feasible. Under this assumption, the first initialization approach follows two steps proposed by Chu and You[61]: 1) find the optimal control actions of the different dynamic tasks involved by minimizing their processing time through NLP optimization and 2) fix the processing times and the control operating policies and optimize the schedule of the batch plant through MIP optimization. As noted by Chu and You[61], the first step is expected to have a positive impact on the profit when considering an economic objective function, given that more time spots become available in the second step to execute tasks and encourage production when processing times are short. Note that I refer to equations that appear in chapter 7 through this section.

A second initialization approach that is suitable for discrete-time SSDO formulations is proposed in this work by considering the external variables reformulation of $Y_{\tau_{i,j},i,j} = Y_{\tau_{i,j},i,j}(e_{i,j})$. Recall that external variables $e_{i,j}$ can be interpreted as the upper bound imposed over processing times, i.e., combining Eqs. 7.1H, 7.4A and 7.4B results in $\tau^{P}_{i,j,t} \leqq \min\{\hat{\delta} e_{i,j}, \tau^{P,max}_{i,j}\}$. Hence, the proposed initialization strategy iteratively increases the values of $e_{i,j}$ until feasibility is attained, i.e., it is assumed that infeasibilities are a result of wrongly selected external variables $e_{i,j}$ related to variable processing times. The iterative procedure starts by fixing external variables $e_{i,j}$ at their lower bound, which is expected to give flexibility to the scheduling constraints of the problem, i.e., if discrete processing times are at their minimum, more time points become available to accommodate tasks within the scheduling horizon (See step A in Table G.1). Then, Step B of the iterative procedure is performed, which consists of solving a MIP scheduling problem that keeps $Y_{\tau_{i,j},i,j} = Y_{\tau_{i,j},i,j}(e_{i,j})$ fixed, and ignores process dynamics, and any nonlinear term in the objective function coming from nonlinear process dynamic models, e.g., integrals of control actions over time. After that, Step C fixes the scheduling variables

obtained from the previous step and optimizes process dynamic models to obtain optimal dynamic variables $\boldsymbol{v}_{i,j,t}^{dyn}$, e.g., task durations in $\boldsymbol{\varphi}_{i,j,t}$ and control actions $\boldsymbol{\mu}_{i,j,t}$. The stopping criteria verification is performed in Step D as follows. If both steps B and C return a solution that is feasible according to eqs 7.3B-7.3D, algorithm in Table G.1 returns this feasible solution. Alternatively, if step B is infeasible or all the values of external variables $e_{i,j}$ have attained their maximum, then iterations stop indicating that no feasible solution was identified. Else, if step B remains feasible but step C is infeasible, then the value of $e_{i,j}$ is increased by one for those pairs (i,j) that are generating infeasibility in the dynamic constraints, e.g., more time may be needed to satisfy the product quality requirements of one of the pairs (i,j). Note that this step assumes that the dynamic models of pairs $(i,j) \in IJ^{\times,dyn}$ are independent, which allows to examine them separately. Then, a new iteration begins with Step B.

Table G.1. Iterative-sequential approach to initialize the D-SDA

Step A. Initialization

Fix $e_{i,j}$ at $\left\lceil \tau_{i,j}^{P,min}/\hat{\delta} \right\rceil, \forall (i,j) \in IJ^{\times}$.

Step B. MIP scheduling optimization

Fix $Y_{\tau_{i,j},i,j}, \forall (i,j) \in IJ^{\times}, \forall \tau_{i,j} \in \mathrm{T}_{i,j}$ in Eq. 7.3B based on the current values of $e_{i,j}, \forall (i,j) \in IJ^{\times}$ through Eq. 7.4B. Then, solve the MIP problem given by Eqs. 7.3A, 7.3B and 7.3D, ignoring the nonlinear dynamic terms in Eq. 7.3A.

Step C. NLP dynamic optimization

Fix the scheduling variables $\boldsymbol{v}_{\tau_{i,j},i,j,m,t}^{sch}$ found in the previous step in Eqs. 7.3A, 7.3B and 7.3D. Then, solve the NLP problem given by Eqs. 7.3A, 7.3C and 7.3D, using the unmodified objective function in Eq. 7.3A.

Step D. Stopping conditions

If Steps B and C returned a feasible solution: stop, a feasible solution to the problem in Eqs. 7.3A-7.3D has been detected.

Else if step B is infeasible or $e_{i,j} \geq \left\lceil \tau_{i,j}^{P,max}/\hat{\delta} \right\rceil, \forall (i,j) \in IJ^{\times}$: stop, no feasible solution to the problem in Eqs. 7.3A-7.3D was detected.

Else: for every $(i,j) \in IJ^{\times,dyn}$, solve a feasibility problem involving the constraints in Eq. 7.3C that define the dynamic model of pair (i,j). Then, update $e_{i,j} \leftarrow e_{i,j} + 1$ for every pair $(i,j) \in IJ^{\times,dyn}$ whose feasibility problem resulted infeasible and go to Step B.

G.2 Supplementary tables for case study 1

This section presents Tables G.2-G.6, showing additional parameters required to define the first case study presented in section 7.3.1 of chapter 7.

Table G.2. Maximum and minimum capacity of the units (case study 1)

Unit $j \in J$	b_j^{min} $[m^3]$	b_j^{max} $[m^3]$
$U1$	1	10
$U2$	1	5
$U3$	1	8
$U4$	1	20

Table G.3. Parameters related to states $m \in M$ (case study 1)

State $m \in M$	s_m^{init} $[m^3]$	s_m^{max} $[m^3]$	c_m^{mat} $[\$/m^3]$	c_m^{out} $[\$/m^3]$
$S1$	400	400	50	-
$S2$	400	400	150	-
$S3$	400	400	200	-
$S4$	0	100	-	-
$S5$	0	15	-	-
$S6$	0	50	-	-
$S7$	0	100	-	-
$S8$	0	400	-	300
$S9$	0	400	-	400

Table G.4. Parameters related to pairs $(i,j) \in IJ^{\times}$ (case study 1)

Pair $(i,j) \in IJ^{\times}$	$\tau_{i,j}^{P,min}$ $[h]$	$\tau_{i,j}^{P,max}$ $[h]$	$\tau_{i,j}^{P,fixed}$ $[h]$	$c_{i,j}^{fix}$ $[\$]$	$c_{i,j}^{var_1}$ $[\$/m^3]$
$(T1,U1)$	0.5	0.5	0.5	10	10
$(T2,U2)$	0.1	2	-	15	-
$(T2,U3)$	0.1	2	-	30	-
$(T3,U2)$	1	1	1	5	20
$(T3,U3)$	2.5	2.5	2.5	25	30
$(T4,U2)$	1	1	1	5	20
$(T4,U3)$	5	5	5	20	35
$(T5,U4)$	1.5	1.5	1.5	20	10

Table G.5. Parameters in the dynamic models (case study 1)

Parameter	Value	Description
$k_1\ [m^3/kmol\ h]$	4	Pre-exponential factor of reaction 1
$k_2\ [m^3/h]$	0.8	Pre-exponential factor of reaction 2
$Ea_1\ [kJ/kmol]$	6000	Activation energy of reaction 1
$Ea_2\ [kJ/kmol]$	20000	Activation energy of reaction 2
$\Delta H_1\ [kJ/kmol]$	-30000	Enthalpy of reaction 1
$\Delta H_2\ [kJ/kmol]$	-10000	Enthalpy of reaction 2
$C_{IN}^{A}\ [kmol/m^3]$	10.62	Concentration of reactant A coming from state $S2$
$C_{IN}^{B}\ [kmol/m^3]$	20	Concentration of reactant B coming from state $S3$
$C_{DES}^{C}\ [kmol/m^3]$	4	Desired concentration of C in state $S5$
$V_0\ [m^3]$	1	Initial volume of the semi-batch reactor
$T_{R0}\ [K]$	293.15	Initial reactor temperature
$T_H\ [K]$	323.15	Temperature of heating fluid
$T_C\ [K]$	293.15	Temperature of cooling fluid
$R\ [kJ/kmol\ K]$	8.31	Ideal gas constant
$C_{PJ}\ [kJ/kmol\ K]$	75	Heat capacity of reaction mixture
$VJ_{U2}\ [m^3]$	0.3	Volume (jacket side, reactor $U2$)
$VJ_{U3}\ [m^3]$	0.5	Volume (jacket side, reactor $U3$)
$\rho_J\ [kg/m^3]$	1000	Density (jacked side)
$C_{PJ}\ [kJ/kg\ K]$	4.2	Heat capacity (jacket side)
$T_{J0}\ [K]$	293.15	Initial temperature (jacket side)
$Ua_{U2}\ [kJ/h\ K]$	30000	Heat transfer coefficient (reactor $U2$)
$Ua_{U3}\ [kJ/h\ K]$	20000	Heat transfer coefficient (reactor $U3$)

Table G.6. Variables in the dynamic models (case study 1)

Variable	Bounds	Description
$\chi_{i,j,t}^{A}(\psi)\ [kmol/m^3]$	[0,100]	Concentration of A
$\chi_{i,j,t}^{B}(\psi)\ [kmol/m^3]$	[0,100]	Concentration of B
$\chi_{i,j,t}^{C}(\psi)\ [kmol/m^3]$	[0,100]	Concentration of C
$\chi_{i,j,t}^{Vol}(\psi)\ [m^3]$	$[b_j^{min}, b_j^{max}]$	Volume of reaction mixture
$\chi_{i,j,t}^{TR}(\psi)\ [K]$	[293.15,323.15]	Reactor temperature
$\chi_{i,j,t}^{TJ}(\psi)\ [K]$	[293.15,323.15]	Jacket temperature
$\zeta_{i,j,t}^{A}(\psi)\ [kmol/m^3\ h]$	*	Reaction rate of A in reaction 1
$\zeta_{i,j,t}^{C}(\psi)\ [kmol/m^3\ h]$	*	Reaction rate of A in reaction 2
$\mu_{i,j,t}^{B}(\psi)\ [m^3/h]$	[0,5]	Feed rate of B
$\mu_{i,U2,t}^{hot}(\psi)\ [m^3/h]$	[0,8]	Flow of heating fluid (reactor $U2$)
$\mu_{i,U2,t}^{cold}(\psi)\ [m^3/h]$	[0,8]	Flow of cooling fluid (reactor $U2$)
$\mu_{i,U3,t}^{hot}(\psi)\ [m^3/h]$	[0,10]	Flow of heating fluid (reactor $U3$)
$\mu_{i,U3,t}^{cold}(\psi)\ [m^3/h]$	[0,10]	Flow of cooling fluid (reactor $U3$)
$\varphi_{i,j,t}^{P}\ [h]$	$[0, \tau_{i,j}^{P,max}]$	Reaction time
$\varphi_{i,j,t}^{A0}\ [kmol/m^3]$	$[0, C_{IN}^{A}]$	Initial concentration of A
$\varphi_{i,j,t}^{B0}\ [kmol/m^3]$	$[0, C_{IN}^{B}]$	Initial concentration of B

*: Defined based on the bounds imposed over other variables

The domain of the ordered discrete decisions in master problem (7.5A) is summarized in Table G.7. The domain of variables $n_{i,j}$ represent the possible number of times task i can be executed in unit j, while the domain of variable $e_{i,j}$ represents the possible number of discrete points needed to execute task i in unit j. In this case, the reformulation with external variables $e_{i,j}$ allowed to reformulate a problem that originally had 18 Boolean variables into a problem with 8 external variables. Note that variables $e_{i,j}$ with a single value in their domain represent those unit-task pairs with a fixed processing time; hence, all the integer variables and only two external variables ($e_{T2,U2}$ and $e_{T2,U3}$) are needed by the D-SDA.

Table G.7. Domain of ordered discrete decisions (case study 1)

Variable	Domain	Variable	Domain
$n_{T1,U1}$	{0,1, … ,7}	$e_{T1,U1}$	{2}
$n_{T2,U2}$	{0,1, … ,15}	$e_{T2,U2}$	{1,2, … ,6}
$n_{T2,U3}$	{0,1, … ,15}	$e_{T2,U3}$	{1,2, … ,6}
$n_{T3,U2}$	{0,1, … ,5}	$e_{T3,U2}$	{3}
$n_{T3,U3}$	{0,1}	$e_{T3,U3}$	{8}
$n_{T4,U2}$	{0,1, … ,5}	$e_{T4,U2}$	{3}
$n_{T4,U3}$	{0,1}	$e_{T4,U3}$	{15}
$n_{T5,U4}$	{0,1, … ,3}	$e_{T5,U4}$	{5}

G.3 Supplementary tables for case study 2

This section presents Tables G.8-G.12, showing additional parameters required to define the second case study presented in section 7.3.2 of chapter 7.

Table G.8. Maximum and minimum capacity of the units (case study 2)

Unit $j \in J$	b_j^{min} $[m^3]$	b_j^{max} $[m^3]$
Mix	0.2	2
Rl	0.15	1.5
Rs	0.1	1
Sep	0.2	2
$Pack$	0.1	1

Table G. 9. Parameters related to states $m \in M$ (case study 2)

State $m \in M$	s_m^{init} $[m^3]$	s_m^{max} $[m^3]$	c_m^{mat} $[\$/m^3]$	c_m^{out} $[\$/m^3]$
$M1$	10	10	500	-
$M2$	10	10	300	-
$M3$	10	10	400	-
$S1$	10	10	-	-
$I1$	0	2	-	-
$I2$	0	2	-	-
$I3$	0	2	-	-
$I4$	0	2	-	-
$I5$	0	5	-	-
$I6$	0	5	-	-
$W1$	0	10	-	-
$P1$	0	10	-	1400
$P2$	0	10	-	1200

Table G.10. Parameters related to pairs $(i,j) \in IJ^{\times}$ (case study 2)

Pair $(i,j) \in IJ^{\times}$	$\tau_{i,j}^{P,min}$ $[h]$	$\tau_{i,j}^{P,max}$ $[h]$	$\tau_{i,j}^{P,fixed}$ $[h]$	$c_{i,j}^{fix}$ $[\$]$	$c_{i,j}^{var_1}$ $[\$/m^3]$
(Mix, Mix)	1.5	1.5	1.5	10	30
$(R1, Rl)$	1	4	-	30	-
$(R1, Rs)$	1	4	-	20	-
$(R2, Rl)$	1	4	-	30	-
$(R2, Rs)$	1	4	-	20	-
$(R3, Rl)$	1	4	-	30	-
$(R3, Rs)$	1	4	-	20	-
(Sep, Sep)	3	3	3	100	100
$(Pack1, Pack)$	1.5	1.5	1.5	50	50
$(Pack2, Pack)$	1.5	1.5	1.5	50	50

Table G.11. Parameters in the dynamic models (case study 2)

Parameter	Value	Description
$k_{R1}\ [m^3/kmol\ h]$	10000000	Pre-exponential factor of reaction $R1$
$k_{R2}\ [m^3/kmol\ h]$	1200	Pre-exponential factor of reaction $R2$
$k_{R3}\ [m^3/kmol\ h]$	20000	Pre-exponential factor of reaction $R3$
$\widetilde{Ea}_{R1}\ [K]$	5000	Normalized activation energy of reaction $R1$
$\widetilde{Ea}_{R2}\ [K]$	2000	Normalized activation energy of reaction $R2$
$\widetilde{Ea}_{R3}\ [K]$	3000	Normalized activation energy of reaction $R3$
$\boldsymbol{C}_{R1,0}\ [kmol/m^3]$	[1,1,0]	Initial concentration of reaction $R1$
$\boldsymbol{C}_{R2,0}\ [kmol/m^3]$	[1,0.95,0]	Initial concentration of reaction $R2$
$\boldsymbol{C}_{R3,0}\ [kmol/m^3]$	[1.0125,0.9,0]	Initial concentration of reaction $R3$
$\boldsymbol{C}_{R1,DES}\ [kmol/m^3]$	[0.05,0.05,1.9]	Desired concentration of reaction $R1$
$\boldsymbol{C}_{R2,DES}\ [kmol/m^3]$	[0.1,0.05,1.8]	Desired concentration of reaction $R2$
$\boldsymbol{C}_{R3,DES}\ [kmol/m^3]$	[0.2125,0.1,0.8]	Desired concentration of reaction $R3$
$Ua_{Rs}\ [kJ/h\ K]$	20000	Heat transfer coefficient (reactor Rs)
$Ua_{Rl}\ [kJ/h\ K]$	30000	Heat transfer coefficient (reactor Rl)
$C_{P,R1}\ [kJ/kg\ K]$	3	Heat capacity of reaction mixture $R1$
$C_{P,R2}\ [kJ/kg\ K]$	3.5	Heat capacity of reaction mixture $R2$
$C_{P,R3}\ [kJ/kg\ K]$	4	Heat capacity of reaction mixture $R3$
$C_{PJ}\ [kJ/kg\ K]$	4.2	Heat capacity (jacket side)
$\rho_R\ [kg/m^3]$	1000	Density of reaction mixture
$\rho_J\ [kg/m^3]$	1000	Density (jacket side)
$\Delta H_{R1}\ [kJ/kmol]$	1000	Enthalpy of reaction $R1$
$\Delta H_{R2}\ [kJ/kmol]$	-2000	Enthalpy of reaction $R2$
$\Delta H_{R3}\ [kJ/kmol]$	5000	Enthalpy of reaction $R3$
$T_{R0}\ [K]$	300	Initial reactor temperature
$T_H\ [K]$	370	Temperature of heating fluid
$T_C\ [K]$	300	Temperature of cooling fluid
$T_{J0}\ [K]$	300	Initial temperature (jacket side)
$VJ_{Rs}\ [m^3]$	0.3	Volume (jacket side, reactor Rs)
$VJ_{Rl}\ [m^3]$	0.5	Volume (jacket side, reactor Rl)
$T_R^{max}\ [K]$	320	Maximum reactor temperature at $\tau = 1$

Table G.12. Variables in the dynamic models (case study 2)

Variable	Bounds	Description
$\boldsymbol{\chi}_{i,j}(\psi)$ $[kmol/m^3]$	[0,5]	Vector of concentrations
$\chi_{i,j}^{TR}(\psi)$ $[K]$	[295,370]	Reactor temperature
$\chi_{i,j}^{TJ}(\psi)$ $[K]$	[295,370]	Jacket temperature
$\zeta_{i,j}(\psi)$ $[kmol/m^3\ h]$	*	Reaction rates
$\mu_{i,Rs}^{hot}(\psi)$ $[m^3/h]$	[0,5]	Flow of heating fluid (reactor Rs)
$\mu_{i,Rs}^{cold}(\psi)$ $[m^3/h]$	[0,5]	Flow of cooling fluid (reactor Rs)
$\mu_{i,Rl}^{hot}(\psi)$ $[m^3/h]$	[0,10]	Flow of heating fluid (reactor Rl)
$\mu_{i,Rl}^{cold}(\psi)$ $[m^3/h]$	[0,10]	Flow of cooling fluid (reactor Rl)
$\varphi_{i,j}^{P}$ $[h]$	$[0, \tau_{i,j}^{P,max}]$	Reaction time
$\varphi_{i,j}^{Vol}$ $[m^3]$	$[b_j^{min}, b_j^{max}]$	Reaction volume

*: Defined based on the bounds imposed over other variables

The ordered discrete decisions of this problem are shown in Table G.13. In this case, the reformulation with external variables allowed to reformulate 52 Boolean variables into 10 external variables $e_{i,j}$, out of which 6 variables are used by the D-SDA to optimize the processing time of the different reactions considered in this network.

Table G.13. Domain of ordered discrete decisions (case study 2)

Variable	Domain	Variable	Domain
$n_{Mix,Mix}$	{0,1, … ,9}	$e_{Mix,Mix}$	{3}
$n_{R1,R(large)}$	{0,1, … ,14}	$e_{R1,R(large)}$	{2,3, … ,8}
$n_{R1,R(small)}$	{0,1, … ,14}	$e_{R1,R(small)}$	{2,3, … ,8}
$n_{R2,R(large)}$	{0,1, … ,14}	$e_{R2,R(large)}$	{2,3, … ,8}
$n_{R2,R(small)}$	{0,1, … ,14}	$e_{R2,R(small)}$	{2,3, … ,8}
$n_{R3,R(large)}$	{0,1, … ,14}	$e_{R3,R(large)}$	{2,3, … ,8}
$n_{R3,R(large)}$	{0,1, … ,14}	$e_{R3,R(large)}$	{2,3, … ,8}
$n_{Sep,Sep}$	{0,1, … ,4}	$e_{Sep,Sep}$	{6}
$n_{Pack1,Pack}$	{0,1, … ,9}	$e_{Pack1,Pack}$	{3}
$n_{Pack2,Pack}$	{0,1, … ,9}	$e_{Pack2,Pack}$	{3}

G.4 Illustrative MINLP example

Consider the illustrative MINLP problem shown in Eq. G.1, where e is an integer variable and z is a continuous variable.

$$\min_{e,z} z \tag{G.1}$$

$$s.t.$$

$$-e\sin(3.5\pi e) - z \leq 0$$

$$2 \leq e \leq 5, e \in \mathbb{Z}$$

$$z \in \mathbb{R}$$

The optimization problem above is a straightforward optimization problem that can be solved by performing a complete enumeration of integer variable e. Then, for each value of e, it is possible to solve the problem by minimizing z. This complete enumeration of the problem is illustrated by the red points in Figures G.1A and G.1B. If the integer requirement of variable e is relaxed and a complete enumeration is again performed, then it is possible to obtain the solid blue line in Figure G.1. This suggests that variable e makes this optimization problem non-convex, according to traditional convexity definitions, i.e., the epigraph of the solid blue line in Figure G.1A is not convex. This problem can also be analyzed from a discrete convex analysis perspective, which does not rely on the continuous relaxation of e. Instead, it defines a local convex extension of the problem, which in this one-dimensional case can be simply obtained by connecting the red points using straight lines (see the blue dashed lines in Figure G.1B). Given that the epigraph of the dashed blue line in Figure G.1B is convex, it is concluded that this optimization problem is integrally convex.

Now, consider the decomposition of this problem using Generalized Benders Decomposition (GBD). Taking variable e as the complicating variable, this decomposition results in master problem (G.2) and subproblem (G.3), for GBD. Note that the GBD variant herein considered relies on repeated complicating variables $\bar{e}$ and uses the non-linearized version of the Lagrange function, which is the variant commonly used for SSDO[61,62]. Superscript k is used to denote the iteration number, set K contains all iterations that have been performed, and λ corresponds to the optimal Lagrange multiplier of constraint $e - \bar{e}^k = 0$.

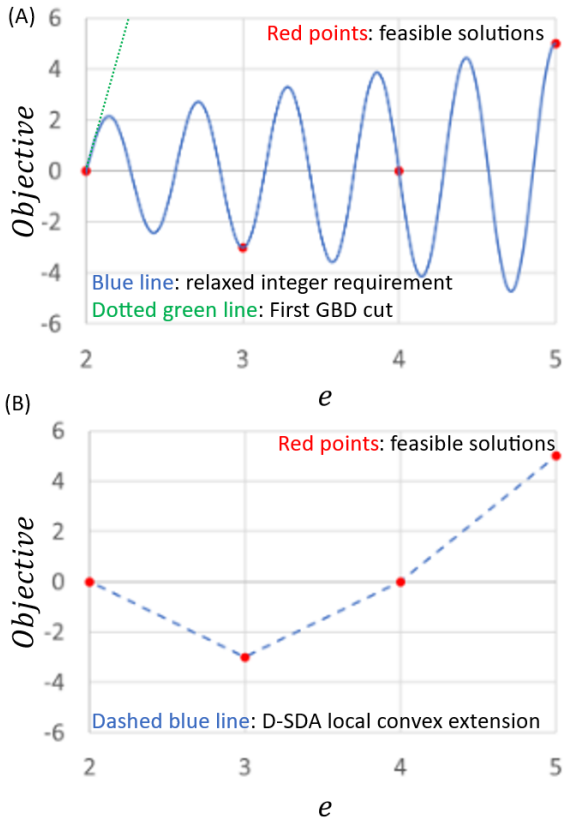


Figure G.1. A): analysis of the problem from the traditional convexity perspective, B): analysis of the problem from the discrete-convex-analysis perspective.

$$\min_{z,\bar{e}} z \tag{G.2}$$

$$s.t.$$

$$z^k + \lambda^k\left(\bar{e} - e^k\right) \leq z, \forall k \in K$$

$$2 \leq e \leq 5, e \in \mathbb{Z}$$

$$z^k = \min_{e,z} z \tag{G.3}$$

$$s.t.$$

$$-e\sin(3.5\pi e) - z \leq 0$$

$$e - \bar{e}^k = 0$$

$$z \in \mathbb{R}$$

$$e \in \mathbb{R}$$

If this problem is initialized at $\bar{e}^1 = 2$, the fisrt step in GBD is to solve the subproblem with $\bar{e}^1 = 2$ fixed. This returns the first subproblem objective function $z^1 = 0$, integer variable $e^1 = 2$ and optimal Lagrange multiplier $\lambda^1 = 21.99$. With this information, it is possible to construct the first GBD benders cut $21.99(\bar{e} - 2) \leq z$, which is added to the master problem. Then, the master problem is solved returning objective function $z = 0$, indicating that the gap between the master problem and the subproblem closes and the algorithm stops. As it is shown in Figure G.1A, GBD fails to find alternative discrete solutions to this problem because the first cut it generates (dotted green line in Figure G.1A) represents the sensitivity of the objective function when constraint $e - \bar{e}^k = 0$ is varied continuously. Nevertheless, from the solution of the problem using complete enumeration it was found that the Relaxed-MINLP version of this problem is non-convex and multimodal, hence, GBD cuts may not underestimate the feasible region of the problem and lead the algorithm towards suboptimal solutions. On the contrary, an algorithm based on discrete-convex analysis such as the D-SDA is guaranteed to find the global optimum of this problem independently of its initialization, thanks to the integrally convex characteristics of the problem.

Appendix H
Supplementary Material for Chapter 8

H.1 Reformulation of complicating variables

Ordered discrete decisions appear in vector $\boldsymbol{v}^{sch}$ and include integer variables $\boldsymbol{n} \in \mathbb{Z}^{n_n}$ and Boolean/binary variables $\boldsymbol{y}$ with ordered structures. The key idea behind MLD-BD is to improve the search strategy of ordered discrete decisions $[\boldsymbol{n}, \boldsymbol{y}]$, by considering them as complicating variables. In order to capture the ordered structures behind Boolean/binary variables, vector $\boldsymbol{y}$ can be reformulated with a reduced number of auxiliar integer variables known as external variables $\boldsymbol{e} \in \mathbb{Z}^{n_e}$, i.e., $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{e})$. Note that integer variables $\boldsymbol{n}$ are ordered by default, hence these do not need to be reformulated. For instance, integer variables $\boldsymbol{n}$ that appear in the scheduling formulation may be integer batching variables, which decide the total number of times a task is executed within the scheduling horizon[389]. A similar example of integer variables $\boldsymbol{n}$ are batching-assignment variables, which refer to the number of times a task is processed in a given unit[361]. An introductory example of binary variables $\boldsymbol{y}$ with an ordered structure and their reformulation $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{e})$ is provided below. An example of binary/Boolean variables with ordered structures are the Boolean variables $\boldsymbol{y}$ associated to the processing time of tasks in the discrete-time STN scheduling model shown in chapter 8. The reformulated version of the optimization problem (8.1A) in chapter 8 is shown in equation (H.1A), which incorporates both the variable partitioning introduced in equation (8.1B) in chapter 8 and reformulation $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{e})$. The upper and lower bounds imposed directly over $\boldsymbol{e}$ and $\boldsymbol{n}$ are represented through sets E and N, to highlight that complicating variables are bounded. The reformulation of variables $\boldsymbol{y}$ into external variables $\boldsymbol{e}$ is formally defined in equations (H.1B) and (H.1C) when variables in $\boldsymbol{y}$ are declared as binary variables, and in equation (H.1D) and (H.1E) when variables in $\boldsymbol{y}$ are declared as Boolean variables. These equations are explained in detail below.

$$\min_{s.t.} f\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}\right) \tag{H.1A}$$
$$\boldsymbol{v}^{sch} = [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}] \in SCH, \boldsymbol{v}^{dyn} \in DYN, [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}] \in LINK$$
$$\boldsymbol{e} \in E, \boldsymbol{n} \in N$$

$$\boldsymbol{y} \coloneqq y_{b,\tilde{e}_b}, \forall \tilde{e}_b \in E_b, \forall b \in B, \text{ with } b = (i_1, i_2, \dots, i_{\tilde{n}}), B = I_1 \times I_2 \times, \dots, \times I_{\tilde{n}} \tag{H.1B}$$

$$E_b \subset \mathbb{Z}, 0 < |E_b| < \infty\ ,\ \sum_{\tilde{e}_b \in E_b} y_{b,\tilde{e}_b} = 1, \forall b \in B \iff \boldsymbol{e} \in E, \boldsymbol{y} = \boldsymbol{y}(\boldsymbol{e}) \tag{H.1C}$$

$$\boldsymbol{e} \in E \coloneqq e_b \in E_b, \forall b \in B \tag{H.1D}$$
$$\boldsymbol{y}(\boldsymbol{e}) \coloneqq y_{b,\tilde{e}_b} = \begin{cases} 1, \tilde{e}_b = e_b \\ 0, \tilde{e}_b \neq e_b \end{cases}, \forall \tilde{e}_b \in E_b, \forall b \in B$$

$$E_b \subset \mathbb{Z}, 0 < |E_b| < \infty,\ \underset{\tilde{e}_b \in E_b}{\vee} y_{b,\tilde{e}_b}\ , \forall b \in B \iff \boldsymbol{e} \in E, \boldsymbol{y} = \boldsymbol{y}(\boldsymbol{e}) \tag{H.1E}$$

$$\boldsymbol{e} \in E \coloneqq e_b \in E_b, \forall b \in B \tag{H.1F}$$
$$\boldsymbol{y}(\boldsymbol{e}) \coloneqq y_{b,\tilde{e}_b} = \begin{cases} True, \tilde{e}_b = e_b \\ False, \tilde{e}_b \neq e_b \end{cases}, \forall \tilde{e}_b \in E_b, \forall b \in B$$

Variables $\boldsymbol{y}$ in optimization problems are often defined using multiple sub-indices (also called indices) that allow to define multiple variables over multiple sets. Exploiting the structure of these indices is the basis to perform the external variables reformulation, and subsequently, the master-subproblem decomposition. This study focuses on the case where vector $\boldsymbol{y}$ contains a single type of indexed variables (e.g., the user declares indexed variables $y_{i_1,i_2}, \forall i_1 \in I_1, \forall i_2 \in I_2$), but the reader is referred to previous works for more general scenarios, e.g., two types of indexed variables $yp_{i_1,i_2}, yq_{i_3,i_4}$[4]. The key idea of the reformulation with external variables is to take advantage of the special ordered structure of one of the indices of variables in $\boldsymbol{y}$. Thus, the reformulation herein presented focuses on a general index structure, where variables in $\boldsymbol{y}$ are defined with respect to $\tilde{n} + 1$ indices, where $\tilde{n}$ of those indices are ordinary indices, i.e., $i_1 \in I_1, i_2 \in I_2, \dots, i_{\tilde{n}} \in I_{\tilde{n}}$, plus an extra index $\tilde{e}_{i_1,i_2\dots,i_{\tilde{n}}} \in E_{i_1,i_2\dots,i_{\tilde{n}}}$ with special ordered properties. Note that special index $\tilde{e}_{i_1,i_2\dots,i_{\tilde{n}}}$ is defined over an indexed set $E_{i_1,i_2\dots,i_{\tilde{n}}}$ to account for the most general case where $E_{i_1,i_2\dots,i_{\tilde{n}}}$ has elements that vary depending on the value of the remaining indices $i_1, i_2 \dots, i_{\tilde{n}}$. For example, the SSDO formulation introduced in chapter 7 involves Boolean variables $Y_{\tau_{i,j},i,j}$ used to indicate if discrete processing time $\tau_{i,j}$ is selected for the execution of task i in unit j. Given that the optimal processing time may depend on the characteristics of the task (e.g., reaction or separation) or the unit (e.g., large reactor or small reactor), the set of potential processing times $\mathrm{T}_{i,j}$ may differ between task-unit pairs (i,j) to avoid combinations of $(\tau_{i,j}, i, j)$ that are *a-priori* known to result in infeasibilities. The procedure to

reformulate Boolean variables $Y_{\tau_{i,j},i,j}$ related to variable processing times in the discrete-time STN SSDO formulation is available in section 8.1.2. In this section, the author provides more general guidelines to perform the reformulation of variables in $\boldsymbol{y}$ using external variables $\boldsymbol{e}$, as explained below.

To simplify the notation, indices of $\boldsymbol{y}$ are regrouped in this work as a tuple of indices $b \in B$, and single index $\tilde{e}_b \in E_b$ to perform the reformulation with external variable, i.e., the reformulation will be applied over index $\tilde{e}_b$, hence it is segregated from the remaining indices in b. For example, consider variables $\boldsymbol{y} = [y_{\psi,\theta,1}, y_{\psi,\theta,2}, y_{\psi,\gamma,1}]$, with $I_1 = \{\psi\}$, $I_2 = \{\theta,\gamma\}, E_{\psi,\theta} = \{1,2\}, E_{\psi,\gamma} = \{1\}$. These variables can be regrouped as $\boldsymbol{y} = [\boldsymbol{y}_{\psi,\theta}, \boldsymbol{y}_{\psi,\gamma}]$, with $\boldsymbol{y}_{\psi,\theta} = [y_{\psi,\theta,1}, y_{\psi,\theta,2}]$ and $\boldsymbol{y}_{\psi,\gamma} = [y_{\psi,\gamma,1}]$, or equivalently, $y_{b,\tilde{e}_b}, \forall \tilde{e}_b \in E_b, \forall b \in \{(\psi,\theta),(\psi,\gamma)\}$. Following this notation, variables $\boldsymbol{y}$ tuples b, and set B are defined in equation (H.1B). The left-hand-side of the equivalence in equations (H.1C) and (H.1E) shows the requirements that the sub-indices of $\boldsymbol{y}$ must satisfy to perform the reformulation. The first requirement is shown below, which is needed to ensure that binary/Boolean variables $\boldsymbol{y}$ can be reformulated into integer external variables with a bounded search space.

R1: Sets $E_b, \forall b \in B$ must be non-empty finite subset of $\mathbb{Z}$, i.e., $E_b \subset \mathbb{Z}, 0 < |E_b| < \infty,$ where $|E_b|$ denotes the cardinality of E_b in equations (H.1C) and (H.1E). Besides, the elements of E_b must be consecutive numbers ordered with respect to the standard ascending order $\leq$, e.g., $E_b = \{2,3,4,5\}$.

The second requirement is related to the physical meaning behind the data used to construct set E_b. Based on R1, set E_b must be defined using ordered integer numbers. However, this does not prohibit the use of other types of data in the formulation, e.g., the elements of a set can be replaced by integers. Discrete data can be catalogued as categorical and numerical, where the former includes nominal (e.g., set of reactants) and ordinal data (e.g., set of execution priorities), while the latter includes interval (e.g., set of evenly-spaced temperature operation points) and ratio data (e.g., set of evenly-spaced processing times)[376]. The key difference between these data types is that a set created using ordinal, interval or ratio data inherently imposes an intuitive ascending order between its elements that can be used to define E_b, e.g., the execution of a task in a scheduling problem may have execution priorities "low", "medium", and "high" that can be mapped into a set $E_b = \{1,2,3\}$ assigning 1 to "low", 2 to "medium" and 3 to "high". On the contrary, a set created using nominal data does not impose a particular order over its elements, making it redundant to assign nominal data to ordered elements in E_b, e.g., there are multiple ways to define a one-to-one correspondence between members of a set of

chemical components $\{A, B, C, D\}$ and set $E_b = \{1,2,3,4\}$. This idea is formally expressed formally in the second requirement shown below.

R2: The data that is encoded by the user into $E_b, \forall b \in B$ in equations (H.1C) and (H.1E) can only be ordinal, discrete interval or discrete ratio data, but not nominal data. Besides, this data must be encoded in E_b using their underlying intuitive ascending order, e.g., "small" is assigned to 1, "medium" is assigned to 2, and "large" is assigned to 3.

The last requirement that is imposed over the problem formulation to enable the reformulation with external variables is the presence of constraints that enforce that, for every b in B, exactly one variable from $y_{b,\tilde{e}_b}, \forall \tilde{e}_b \in E_b$ must be activated, i.e.,

R3: Exactly one variable from $y_{b,\tilde{e}_b}, \forall \tilde{e}_b \in E_b$ is equal to 1 or $True$, as specified by the summation expression in equation (H.1C) and the exclusive or ($\veebar$) expression in equation (H.1E).

This requirement, suggests that the goal of a group of variables $y_{b,\tilde{e}_b}, \forall \tilde{e}_b \in E_b$ in the optimization problem is to identify the optimal index e_b from E_b that, when activating $y_{b,\tilde{e}_b}$ at $\tilde{e}_b = e_b$, optimizes objective function f. Note that this idea applies for every index $\tilde{e}_b, \forall b \in B$. Thanks to R1, the sought indices $e_b, \forall b \in B$ may take integer values within a bounded ranges $\min(E_b) \leq e_b \leq \max(E_b)$, hence they can be regarded as an integer variables. These integer variables can be grouped in vector of external variables $\boldsymbol{e} \in E$, where E limits variables e_b in $\boldsymbol{e}$ within their upper and lower bounds $\max(E_b)$ and $\min(E_b)$. Furthermore, requirement R2 ensures that increasing or decreasing the value of the variables in $\boldsymbol{e}$ has the physical interpretation of evaluating different levels in an ordered scale of measures, e.g., temperature levels, discrete points in time, ordered locations, etc. R2 becomes particularly relevant for the proposed MLD-BD algorithm, whose local optimality guarantees rely on a local sensitivity analysis that evaluates the effect of increasing and decreasing integer variables $\boldsymbol{n}$ and external variables $\boldsymbol{e}$ in the objective function.

The equations that allow to evaluate variables $\boldsymbol{y}$ in terms of the value of external variables $\boldsymbol{e}$ is denoted by $\boldsymbol{y}(\boldsymbol{e})$ and formally defined in equations (H.1D) and (H.1F) for the binary and Boolean cases, respectively. Note that the solution strategy presented in this work is only applicable when requirements R1-R3 are satisfied by the variables that are aimed to be reformulated in the optimization problem at hand, as in the case of variables $Y_{i,j,\tau_{i,j}}$ in the SSDO problem considered in this work. The ordered structures (if any) of those binary/Boolean variables that do not satisfy R1-R3 cannot be

deduced by the proposed reformulation. For example, binary variables $x_{i,j,t}$ in $\boldsymbol{y}^{sch}$ are used in chapter 8 to find the optimal timing of tasks within a discrete time set that satisfies R1 and R2, however, constraints of type R3 are not defined for these variables. This is because it is not known a-priori how many times a task will be executed within the scheduling horizon, i.e., an unknown number of time points in the discrete time set will have active binary variables $x_{i,j,t}$ to indicate the starting-time of tasks. Although R3 can be extended to consider more general scenarios such as exactly $n_{i,j}$ from $x_{i,j,t}, \forall t \in T$ take the value of 1, the fact that $n_{i,j}$ is variable makes it non-trivial to reformulate the timing of variables as integer external variables using the ideas discussed above, thus leaving the door open for further research. An example that illustrates reformulation requirements R1-R3 and the reformulation with external variables is provided below.

Example H.1: Assume $\boldsymbol{y}$ contains binary variables $y_{p,t}, \forall p \in P, \forall t \in T = \{1,2, \dots, n_T\}$, where index p represents a product, index t represents a time slot, and $y_{p,t} = 1$ means that production of product p is assigned to time slot t. In this example, set T satisfies the requirements in R1, i.e., it is a non-empty finite subset of $\mathbb{Z}$, and its elements are consecutive integers in ascending order. Set T also satisfies the requirement in R2, as long as time slots t are defined in ascending order. Considering the cyclic scheduling of a single-unit multiproduct process leads to a constraint of the form $\sum_{t \in T} y_{p,t} = 1, \forall p \in P$, indicating that product p can only be manufactured once in each cycle[81]. This constraint satisfies requirement R3. Thus, this problem can be reformulated using external variables $e_p \in \{1,2, \dots, n_T\}, \forall P$, where integer variable e_p represents the time slot selected to manufacture p, and the original variables of the problem $y_{p,t}$ can be expressed as a function of e_p, i.e., $y_{p,t}$ takes the value of 1 if $t = e_p$; 0 otherwise.

H.2 MLD-BD background

As discussed in chapter 8, GBD is currently the standard to solve SSDO problems though existing GBD strategies still present a few limitations. In general, optimization problems involving multiple binary variables as in SSDO are recognized for their adverse impact in the computational speed of GBD[390]. Thus, MLD-BD incorporates a reformulation step that allows to deal with the combinatorial complexity added by those binary variables used to represent ordered structures in the formulation, e.g., Boolean variables used to decide between processing time options in discrete-time SSDO formulations. Another characteristic of previous GBD implementations is that they rely on traditional convexity notions, e.g.,

when the projection of optimization problem (8.1A) in chapter 8 onto its complicating variables is convex, then global optimality is guaranteed[380]. Instead, MLD-BD relies on discrete convex analysis principles, which acknowledge that alternative discrete convexity definitions are available for MINLP problems, e.g., integrally convex, separable convex, etc[4,6]. The proposed MLD-BD strategy relies on separable convexity, meaning that the local optimum it identifies is a global optimum if the projection of problem (8.4) in chapter 8 onto its complicating variables is separable convex. A current limitation of discrete convex analysis theory is that the discrete convexity characteristics of a general MINLP problem can only be assessed through exhaustive enumeration techniques. Nevertheless, proposing solution algorithms such as MLD-BD that rely on alternative convexity characteristics of MINLP problems is relevant to broaden the scope of problems that can be solved to global optimality with local or heuristic methods, as highlighted by example H.2.

Example H.2: consider the optimization problem of minimizing z subject to $-e\sin(3.5\pi e) - z \leq 0$, with $z \in \mathbb{R}$, $e \in \mathbb{Z}$ and $2 \leq e \leq 5$. When relaxing the integer requirement over e, the resulting relaxed problem is not convex, hence, global optimality cannot be guaranteed with traditional local MINLP solvers. On the contrary, this problem can be analyzed form a discrete convex analysis perspective by performing an exhaustive enumeration over e and solving the subproblems that result from fixing every feasible solution of e. Using the definition of separable convexity proposed by Murota[6], this exhaustive enumeration can be used to conclude that this problem is separable convex. This implies that the local search performed by MLD-BD would return the global optimum of the problem. On the contrary, GBD may stagnate at local or non-stationary points depending in the initialization selected[6,15].

This example illustrates how some problems may benefit from the local search performed by MLD-BD to identify globally optimal solutions that cannot be attained with a standard GBD method. Despite this, the authors acknowledge that local MINLP algorithms based on discrete convex analysis do not aim to solve a larger class of problems to global optimality than traditional local MINLP methods. To illustrate this, an example of an optimization problem for which traditional local optimization techniques are preferred over local discrete convex analysis techniques is provided in example H.3.

Example H.3: consider the minimization of $4|2e_1 + e_2 - 8| + |e_1 - 2e_2 + 6|$, with $[e_1, e_2] \in \mathbb{Z}^2$ and $1 \leq e_i \leq 4, \forall i \in \{1,2\}$[294]. When relaxing the integer requirement over $[e_1, e_2]$, the resulting relaxed problem is convex. However, when analyzed from a discrete convex analysis point of view,

this problem is not integrally convex and it has two local optima at [2,4] and [3,2], where [2,4] is the global optimum.

Another advantage of MLD-BD is the way it handles infeasibilities. Previous works in GBD for SSDO assume subproblem feasibility for all scheduling combinations, which is impractical when incorporating variable processing times that can potentially result in infeasible scenarios, e.g., saturation violations in the manipulated variables when aiming to satisfy constraints within a limited processing time. Although feasibility cuts can be implemented within GBD, these may not be tight enough to cut off large infeasible regions (see appendix B for further details). In contrast, the proposed MLD-BD framework is able to move away from infeasibilities once feasible regions are explored and still guarantee convergence to a feasible solution in a finite number of iterations, without the need of including feasibility cuts. A proof that supports this claim is provided in the next section.

H.3 LBBD feasibility algorithm

The feasibility algorithm proposed in this work to identify feasible combinations of complicating variables from an infeasible point $[\boldsymbol{n}^{init}, \boldsymbol{e}^{init}]$ follows the LBBD guidelines provided in the work by Hooker[10]. This feasibility algorithm considers the master-subproblem decomposition shown in equations (H.2A) and (H.2B). This custom decomposition keeps all the constraints of the problem in the subproblem (H.2A) with ordered discrete decisions $\boldsymbol{e}^k$ and $\boldsymbol{n}^k$ fixed at a given iteration k. Note that ordered binary/Boolean variables in $\boldsymbol{y}$ also remain fixed in the subproblems, given that they can be calculated as a function of $\boldsymbol{e}^k$. Thus, the objective of master problem (H.2B) is to find a trial combination of ordered discrete decisions $\boldsymbol{e}^k$ and $\boldsymbol{n}^k$ that, when fixed, return a feasible solution to the SSDO subproblem (H.2A). To achieve this goal, feasibility cuts $\boldsymbol{\varkappa}_{[\boldsymbol{e},\boldsymbol{n}]^{\hat{k}}}(\boldsymbol{e},\boldsymbol{n}) \leq 0, \forall\, \hat{k} \in \{1,2,\dots,k\}$ are included in the master problem. These cuts are constraints that are violated whenever $[\boldsymbol{e},\boldsymbol{n}]$ is equal to $\left[\boldsymbol{e}^{\hat{k}}, \boldsymbol{n}^{\hat{k}}\right], \forall\, \hat{k} \in \{1,2,\dots,k\}$, which allows to evaluate unexplored combinations of ordered discrete decisions in subsequent iterations. Based on the above, a feasibility LBBD procedure iterates between a master problem and a feasibility subproblem until a feasible combination of complicating variables from the master problem returns a feasible subproblem.

$$[\boldsymbol{y}(\boldsymbol{e}^k), \boldsymbol{n}^k, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}] \in SCH, \boldsymbol{v}^{dyn} \in DYN,\ [\boldsymbol{y}(\boldsymbol{e}^k), \boldsymbol{n}^k, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}] \in LINK \qquad \text{(H.2A)}$$

$$[\boldsymbol{e}^{k+1}, \boldsymbol{n}^{k+1}] \in \text{argmin}\, f_1\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}\right) \quad \text{(H.2B)}$$
$$s.t. [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}] \in SCH, [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}] \in LINK$$
$$\boldsymbol{\varkappa}_{[\boldsymbol{e},\boldsymbol{n}]^{\hat{k}}}(\boldsymbol{e}, \boldsymbol{n}) \leq 0, \forall\, \hat{k} \in \{1,2, \dots, k\}, \boldsymbol{e} \in E, \boldsymbol{n} \in N$$

For the proposed feasibility LBBD algorithm, the master problem has been augmented with the linear constraints of the problem, which in this case are the scheduling and linking constraints with their corresponding variables. This allows to narrow the search space of ordered discrete decisions $[\boldsymbol{e}, \boldsymbol{n}]$ to those combinations that satisfy scheduling requirements, while keeping the master problem as an MIP problem. Accordingly, objective function f_1 is considered in the master problem, which incorporates linear scheduling terms only. Based on the master-subproblem decomposition above, the present work relies on heuristics adopted to generate feasibility cuts and to accelerate the convergence of the feasibility LBBD algorithm. Note that the use of heuristics within feasibility LBBD strategies is a well-stablished practice to accelerate LBBD algorithms[8,276,391–393]. The proposed cut generation strategy relies on the approximated solution of subproblems and the solution of auxiliary subproblems, as discussed below.

Approximated solution of subproblems in equation (H.3A): a strategy to simplify the solution of subproblems similar to the one considered in section 8.2.3.2 is used. According to the formulation in section 8.1.1, binary scheduling decisions in ($\boldsymbol{y}^{sch}$) only affect the sequencing and starting execution time of batch tasks, whose dynamic performance is predefined by initial conditions and end-point constraints. Hence the variables in $\boldsymbol{y}^{sch}$ are not expected to have a direct impact on the feasibility of dynamic variables $\boldsymbol{v}^{dyn}$. The feasibility of dynamic constraints $\boldsymbol{v}^{dyn} \in DYN$ is rather dictated by other scheduling constraints such as processing times and batch sizes defined through $\boldsymbol{x}^{sch}$ and $\boldsymbol{y}$, e.g., product quality requirements may not be satisfied for a large batch size and a short processing time. Therefore, it is proposed to keep variables in $\boldsymbol{y}^{sch}$ fixed in feasibility subproblems. The outcome of this approximation is that the resulting subproblems become NLPs that can be solved faster than the original MINLPs. Note that despite the assumption of keeping variables $\boldsymbol{y}^{sch}$ fixed, these simplified feasibility subproblems shown in equation (H.3A) still consider the interactions between scheduling variables $\boldsymbol{x}^{sch}$ and dynamic variables $\boldsymbol{v}^{dyn}$.

Auxiliary subproblems (H.3B) to generate the cuts used in equation (H.3C): no general strategy is available to generate feasibility cuts feasibility cuts $\boldsymbol{\varkappa}_{[\boldsymbol{e},\boldsymbol{n}]^{\hat{k}}}(\boldsymbol{e}, \boldsymbol{n}) \leq 0, \forall\, \hat{k} \in \{1,2, \dots, k\}$, hence these

must be derived on a case-by-case basis according to LBBD theory[10]. The most basic cuts that can be generated are called no-good cuts, which simply exclude infeasible combinations of variables $\boldsymbol{e}^k$ and $\boldsymbol{n}^k$ from the feasible region. Another strategy available to generate these cuts is that, after infeasibility has been detected in subproblem (H.2A), auxiliary subproblems can be used to detect the source of this infeasibility and generate feasibility cuts. This work relies on cuts derived from auxiliary subproblems that take advantage of special properties of the optimization problem shown in section 8.1.1 and its decomposition. Given that scheduling SCH and linking $LINK$ constraints were included in master problem (H.2B), infeasibility in the subproblems is expected to be caused by dynamic constraints $\boldsymbol{v}^{dyn} \in DYN$. Consequently, feasibility cuts derived from the auxiliary subproblems should aim to make dynamic constraints feasible in subsequent master problem iterations. The complicating variables in the master problem that are directly connected to dynamic constraints are external variables $\boldsymbol{e}$ that decide the number of discrete time points used for the execution of a batch operation. If infeasibility is detected for a fixed combination of external variables, then more discrete points are expected to be needed to satisfy dynamic requirements, e.g., end point product quality constraints. Thus, the feasibility cuts proposed in this work update the lower bounds imposed over external variables $\boldsymbol{e}$ at each iteration. For this purpose, I take advantage of the fact that the dynamic constraints of the problem have a separable structure, i.e., $\boldsymbol{v}^{dyn} \in DYN \Leftrightarrow \boldsymbol{v}_{i,j}^{dyn} \in DYN_{i,j} \forall (i,j) \in IJ^{\times,dyn}$. Therefore, auxiliary subproblems in equation (H.3B) are defined and solved iteratively for every scheduled task-unit pair $(i,j) \in IJ^{\times dyn}$ with the purpose of understanding which task-unit pair is generating infeasibilities. Then, cuts are added to the master problem for those scheduled tasks (i.e., $n_{i,j} \geq 1$) with infeasible dynamics, as shown in master problem (H.3C).

$$[\boldsymbol{y}(\boldsymbol{e}^k), \boldsymbol{n}^k, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch,k}] \in SCH, \boldsymbol{v}^{dyn} \in DYN,\ [\boldsymbol{y}(\boldsymbol{e}^k), \boldsymbol{n}^k, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch,k}, \boldsymbol{v}^{dyn}] \in LINK \tag{H.3A}$$

$$[\boldsymbol{y}(\boldsymbol{e}^k), \boldsymbol{n}^k, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch,k}] \in SCH, \boldsymbol{v}_{i,j}^{dyn} \in DYN_{i,j},\ [\boldsymbol{y}(\boldsymbol{e}^k), \boldsymbol{n}^k, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch,k}, \boldsymbol{v}^{dyn}] \in LINK \tag{H.3B}$$

$$[\boldsymbol{e}^{k+1}, \boldsymbol{n}^{k+1}] \in \operatorname{argmin} f_1\left(\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}\right) \quad \text{(H.3C)}$$

$$s.t.$$

$$[\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}] \in SCH, [\boldsymbol{y}(\boldsymbol{e}), \boldsymbol{n}, \boldsymbol{x}^{sch}, \boldsymbol{y}^{sch}, \boldsymbol{v}^{dyn}] \in LINK$$

$$e_{i,j} \geq e_{i,j}^{\hat{k}} + 1, \forall (i,j) \in INFEAS^{\hat{k}}, \forall\, \hat{k} \in \{1,2,\dots,k\}$$

$$INFEAS^{\hat{k}} = \left\{(i,j) \in IJ^{\times,dyn} : n_{i,j}^{\hat{k}} \geq 1, \text{(H. 2B) } infeasible\ at\ \hat{k}\right\}$$

Based on the above decomposition and cut generation strategy, the standard feasibility LBBD algorithm proposed by Hooker[10] is executed, which terminates once a feasible solution is detected. Then, the execution of the MLD-BD algorithm starts from a feasible combination of complicating variables. Note that MLD-BD does not need feasibility cuts, because the algorithm is guaranteed to converge to a feasible solution without them, as demonstrated formally through prepositions H.1 and H.2.

Proposition H.1. Assume that the search space of complicating variables is bounded. Then, the MLD-BD algorithm in Figure 8.1 will terminate its exploration in a finite number of iterations.

Proof of Proposition H.1. The following scenarios are possible for the updated complicating variables $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ at any iteration k:

1. The conditions stated in Step 5 of the MLD-BD algorithm are satisfied: this means that there exists $\hat{k} \in \{1,2,\dots,k\}$ such that $\left[\boldsymbol{n}^{\hat{k}}, \boldsymbol{e}^{\hat{k}}\right] = [\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ and $f^*\left(\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}\right) \neq \min_{[\boldsymbol{n},\boldsymbol{e}] \in D^k} \{f^*(\boldsymbol{n}, \boldsymbol{e})\}$. Following Step 5 of the MLD-BD algorithm, $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ will be updated as $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}] \leftarrow \operatorname{argmin}_{[\boldsymbol{n},\boldsymbol{e}] \in D^k} \{f^*(\boldsymbol{n}, \boldsymbol{e})\}$, which is expected to be either a previously evaluated central point or a previously evaluated neighboring point (as defined in section 8.2.4). In the former case (i.e., $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ is a previously evaluated central point), having $f^*\left(\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}\right) = \min_{[\boldsymbol{n},\boldsymbol{e}] \in D^k} \{f^*(\boldsymbol{n}, \boldsymbol{e})\}$ means that new subproblems will not be solved at iteration $k+1$, i.e., the algorithm will stop at iteration $k+1$. In the latter case (i.e., $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ is a previously evaluated neighboring point), neighboring point $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ becomes the central point for iteration $k+1$.

2. The conditions stated in Step 5 of the MLD-BD algorithm are not satisfied: if there exists $\hat{k} \in \{1,2,\dots,k\}$ such that $[\boldsymbol{n}^{\hat{k}}, \boldsymbol{e}^{\hat{k}}] = [\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ and $f^*(\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}) = \min_{[\boldsymbol{n},\boldsymbol{e}]\in D^k}\{f^*(\boldsymbol{n},\boldsymbol{e})\}$, then $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ can be considered a central point evaluated in previous iterations, new subproblems will not be solve at iteration $k+1$, and the algorithm is expected to stop at iteration $k+1$. On the contrary, if there is no $\hat{k} \in \{1,2,\dots,k\}$ such that $[\boldsymbol{n}^{\hat{k}}, \boldsymbol{e}^{\hat{k}}] = [\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$, then $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$ will either be a previously evaluated neighboring point, or a new trial solution whose subproblem has not been solved, so it will become the central point for iteration $k+1$.

Based on the above, at any iteration k, MLD-BD is either expected to terminate at iteration $k+1$ or move to an unexplored central point $[\boldsymbol{n}^{k+1}, \boldsymbol{e}^{k+1}]$. There are only finitely many possible combinations of central points because the search space of complicating variables is bounded, and complicating variables are integers. Thus, the algorithm must terminate after a finite number of iterations.

Proposition H.2. Assume that the MLD-BD algorithm in Figure 8.1 is initialized from a feasible point $[\boldsymbol{n}^{init}, \boldsymbol{e}^{init}]$, and none of the subproblems has an unbounded objective function. Then the solution obtained with MLD-BD is feasible.

Proof of Proposition H.2. The solution found by MLD-BD must satisfy $f^*(\boldsymbol{n}^k, \boldsymbol{e}^k) = \min_{[\boldsymbol{n},\boldsymbol{e}]\in D^k}\{f^*(\boldsymbol{n},\boldsymbol{e})\}$. Assume that the solution obtained is infeasible, then $f^*(\boldsymbol{n}^k, \boldsymbol{e}^k) = +\infty$. $f^*(\boldsymbol{n}^1, \boldsymbol{e}^1) < +\infty$ because $[\boldsymbol{n}^{init}, \boldsymbol{e}^{init}] = [\boldsymbol{n}^1, \boldsymbol{e}^1]$ is feasible, however, having $[\boldsymbol{n}^1, \boldsymbol{e}^1] \in D^k$ indicates that $f^*(\boldsymbol{n}^k, \boldsymbol{e}^k) \neq \min_{[\boldsymbol{n},\boldsymbol{e}]\in D^k}\{f^*(\boldsymbol{n},\boldsymbol{e})\}$, i.e., a contradiction. Furthermore, none of the subproblems is unbounded, suggesting that $[\boldsymbol{n}^k, \boldsymbol{e}^k]$ is feasible.

H.4 Supplementary information for the case studies

H.4.1 Objective function

The different terms of the objective function used for both case study 1 and case study 2 are shown in equations (H.4A)-(H.4F). This objective function minimizes the sum of fixed costs (c^{fix}), material costs (c^{mat}), variable costs for tasks with (c^{var_1}) and without (c^{var_2}) dynamic models, and the negative of the revenue ($-rev$). Variable costs c^{var_1} and c^{var_2} distinguish between variable costs for task-unit pairs without and with the dynamic models available, respectively. Cost c^{var_2}, related to the transient

operation of tasks, assumes that there are heating and cooling utilities available, whose flows $\mu_{i,j,t}^{hot}$ and $\mu_{i,j,t}^{cold}$ can be manipulated. The integral of flows $\mu_{i,j,t}^{hot}$ and $\mu_{i,j,t}^{cold}$ over dimensionless time ψ in equation (H.4D) are multiplied by coefficients c^{hot} (10 $\$/m^3$) and c^{cold} (1 $\$/m^3$) to obtain their costs. For the second case study, the dynamic cost term c^{var_2} also includes the operating cost of the distillation column unit, which is assumed to be proportional to the integral over time of the boil-up rate through coefficient c^{sep} (0.05 $\$/m^3$). Sets M^{in} and M^{out} in equations (H.4E) and (H.4F) are subsets of M, and contain input materials and output products, respectively.

$$f = c^{fix} + c^{var_1} + c^{var_2} + c^{mat} - r \quad \text{(H.4A)}$$

$$c^{fix} = \sum_{i\in I}\sum_{j\in J}\sum_{t\in T} c_{i,j}^{fix} x_{i,j,t} \quad \text{(H.4B)}$$

$$c^{var_1} = \sum_{(i,j)\in IJ^{\times}\setminus IJ^{\times dyn}}\sum_{t\in T} c_{i,j}^{var_1} b_{i,j,t} \quad \text{(H.4C)}$$

$$c^{var_2} = \sum_{(i,j)\in IJ^{\times dyn}}\sum_{t\in T} x_{i,j,t}\tau_{i,j,t}^{P}\left(c^{hot}\int_0^1 \mu_{i,j,t}^{hot}\,d\psi + c^{cold}\int_0^1 \mu_{i,j,t}^{cold}\,d\psi\right) \quad \text{(for case study 1)} \quad \text{(H.4D)}$$

$$c^{var_2} = \sum_{(i,j)\in IJ^{\times rx}}\sum_{t\in T} x_{i,j,t}\tau_{i,j,t}^{P}\left(c^{hot}\int_0^1 \mu_{i,j,t}^{hot}\,d\psi + c^{cold}\int_0^1 \mu_{i,j,t}^{cold}\,d\psi\right) + \sum_{(i,j)\in IJ^{\times sep}}\sum_{t\in T} x_{i,j,t}\tau_{i,j,t}^{P}\left(c^{sep}\int_0^1 \mu_{i,j,t}^{V}\,d\psi\right) \quad \text{(for case study 2)}$$

$$c^{mat} = \sum_{m\in M^{in}} c_m^{mat}\left(s_m^{init} - s_{m,n_T}\right) \quad \text{(H.4E)}$$

$$rev = \sum_{m\in M^{out}} c_m^{out} s_{m,n_T} \quad \text{(H.4F)}$$

H.4.2 Case study 1

Tables H.1-H.5 show the parameters required to define the first case study of chapter 8.

Table H.1. Maximum and minimum capacity of the units (case study 1)

Unit $j \in J$	b_j^{min} $[m^3]$	b_j^{max} $[m^3]$
Mix	0.2	2
Rl	0.15	1.5
Rs	0.1	1
Sep	0.2	2
$Pack$	0.1	1

Table H.2. Parameters related to states $m \in M$ (case study 1)

State $m \in M$	s_m^{init} $[m^3]$	s_m^{max} $[m^3]$	c_m^{mat} $[\$/m^3]$	c_m^{out} $[\$/m^3]$
$M1$	10	10	500	-
$M2$	10	10	300	-
$M3$	10	10	400	-
$S1$	10	10	-	-
$I1$	0	2	-	-
$I2$	0	2	-	-
$I3$	0	2	-	-
$I4$	0	2	-	-
$I5$	0	5	-	-
$I6$	0	5	-	-
$W1$	0	10	-	-
$P1$	0	10	-	1400
$P2$	0	10	-	1200

Table H.3. Parameters related to pairs $(i,j) \in IJ^{\times}$ (case study 1)

Pair $(i,j) \in IJ^{\times}$	$\tau_{i,j}^{P,min}$ $[h]$	$\tau_{i,j}^{P,max}$ $[h]$	$\tau_{i,j}^{P,fixed}$ $[h]$	$c_{i,j}^{fix}$ $[\$]$	$c_{i,j}^{var_1}$ $[\$/m^3]$
(Mix, Mix)	1.5	1.5	1.5	10	30
$(R1, Rl)$	1	4	-	30	-
$(R1, Rs)$	1	4	-	20	-
$(R2, Rl)$	1	4	-	30	-
$(R2, Rs)$	1	4	-	20	-
$(R3, Rl)$	1	4	-	30	-
$(R3, Rs)$	1	4	-	20	-
(Sep, Sep)	3	3	3	100	100
$(Pack1, Pack)$	1.5	1.5	1.5	50	50
$(Pack2, Pack)$	1.5	1.5	1.5	50	50

Table H.4. Parameters in the dynamic models (case study 1)

Parameter	Value	Description
k_{R1} $[m^3/kmol\ h]$	10000000	Pre-exponential factor of reaction $R1$
k_{R2} $[m^3/kmol\ h]$	1200	Pre-exponential factor of reaction $R2$
k_{R3} $[m^3/kmol\ h]$	20000	Pre-exponential factor of reaction $R3$
$\widetilde{Ea}_{R1}$ $[K]$	5000	Normalized activation energy of reaction $R1$

$\widetilde{Ea}_{R2}$ $[K]$	2000	Normalized activation energy of reaction $R2$
$\widetilde{Ea}_{R3}$ $[K]$	3000	Normalized activation energy of reaction $R3$
$\boldsymbol{C}_{R1,0}$ $[kmol/m^3]$	[1,1,0]	Initial concentration of reaction $R1$
$\boldsymbol{C}_{R2,0}$ $[kmol/m^3]$	[1,0.95,0]	Initial concentration of reaction $R2$
$\boldsymbol{C}_{R3,0}$ $[kmol/m^3]$	[1.0125,0.9,0]	Initial concentration of reaction $R3$
$\boldsymbol{C}_{R1,DES}$ $[kmol/m^3]$	[0.05,0.05,1.9]	Desired concentration of reaction $R1$
$\boldsymbol{C}_{R2,DES}$ $[kmol/m^3]$	[0.1,0.05,1.8]	Desired concentration of reaction $R2$
$\boldsymbol{C}_{R3,DES}$ $[kmol/m^3]$	[0.2125,0.1,0.8]	Desired concentration of reaction $R3$
Ua_{Rs} $[kJ/h\ K]$	20000	Heat transfer coefficient (reactor Rs)
Ua_{Rl} $[kJ/h\ K]$	30000	Heat transfer coefficient (reactor Rl)
$C_{P,R1}$ $[kJ/kg\ K]$	3	Heat capacity of reaction mixture $R1$
$C_{P,R2}$ $[kJ/kg\ K]$	3.5	Heat capacity of reaction mixture $R2$
$C_{P,R3}$ $[kJ/kg\ K]$	4	Heat capacity of reaction mixture $R3$
C_{PJ} $[kJ/kg\ K]$	4.2	Heat capacity (jacket side)
ρ_R $[kg/m^3]$	1000	Density of reaction mixture
ρ_J $[kg/m^3]$	1000	Density (jacket side)
ΔH_{R1} $[kJ/kmol]$	1000	Enthalpy of reaction $R1$
ΔH_{R2} $[kJ/kmol]$	-2000	Enthalpy of reaction $R2$
ΔH_{R3} $[kJ/kmol]$	5000	Enthalpy of reaction $R3$
T_{R0} $[K]$	300	Initial reactor temperature
T_H $[K]$	370	Temperature of heating fluid
T_C $[K]$	300	Temperature of cooling fluid
T_{J0} $[K]$	300	Initial temperature (jacket side)
VJ_{Rs} $[m^3]$	0.3	Volume (jacket side, reactor Rs)
VJ_{Rl} $[m^3]$	0.5	Volume (jacket side, reactor Rl)
T_R^{max} $[K]$	320	Maximum reactor temperature at $\tau = 1$

Table H.5. Variables in the dynamic models (case study 1)

Variable	Bounds	Description
$\boldsymbol{\chi}_{i,j}(\psi)$ $[kmol/m^3]$	[0,5]	Vector of concentrations
$\chi_{i,j}^{TR}(\psi)$ $[K]$	[295,370]	Reactor temperature
$\chi_{i,j}^{TJ}(\psi)$ $[K]$	[295,370]	Jacket temperature
$\zeta_{i,j}(\psi)$ $[kmol/m^3\ h]$	*	Reaction rates
$\mu_{i,Rs}^{hot}(\psi)$ $[m^3/h]$	[0,5]	Flow of heating fluid (reactor Rs)
$\mu_{i,Rs}^{cold}(\psi)$ $[m^3/h]$	[0,5]	Flow of cooling fluid (reactor Rs)
$\mu_{i,Rl}^{hot}(\psi)$ $[m^3/h]$	[0,10]	Flow of heating fluid (reactor Rl)
$\mu_{i,Rl}^{cold}(\psi)$ $[m^3/h]$	[0,10]	Flow of cooling fluid (reactor Rl)
$\varphi_{i,j}^{P}$ $[h]$	$[0, \tau_{i,j}^{P,max}]$	Reaction time
$\varphi_{i,j}^{Vol}$ $[m^3]$	$[b_j^{min}, b_j^{max}]$	Reaction volume

*: Defined based on the bounds imposed over other variables

The ordered discrete decisions of this problem are shown in Table H.6. In this case, the reformulation with external variables allowed to reformulate 52 Boolean variables into 10 external variables $e_{i,j}$, out of which 6 variables are used by MLD-BD to optimize the processing time of the different reactions considered in this network.

Table H.6. Domain of ordered discrete decisions (case study 1)

Variable	Domain	Variable	Domain
$n_{Mix,Mix}$	{0,1, …,9}	$e_{Mix,Mix}$	{3}
$n_{R1,R(large)}$	{0,1, …,14}	$e_{R1,R(large)}$	{2,3, …,8}
$n_{R1,R(small)}$	{0,1, …,14}	$e_{R1,R(small)}$	{2,3, …,8}
$n_{R2,R(large)}$	{0,1, …,14}	$e_{R2,R(large)}$	{2,3, …,8}
$n_{R2,R(small)}$	{0,1, …,14}	$e_{R2,R(small)}$	{2,3, …,8}
$n_{R3,R(large)}$	{0,1, …,14}	$e_{R3,R(large)}$	{2,3, …,8}
$n_{R3,R(large)}$	{0,1, …,14}	$e_{R3,R(large)}$	{2,3, …,8}
$n_{Sep,Sep}$	{0,1, …,4}	$e_{Sep,Sep}$	{6}
$n_{Pack1,Pack}$	{0,1, …,9}	$e_{Pack1,Pack}$	{3}
$n_{Pack2,Pack}$	{0,1, …,9}	$e_{Pack2,Pack}$	{3}

Based on the above, it is possible to formulate the constraints that define this case study, in addition to those shown in section 8.1.1. An additional scheduling constraint that is included in set SCH is shown in equation (H.5), which indicates that a minimum demand constraint must be satisfied at the end of the scheduling horizon for products $P1$ and $P2$. Also, a pure discrete-time representation is selected in this case, i.e., $\tau_{i,j,t}^{P} = \min\{\hat{\delta}\tau_{i,j}, \tau_{i,j}^{P,max}\}$.

$$s_{m,n_T} \geq s_{m,n_T}^{demand}, \forall m \in \{P1, P2\} \tag{H.5}$$

$$s_{P1,n_T}^{demand} = 0.5\ m^3, s_{P2,n_T}^{demand} = 1\ m^3\ (for\ \eta = 50400\ seconds)$$

$$s_{P1,n_T}^{demand} = 1\ m^3, s_{P2,n_T}^{demand} = 2\ m^3\ (for\ \eta = 100800\ seconds)$$

In this case, task $R1$ represents reaction $B + C \rightarrow 2D$, task $R2$ represents reaction $A + D \rightarrow 2E$, and task $R3$ represents reaction $C + E \rightarrow F$. The differential-algebraic equations in DYN used to model these reactions are shown in equations (H.6A)-(H.6F), where sets $Q_i, \forall i \in \{R1, R2, R3\}$ are introduced to denote the chemical species involved in each reaction, i.e., $Q_{R1} = \{B, C, D\}$, $Q_{R2} = \{A, D, E\}$ and $Q_{R3} = \{C, E, F\}$. $\boldsymbol{\chi}_{i,j}$ and $\boldsymbol{\nu}_i$ represent the concentrations and stochiometric coefficients of reaction

executed in task i, respectively, e.g., $\boldsymbol{\chi}_{R1,j} = \left[\chi^{q}_{R1,j}\right]_{q\in Q_{R1}} = \left[\chi^{B}_{R1,j}, \chi^{C}_{R1,j}, \chi^{D}_{R1,j}\right]$ and $\boldsymbol{\nu}_{R1} = \left[\nu^{q}_{R1}\right]_{q\in Q_{R1}} = [\nu^{B}_{R1}, \nu^{C}_{R1}, \nu^{D}_{R1}] = [-1, -1, 2]$, where notation $[\cdot]_{q\in Q_i}$ is used as short notation for these vectors.

Reaction rates (H.6A)

$$\zeta_{i,j} = -k_i \exp\left(\frac{-\widetilde{Ea}_i}{\chi^{TR}_{i,j}}\right) \prod_{\substack{q\in Q_i \\ \nu^{q}_{i}<0}} \left(\chi^{q}_{i,j}\right)^{-\nu^{q}_{i}}, \forall (i,j) \in IJ^{\times dyn}$$

Concentration balances (H.6B)

$$\dot{\boldsymbol{\chi}}_{i,j} = -\varphi^{P}_{i,j}\zeta_{i,j}\boldsymbol{\nu}_i, \boldsymbol{\chi}_{i,j}(\psi = 0) = \boldsymbol{C}_{i,0}, \forall (i,j) \in IJ^{\times dyn}$$

Energy balance (reactor temperature) (H.6C)

$$\dot{\chi}^{TR}_{i,j} = \varphi^{P}_{i,j}\left(\frac{Ua\left(\chi^{TJ}_{i,j} - \chi^{TR}_{i,j}\right)}{\varphi^{Vol}_{i,j} C_{P,i}\rho_R} + \frac{\Delta H_i \zeta_{i,j}}{C_{P,i}\rho_R}\right), \chi^{TR}_{i,j}(\psi = 0) = T_{R0}, \forall (i,j) \in IJ^{\times dyn}$$

Energy balance (jacket temperature) (H.6D)

$$\dot{\chi}^{TJ}_{i,j} = \varphi^{P}_{i,j}\left(\frac{\mu^{hot}_{i,j}\left(T_H - \chi^{TJ}_{i,j}\right) + \mu^{cold}_{i,j}\left(T_C - \chi^{TJ}_{i,j}\right)}{VJ_j} + \frac{Ua_j\left(\chi^{TR}_{i,j} - \chi^{TJ}_{i,j}\right)}{VJ_j\rho_J C_{PJ}}\right), \chi^{TJ}_{i,j}(\psi = 0) = T_{J0},$$
$$\forall (i,j) \in IJ^{\times dyn}$$

Maximum temperature constraint at the end (reactor cooling) (H.6E)

$$\chi^{TR}_{i,j}(\psi = 1) \leq T^{max}_{R}, \forall (i,j) \in IJ^{\times dyn}$$

Product concentration requirement (H.6F)

$$\boldsymbol{\chi}_{i,j}(\psi = 1) = \boldsymbol{C}_{i,DES}, \forall (i,j) \in IJ^{\times dyn}$$

An additional linking constraint ($LINK$) for this problem is shown in equation (H.7). It ensures that if $x_{i,j,t} = 1$, then the reactive mixture volume $\varphi^{Vol}_{i,j}$ agrees with batch sizes $b_{i,j,t}$ for every task-unit pair $(i,j) \in IJ^{\times dyn}$ and every $t \in T$.

$$-\mathrm{M}^{min}_{1,i,j}\left(1 - x_{i,j,t}\right) \leq b_{i,j,t} - \varphi^{Vol}_{i,j} \leq \mathrm{M}^{max}_{1,i,j}\left(1 - x_{i,j,t}\right), \forall (i,j) \in IJ^{\times dyn}, \forall t \in T \quad \text{(H.7)}$$

Detailed problem statistics showing the number of variables and constraints for case studies with $\eta = 50400$ seconds and $\eta = 100800$ seconds are shown in Tables H.7 and H.8, respectively. These statistics were obtained from solver SHOT[394] after performing a big-M reformulation of the problem.

Table H.7. Problem statistics (case study 1, $\eta = 50400\ s$)

Number of linear constraints	12686
Number of nonlinear constraints	7562
Total number of constraints	**20248**
Number of continuous variables	10628
Number of Boolean variables	52
Number of binary variables	405
Number of integer variables	10
Total number of variables	**11095**

Table H.8. Problem statistics (case study 1, $\eta = 100800\ s$)

Number of linear constraints	20282
Number of nonlinear constraints	7562
Total number of constraints	**27844**
Number of continuous variables	12504
Number of Boolean variables	52
Number of binary variables	801
Number of integer variables	10
Total number of variables	**13367**

H.4.3 Case study 2

Tables H.9-H.15 show the parameters required to define the second case study in chapter 8.

Table H.9. Maximum and minimum capacity of the units (case study 2)

Unit $j \in J$	$b_j^{min}\ [m^3]$	$b_j^{max}\ [m^3]$
$U1$	1	10
$U2$	1	5
$U3$	1	8
$U4$	1	20

Table H.10. Parameters related to states $m \in M$ (case study 2)

State $m \in M$	s_m^{init} $[m^3]$	s_m^{max} $[m^3]$	c_m^{mat} $[\$/m^3]$	c_m^{out} $[\$/m^3]$
$S1$	400	400	50	-
$S2$	400	400	150	-
$S3$	400	400	200	-
$S4$	0	100	-	-
$S5$	0	15	-	-
$S6$	0	50	-	-
$S7$	0	100	-	-
$S8$	0	400	-	300
$S9$	0	400	-	400

Table H.11. Parameters related to pairs $(i,j) \in IJ^{\times}$ (case study 2)

Pair $(i,j) \in IJ^{\times}$	$\tau_{i,j}^{P,min}$ $[h]$	$\tau_{i,j}^{P,max}$ $[h]$	$\tau_{i,j}^{P,fixed}$ $[h]$	$c_{i,j}^{fix}$ $[\$]$	$c_{i,j}^{var_1}$ $[\$/m^3]$
$(T1,U1)$	0.5	0.5	0.5	10	10
$(T2,U2)$	0.1	2	-	15	-
$(T2,U3)$	0.1	2	-	30	-
$(T3,U2)$	1	1	1	5	20
$(T3,U3)$	2.5	2.5	2.5	25	30
$(T4,U2)$	1	1	1	5	20
$(T4,U3)$	5	5	5	20	35
$(T5,U4)$	0.1	3	-	20	-

Table H.12. Reactor parameters in the dynamic models (case study 2)

Parameter	Value	Description
k_1 $[m^3/kmol\ h]$	4	Pre-exponential factor of reaction 1
k_2 $[m^3/h]$	0.8	Pre-exponential factor of reaction 2
Ea_1 $[kJ/kmol]$	6000	Activation energy of reaction 1
Ea_2 $[kJ/kmol]$	20000	Activation energy of reaction 2
ΔH_1 $[kJ/kmol]$	-30000	Enthalpy of reaction 1
ΔH_2 $[kJ/kmol]$	-10000	Enthalpy of reaction 2
C_{IN}^{A} $[kmol/m^3]$	10.62	Concentration of reactant A coming from state $S2$
C_{IN}^{B} $[kmol/m^3]$	20	Concentration of reactant B coming from state $S3$
C_{DES}^{C} $[kmol/m^3]$	4	Desired concentration of C in state $S5$
V_0 $[m^3]$	1	Initial volume of the semi-batch reactor
T_{R0} $[K]$	293.15	Initial reactor temperature
T_H $[K]$	323.15	Temperature of heating fluid
T_C $[K]$	293.15	Temperature of cooling fluid
R $[kJ/kmol\ K]$	8.31	Ideal gas constant
C_{PJ} $[kJ/kmol\ K]$	75	Heat capacity of reaction mixture
VJ_{U2} $[m^3]$	0.3	Volume (jacket side, reactor $U2$)
VJ_{U3} $[m^3]$	0.5	Volume (jacket side, reactor $U3$)
ρ_J $[kg/m^3]$	1000	Density (jacked side)

$C_{PJ}\ [kJ/kg\ K]$	4.2	Heat capacity (jacket side)
$T_{J0}\ [K]$	293.15	Initial temperature (jacket side)
$Ua_{U2}\ [kJ/h\ K]$	30000	Heat transfer coefficient (reactor $U2$)
$Ua_{U3}\ [kJ/h\ K]$	20000	Heat transfer coefficient (reactor $U3$)

Table H.13. Reactor variables in the dynamic models (case study 2)

Variable	Bounds	Description
$\chi_{i,j,t}^{A}(\psi)\ [kmol/m^3]$	[0,100]	Concentration of A
$\chi_{i,j,t}^{B}(\psi)\ [kmol/m^3]$	[0,100]	Concentration of B
$\chi_{i,j,t}^{C}(\psi)\ [kmol/m^3]$	[0,100]	Concentration of C
$\chi_{i,j,t}^{Vol}(\psi)\ [m^3]$	$[b_j^{min}, b_j^{max}]$	Volume of reaction mixture
$\chi_{i,j,t}^{TR}(\psi)\ [K]$	[293.15,323.15]	Reactor temperature
$\chi_{i,j,t}^{TJ}(\psi)\ [K]$	[293.15,323.15]	Jacket temperature
$\zeta_{i,j,t}^{A}(\psi)\ [kmol/m^3\ h]$	*	Reaction rate of A in reaction 1
$\zeta_{i,j,t}^{C}(\psi)\ [kmol/m^3\ h]$	*	Reaction rate of A in reaction 2
$\mu_{i,j,t}^{B}(\psi)\ [m^3/h]$	[0,5]	Feed rate of B
$\mu_{i,U2,t}^{hot}(\psi)\ [m^3/h]$	[0,8]	Flow of heating fluid (reactor $U2$)
$\mu_{i,U2,t}^{cold}(\psi)\ [m^3/h]$	[0,8]	Flow of cooling fluid (reactor $U2$)
$\mu_{i,U3,t}^{hot}(\psi)\ [m^3/h]$	[0,10]	Flow of heating fluid (reactor $U3$)
$\mu_{i,U3,t}^{cold}(\psi)\ [m^3/h]$	[0,10]	Flow of cooling fluid (reactor $U3$)
$\varphi_{i,j,t}^{P}\ [h]$	$[0, \tau_{i,j}^{P,max}]$	Reaction time
$\varphi_{i,j,t}^{A0}\ [kmol/m^3]$	$[0, C_{IN}^{A}]$	Initial concentration of A
$\varphi_{i,j,t}^{B0}\ [kmol/m^3]$	$[0, C_{IN}^{B}]$	Initial concentration of B

*: Defined based on the bounds imposed over other variables

Table H.14. Distillation parameters in the dynamic models (case study 2)

Parameter	Value	Description
α^R	2.5	Relative volatility
NT	12	Total number of stages (with condenser and reboiler)
NIN	10	Total number of internal stages
$x^{INIT}\ [kmol/kmol]$	0.8	Initial liquid composition
$x^{DES}\ [kmol/kmol]$	0.95	Desired product composition
$H^T\ [m^3]$	0.01	Tray hold-up
$H^C\ [m^3]$	0.1	Condenser hold-up
r^{ρ}	0.01	Liquid/vapor density constant ratio

Table H.15. Distillation variables in the dynamic models (case study 2)

Variable	Bounds	Description
$\chi_{i,j,t}^{HB}(\psi)[m^3]$	$[0, b_j^{max}]$	Reboiler hold-up
$\chi_{i,j,t}^{HBx}(\psi)[m^3]$	$[0, b_j^{max}]$	Reboiler hold-up per component
$\chi_{i,j,t}^{HR}(\psi)[m^3]$	$[0, b_j^{max}]$	Receiver hold-up
$\chi_{i,j,t}^{HRx}(\psi)[m^3]$	$[0, b_j^{max}]$	Receiver hold-up per component
$\chi_{i,j,t,n}^{x}(\psi)$	[0,1]	Liquid Molar fraction
$\zeta_{i,j,t,1}^{y}(\psi)$	[0,1]	Vapor molar fraction
$\zeta_{i,j,t}^{D}(\psi)\ [m^3/h]$	[0,1000]	Distillate rate
$\zeta_{i,j,t}^{x,avg}(\psi)$	[0,1]	Receiver molar fraction
$\mu_{i,j,t}^{V}(\psi)\ [m^3/h]$	[0,1000]	Boil-up rate
$\mu_{i,j,t}^{R}(\psi)\ [m^3/h]$	[0,1000]	Reflux rate
$\varphi_{i,j,t}^{HB0}$	$[0, b_j^{max}]$	Initial reboiler hold-up
$\varphi_{i,j,t}^{P}$	$[0, \tau_{i,j}^{P,max}]$	Separation time

*: Defined based on the bounds imposed over other variables

The ordered discrete decisions of this problem are shown in Table H.16. In this case, the reformulation with external variables allowed to reformulate 26 Boolean variables into 8 external variables $e_{i,j}$, out of which 3 variables are used by MLD-BD to optimize the processing time of the different reactions and the separation process considered in this network.

Table H.16. Domain of ordered discrete decisions (case study 2)

Variable	Domain	Variable	Domain
$n_{T1,U1}$	$\{0,1,\dots,7\}$	$e_{T1,U1}$	$\{2\}$
$n_{T2,U2}$	$\{0,1,\dots,15\}$	$e_{T2,U2}$	$\{1,2,\dots,6\}$
$n_{T2,U3}$	$\{0,1,\dots,15\}$	$e_{T2,U3}$	$\{1,2,\dots,6\}$
$n_{T3,U2}$	$\{0,1,\dots,5\}$	$e_{T3,U2}$	$\{3\}$
$n_{T3,U3}$	$\{0,1\}$	$e_{T3,U3}$	$\{8\}$
$n_{T4,U2}$	$\{0,1,\dots,5\}$	$e_{T4,U2}$	$\{3\}$
$n_{T4,U3}$	$\{0,1\}$	$e_{T4,U3}$	$\{15\}$
$n_{T5,U4}$	$\{0,1,\dots,15\}$	$e_{T5,U4}$	$\{1,2,\dots,9\}$

Based on the above, it is possible to formulate the constraints that define this case study, in addition to those already shown in section 8.1.1. A mixed-time formulation is considered for this case study, i.e., variable processing times are allowed to vary continuously in the discrete-time scheduling grid, i.e., $\tau_{i,j,t}^{P} \leq \min\{\hat{\delta}\tau_{i,j}, \tau_{i,j}^{P,max}\}$. Units $U2$ and $U3$ are reactors with different capacities that operate in semi-batch mode when executing task $T2$. The exothermic-irreversible reactions involved in $T2$ are

$A + B \rightarrow C$ and $C \rightarrow D$, where C is the desired product, and reactant B is assumed to be dosed through the operation of the semi-batch process. Reactant A is assumed to be coming from state $S2$, whereas reactant B is assumed to be coming from state $S3$. Unit $U4$ is a batch distillation column that executes batch distillation task $T5$. To differentiate between reaction and distillation stages, set $IJ^{\times dyn}$ is expressed as $IJ^{\times dyn} = IJ^{\times rx} \cup IJ^{\times sep}$, where $IJ^{\times rx} = \{(T2, U2), (T2, U3)\}$ and $IJ^{\times sep} = \{(T5, U4)\}$. For the distillation column, the set of stages is defined as $NET = \{1,2, \dots, NT\}$ (index n) with stages numbered from bottom to top. The distillation model constraints include the following assumptions: 1) every component has the same molecular weight, 2) vapor hold-up is negligible, 3) the heat of vaporization of the feed components are equal, 4) the system can be treated as a pseudo-binary system, 5) heat effects such as heats of solution and heat transfer to and from the distillation column are negligible, 6) reflux drum and tray hold-ups are constant, 7) phase equilibrium attained at each tray. This distillation model was adapted from reference[395]. The differential-algebraic equations that define dynamic feasible region DYN are shown in equation (H.8A)-(H.8R).

Reaction rate of A in $A + B \rightarrow C$ (H.8A)

$$\zeta^{A}_{i,j,t} = -k_1 \exp\left(\frac{-Ea_1}{R\chi^{TR}_{i,j,t}}\right)\chi^{A}_{i,j,t}\chi^{B}_{i,j,t}, \forall (i,j) \in IJ^{\times rx}, \forall t \in T$$

Reaction rate of C in $C \rightarrow D$ (H.8B)

$$\zeta^{C}_{i,j,t} = -k_2 \exp\left(\frac{-Ea_2}{R\chi^{TR}_{i,j,t}}\right)\chi^{C}_{i,j,t}, \forall (i,j) \in IJ^{\times rx}, \forall t \in T$$

Concentration balance of A (H.8C)

$$\dot{\chi}^{A}_{i,j,t} = \varphi^{P}_{i,j,t}\left(\zeta^{A}_{i,j,t} - \frac{\mu^{B}_{i,j,t}}{\chi^{Vol}_{i,j,t}}\chi^{A}_{i,j,t}\right),\ \chi^{A}_{i,j,t}(\psi = 0) = \varphi^{A0}_{i,j,t}, \forall (i,j) \in IJ^{\times rx}, \forall t \in T$$

Concentration balance of B (H.8D)

$$\dot{\chi}^{B}_{i,j,t} = \varphi^{P}_{i,j,t}\left(\zeta^{A}_{i,j,t} + \frac{\mu^{B}_{i,j,t}}{\chi^{Vol}_{i,j,t}}\left(C^{B}_{IN} - \chi^{B}_{i,j,t}\right)\right), \chi^{B}_{i,j,t}(\psi = 0) = \varphi^{B0}_{i,j,t},$$

$$\forall (i,j) \in IJ^{\times rx}, \forall t \in T$$

Concentration balance of C (H.8E)

$$\dot{\chi}^{C}_{i,j,t} = \varphi^{P}_{i,j,t}\left(-\zeta^{A}_{i,j,t} + \zeta^{C}_{i,j,t} - \frac{\mu^{B}_{i,j,t}}{\chi^{Vol}_{i,j,t}}\chi^{C}_{i,j,t}\right),\ \chi^{C}_{i,j,t}(\psi = 0) = 0, \forall (i,j) \in IJ^{\times rx},$$

$$\forall t \in T$$

Change in reactive mixture volume (H.8F)

$$\dot{\chi}^{Vol}_{i,j,t} = \varphi^{P}_{i,j,t}\left(\mu^{B}_{i,j,t}\right),\ \chi^{Vol}_{i,j,t}(\psi = 0) = V_0, \forall (i,j) \in IJ^{\times rx}, \forall t \in T$$

Energy balance (reactor temperature) (H.8G)

$$\dot{\chi}^{TR}_{i,j,t} = \frac{\varphi^{P}_{i,j,t}}{\varphi^{A}_{i,j,t}+\varphi^{B}_{i,j,t}}\left(\frac{Ua\left(\chi^{TJ}_{i,j,t}-\chi^{TR}_{i,j,t}\right)}{V_0 C_P} - \frac{C^{B}_{IN}\mu^{B}_{i,j,t}\left(\chi^{TR}_{i,j,t}-T_{B,IN}\right)}{V_0} + \frac{\chi^{Vol}_{i,j,t}}{V_0 C_P}\left(\Delta H_1 \zeta^{A}_{i,j,t} + \Delta H_2 \zeta^{C}_{i,j,t}\right)\right),$$

$$\chi^{TR}_{i,j,t}(\psi = 0) = T_{R0}, \forall (i,j) \in IJ^{\times rx}, \forall t \in T$$

Energy balance (jacket temperature) (H.8H)

$$\dot{\chi}^{TJ}_{i,j,t} = \varphi^{P}_{i,j,t}\left(\frac{\mu^{hot}_{i,j,t}\left(T_H-\chi^{TJ}_{i,j,t}\right)+\mu^{cold}_{i,j,t}\left(T_C-\chi^{TJ}_{i,j,t}\right)}{VJ_j} + \frac{Ua_j\left(\chi^{TR}_{i,j,t}-\chi^{TJ}_{i,j,t}\right)}{VJ_j \rho_J C_{PJ}}\right), \chi^{TJ}_{i,j,t}(\psi = 0) = T_{J0},$$

$$\forall (i,j) \in IJ^{\times rx}, \forall t \in T$$

Reboiler balance (H.8I)

$$\dot{\chi}^{HB}_{i,j,t} = \varphi^{P}_{i,j,t}\left(\mu^{R}_{i,j,t} - r^{\rho}\mu^{V}_{i,j,t}\right), \chi^{HB}_{i,j,t}(\psi = 0) = \varphi^{HB0}_{i,j,t}, \forall (i,j) \in IJ^{\times sep}, \forall t \in T$$

Reboiler balance per component (H.8J)

$$\dot{\chi}^{HBx}_{i,j,t} = \varphi^{P}_{i,j,t}\left(\mu^{R}_{i,j,t}\chi^{x}_{i,j,t,2} - r^{\rho}\mu^{V}_{i,j,t}\zeta^{y}_{i,j,t,1}\right), \chi^{HBx}_{i,j,t}(\psi = 0) = \varphi^{HB0}_{i,j,t}x^{INIT},$$

$$\forall (i,j) \in IJ^{\times sep}, \forall t \in T$$

Definition of reboiler hold-up per component (H.8K)

$$\chi^{HBx}_{i,j,t} = \chi^{HB}_{i,j,t}\chi^{x}_{i,j,t,1}\ , \forall (i,j) \in IJ^{\times sep}, \forall t \in T$$

Receiver balance (H.8L)

$$\dot{\chi}^{HR}_{i,j,t} = \varphi^{P}_{i,j,t}\zeta^{D}_{i,j,t}, \chi^{HR}_{i,j,t}(\psi = 0) = 0, \forall (i,j) \in IJ^{\times sep}, \forall t \in T$$

Receiver balance per component (H.8M)

$$\dot{\chi}_{i,j,t}^{HRx} = \varphi_{i,j,t}^{P}\zeta_{i,j,t}^{D}\chi_{i,j,t,NT}^{x}, \chi_{i,j,t}^{HRx}(\psi = 0) = 0, \forall (i,j) \in IJ^{\times sep}, \forall t \in T$$

Definition of receiver composition (H.8N)

$$\zeta_{i,j,t}^{x,avg} = \frac{\chi_{i,j,t}^{HRx}}{\chi_{i,j,t}^{HR}}, \psi > 0, \forall (i,j) \in IJ^{\times sep}, \forall t \in T$$

Condenser balance per component (H.8O)

$$H^{C}\dot{\chi}_{i,j,t,NT}^{x} = \varphi_{i,j,t}^{P}\Big(r^{\rho}\mu_{i,j,t}^{V}\zeta_{i,j,t,NT-1}^{y} - \big(\mu_{i,j,t}^{R} + \zeta_{i,j,t}^{D}\big)\chi_{i,j,t,NT}^{x}\Big),$$

$$\chi_{i,j,t,NT}^{x}(\psi = 0) = x^{INIT}, \forall (i,j) \in IJ^{\times sep}, \forall t \in T$$

Tray balance per component (H.8P)

$$H^{T}\dot{\chi}_{i,j,t,n}^{x} = \varphi_{i,j,t}^{P}\left(\mu_{i,j,t}^{R}\big(\chi_{i,j,t,n+1}^{x} - \chi_{i,j,t,n}^{x}\big) + r^{\rho}\mu_{i,j,t}^{V}\Big(\zeta_{i,j,t,n-1}^{y} - \zeta_{i,j,t,n}^{y}\Big)\right),$$

$$\chi_{i,j,t,n}^{x}(\psi = 0) = x^{INIT}, \forall (i,j) \in IJ^{\times sep},\ \forall t \in T, \forall n \in NET\backslash\{1, NT\}$$

Equilibrium relationship (H.8Q)

$$\zeta_{i,j,t,n}^{y} = \frac{\alpha^{R}\chi_{i,j,t,n}^{x}}{1+(\alpha^{R}-1)\chi_{i,j,t,n}^{x}}, \forall (i,j) \in IJ^{\times sep}, \forall t \in T, \forall n \in NET$$

Relationship between reflux rate (R), boil-up rate (V) and distillate rate (D) (H.8R)

$$\mu_{i,j,t}^{R} = r^{\rho}\mu_{i,j,t}^{V} - \zeta_{i,j,t}^{D}, \forall (i,j) \in IJ^{\times sep}, \forall t \in T$$

Additional linking constraints required to couple scheduling and dynamic decisions denoted by feasible region $LINK$ are shown in equations (H.9A)-(H.9G).

$$-\mathrm{M}_{1,i,j,t}^{min}\big(1 - x_{i,j,t}\big) \le b_{i,j,t} - \chi_{i,j,t}^{Vol}(\psi = 1) \le \mathrm{M}_{1,i,j,t}^{max}\big(1 - x_{i,j,t}\big), \quad \text{(H.9A)}$$

$$\forall (i,j) \in IJ^{\times rx}, \forall t \in T$$

$$-\mathrm{M}_{2,i,j,t}^{min}\big(1 - x_{i,j,t}\big) \le \rho_{T2,\mathrm{S2}}^{-}\chi_{i,j,t}^{Vol}(\psi = 1) - \frac{\varphi_{i,j,t}^{A0}}{C_{IN}^{A}}V_0 \le \mathrm{M}_{2,i,j,t}^{max}\big(1 - x_{i,j,t}\big), \quad \text{(H.9B)}$$

$$\forall (i,j) \in IJ^{\times rx}, \forall t \in T$$

$$-\mathrm{M}_{3,i,j,t}^{min}\left(1-x_{i,j,t}\right) \leq \left(\rho_{T2,\mathrm{S3}}^{-}-1\right)\chi_{i,j,t}^{Vol}(\psi=1)-\left(\frac{\varphi_{i,j,t}^{B0}}{C_{IN}^{B}}-1\right)V_0 \leq \mathrm{M}_{3,i,j,t}^{max}\left(1-x_{i,j,t}\right), \quad \text{(H.9C)}$$

$$\forall(i,j)\in IJ^{\times rx}, \forall t\in T$$

$$-\mathrm{M}_{4,i,j,t}^{min}\left(1-x_{i,j,t}\right) \leq \chi_{i,j,t}^{C}(\psi=1)-C_{DES}^{C} \leq \mathrm{M}_{4,i,j,t}^{max}\left(1-x_{i,j,t}\right), \quad \text{(H.9D)}$$

$$\forall(i,j)\in IJ^{\times rx}, \forall t\in T$$

$$-\mathrm{M}_{5,i,j,t}^{min}\left(1-x_{i,j,t}\right) \leq b_{i,j,t}-\left(\varphi_{i,j,t}^{HB0}+NIN(H^{T})+H^{C}\right) \leq \mathrm{M}_{5,i,j,t}^{max}\left(1-x_{i,j,t}\right), \quad \text{(H.9E)}$$

$$\forall(i,j)\in IJ^{\times sep}, \forall t\in T$$

$$-\mathrm{M}_{6,i,j,t}^{min}\left(1-x_{i,j,t}\right) \leq \chi_{i,j,t}^{HR}(\psi=1)-\rho_{i,\mathrm{S9}}^{+}\left(\varphi_{i,j,t}^{HB0}+NIN(H^{T})+H^{C}\right) \leq \mathrm{M}_{6,i,j,t}^{max}\left(1-x_{i,j,t}\right), \quad \text{(H.9F)}$$

$$\forall(i,j)\in IJ^{\times sep}, \forall t\in T$$

$$x^{DES}-\zeta_{i,j,t}^{x,avg}(\psi=1) \leq \mathrm{M}_{7,i,j,t}^{max}\left(1-x_{i,j,t}\right), \quad \forall(i,j)\in IJ^{\times sep}, \forall t\in T \quad \text{(H.9G)}$$

Detailed problem statistics showing the number of variables and constraints for this case study are shown in Table H.17. These statistics were obtained from solver SHOT[394] after performing a big-M reformulation of the problem.

Table H.17. Problem statistics (case study 2)

Number of linear constraints	154904
Number of nonlinear constraints	265346
Total number of constraints	**420250**
Number of continuous variables	318637
Number of Boolean variables	26
Number of binary variables	128
Number of integer variables	8
Total number of variables	**318799**

H.5 Supplementary results for case study 1

H.5.1 Case study with η=50400 seconds

Analyzing the first iteration in MLD-BD, MLD-BD(p) and MLD-BD(p,h) allows to perform a fair comparison between the algorithms tested, given that they were all initialized from the same feasible solution. The most computational time demanding steps each iteration are: solution of subproblems (Step 1 in Figure 8.1), generation of Benders cuts (Step 2 in Figure 8.1), and solution of master problem (Step 3 in Figure 8.1). When comparing LD-BD with MLD-BD, the main difference in computational

time is an increase of one order of magnitude in the time needed for cut generation, from 0.0037 seconds to 0.049 seconds, respectively. Despite the additional time needed for cut generation in MLD-BD, this time remains relatively short compared to the subproblem and master problem solution times of 556.44 seconds and 0.41 seconds, respectively. This also indicates that the most time-consuming step in MLD-BD for the current case study is the solution of subproblems. MLD-BD(p) and MLD-BD(p,h) returned similar cut generation (i.e., 0.071 seconds and 0.052 seconds) and master problem solution times (i.e., 0.33 seconds and 0.31 seconds) compared to MLD-BD. However, the main improvements in CPU-time effort can be observed in the solution of subproblems. Thanks to the pruning strategy, MLD-BD(p) reduced the MLD-BD subproblem solution time by 32% from 556.44 seconds to 377.10 seconds. Similarly, the heuristic in MLD-BD(p,h) reduced the MLD-BD(p) subproblem CPU-time by 42% from 377.10 seconds to 220.10 seconds.

H.5.2 Case study with η=100800 seconds

Figure H.1 illustrates the differences obtained in the manipulated variables and concentration profiles when comparing the SIA initialization and the MLD-BD solution for the long scheduling horizon instance of case study 1. Figure H.1 corresponds to the execution of reaction $R2$ in the small reactor. Plots illustrating the execution of the remaining reactions in the available reactors show a similar behavior and are not shown for brevity. The MLD-BD solution shows a lengthier processing time of 12600 seconds, compared to 10800 seconds in the SIA initialization. The SIA solution results in a shorter processing time (10800 seconds) but incurs higher costs due to increased heating and cooling requirements (see Figures H.1A and H.1B). This control strategy aims to accelerate the reaction rate at the beginning of the operation, which results in saturation of the heating flow at its upper bound at the beginning of the operation. Conversely, the MLD-BD solution extends the reaction time by half an hour to minimize heating and cooling flows, while still meeting the product quality requirement at the end of the batch (see Figures H.1C and H.1D).

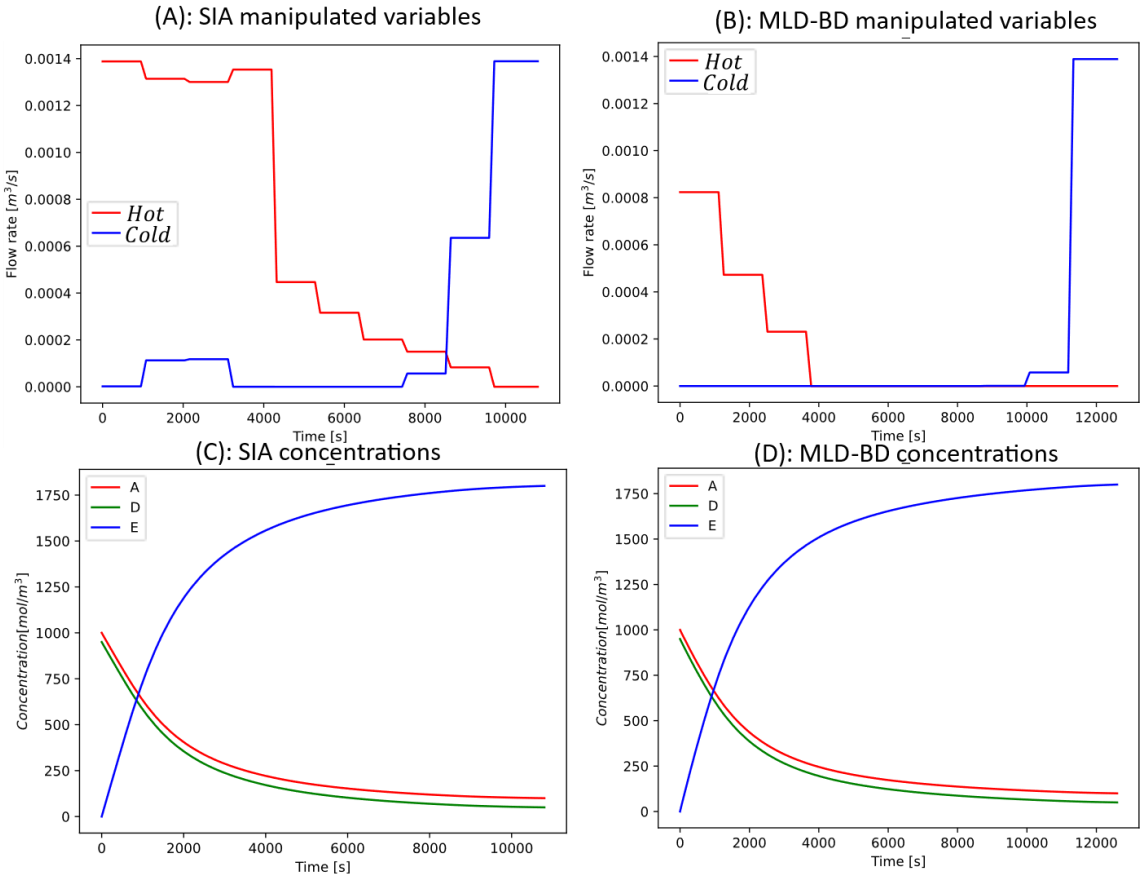


Figure H.1. Control actions and concentrations for the MLD-BD approach (A and B) and the SIA initialization (C and D).