

Graph Property Testing and the Container Method

by

Cameron Seth

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2026

© Cameron Seth 2026

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: Dana Ron
Professor Emerita, School of Electrical and Computer Engineering,
Tel-Aviv University

Supervisor(s): Eric Blais
Associate Professor, Cheriton School of Computer Science,
University of Waterloo

Internal Members: Sepehr Assadi
Associate Professor, Cheriton School of Computer Science,
University of Waterloo

Elena Grigorescu
Professor, Cheriton School of Computer Science,
University of Waterloo

Internal-External Member: Luke Postle
Professor, Department of Combinatorics and Optimization,
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

This thesis contains material based on three previously published papers.

- The material in [Chapter 4](#) is based on part of the paper [\[BS25\]](#), which is joint work with Eric Blais.
- The material in [Chapter 5](#) is based on part of the paper [\[BS24\]](#), which is joint work with Eric Blais.
- The material in [Chapter 6](#) is based on the paper [\[Set25\]](#), which I am the sole author on.

Abstract

Graph property testing algorithms aim to distinguish between graphs that have a specific property and graphs that are far from the property by inspecting a small random portion of the graph. This thesis studies graph property testing in the dense graph model, where graphs are represented by their adjacency matrices, and a graph is ϵ -far from a property if an ϵ -fraction of the entries of its adjacency matrix must be modified to make the graph have the property. A central goal in the dense graph model is to determine the minimum size subgraph that must be sampled to test natural graph properties.

This thesis develops a new framework for analyzing graph property testing algorithms using the graph and hypergraph container method. Although the container method has become a powerful tool throughout extremal combinatorics, prior to this thesis it had not been used to analyze property testing algorithms. We establish a connection between graph property testing and the container method by showing that suitable container lemmas imply strong upper bounds on the sample complexity of canonical property testers for a number of natural properties.

To demonstrate the framework, we develop new graph and hypergraph container lemmas and apply them to three classic property testing problems. First, we prove a new graph container lemma for graphs that are far from having a ρn independent set, and use it to give the optimal, up to logarithmic factors, upper bound on the sample complexity of testing the property of having a ρn independent set. Second, we prove a new hypergraph container lemma for constraint satisfaction problems. This yields a new upper bound on the sample complexity for testing satisfiability, and is the first upper bound that is polynomial in the constraint arity, alphabet size, and distance parameter simultaneously. Finally, we introduce a new container lemma for sparse subgraphs and apply it to tolerant testing the ρn independent set property, giving an optimal sample complexity bound, again up to logarithmic factors, and demonstrating that container methods can be extended beyond independent sets to sparse subgraphs.

Together, these results demonstrate that the container method provides a general framework for analyzing efficient property testing algorithms.

Acknowledgements

I would first like to thank my supervisor Eric Blais. I first met Eric eleven years ago as an undergraduate student in one of his courses. Since then, he has been my supervisor, collaborator and mentor. I am always inspired by his approach to research and his ability to find the intuition behind complex ideas. I am very grateful for our many discussions, both about specific research problems and about how to approach research more broadly.

Along with my supervisor, I would like to thank my coauthors Yuichi Yoshida, Gaia Carenini, Elena Grigorescu and Shreya Nasa. I have thoroughly enjoyed working with each of you and have learned a great deal from our collaborations. I look forward to working on many more problems together.

I would also like to thank Sepehr Assadi for many helpful discussions and suggestions both on this work and on research in general. In particular, one of his suggestions led to the observation that a non-canonical tester for the independent set property follows naturally from the tolerant testing result, which appears as a corollary in this thesis.

In addition to Sepehr Assadi, Elena Grigorescu and Eric Blais, my committee included Luke Postle and Dana Ron. I thank all of my committee members for their feedback on my thesis and for the useful discussions during my defence.

I would like to thank all the members of the Algorithms and Complexity Group at University of Waterloo. I have greatly appreciated the many conversations, seminars, lunches, and collaborations over the years.

I would like to thank the University of Waterloo. The university has been a second home to me ever since I began my undergraduate degree here in 2011. Even before that, I participated in Math Circles and Engineering Science Quest (ESQ) as a child. The fact that both of my parents and all four of my siblings also attended the University of Waterloo speaks to how special this place has been to my family.

And finally, I thank my family. Thank you for making this journey so special.

“Which is more important,” asked Big Panda, “the journey or the destination?”

“The company,” said Tiny Dragon.

— James Norbury, *Big Panda and Tiny Dragon*

Table of Contents

Examining Committee Membership	ii
Author's Declaration	iii
Statement of Contributions	iv
Abstract	v
Acknowledgements	vi
List of Figures	x
1 Introduction	1
1.1 Graph Property Testing	3
1.2 Container Method	4
1.3 Testing the Independent Set Property	11
1.4 Testing Satisfiability	12
1.5 Tolerant Testing the Independent Set Property	15
1.6 Structure of this Thesis	20
2 Background and Prior Work	21
2.1 Graph Property Testing	21
2.2 Brief History of the Container Method	28
2.3 Useful Bounds	33

3	A Warmup Application of the Container Method	35
3.1	Container Procedure and Container Lemma	35
3.2	Proof of Theorem	38
4	Testing Independent Sets	41
4.1	Proof Overview	41
4.2	The Fingerprint and Container Procedure	42
4.3	Container Shrinking Lemma	43
4.4	Proof of Graph Container Lemma	45
4.5	Proof of Theorem 1.3	47
5	Testing Satisfiability	50
5.1	Proof Overview	50
5.2	Container Procedure	53
5.3	Basic Properties	56
5.4	Hypergraph Container Lemma	58
5.5	Proof of Theorem 1.5	62
5.6	Proofs of Corollary 1.7 and Corollary 1.8	63
6	Tolerant Testing Independent Sets	66
6.1	Container Lemma Statement	66
6.2	Proof Overview	68
6.3	Fingerprint and Container Procedures	71
6.4	An Initial Containment Lemma	72
6.5	Revision Lemma	77
6.6	Proof of Graph Container Lemma	79
6.7	A Tolerant Tester for the Independent Set Property	82
6.8	Application to Non-Canonical Testers	86

7	Future Directions	89
7.1	Testing and Tolerant Testing more Properties	89
7.2	Non-Canonical Testing	92
7.3	Combinatorial Applications of our new Container Lemmas	93
7.4	Characterizations of Properties Testable with the Container Method	94
	References	96

List of Figures

1.1	A Graph on 8 vertices with an independent set of size 3.	1
1.2	This graph has many independent sets, three are highlighted on the left side. However, all independent sets in this graph can be covered by three containers as shown on the right.	5
1.3	Testing the ρ -INDEPSET property. In the yes case (left), the sample S includes a sufficient number of vertices from the large independent set with reasonable probability. The challenging part is showing that in the no case (right), the sample does not include a $\rho S $ independent set.	7
1.4	Alice is thinking about the independent set I . How can she tell Bob about her independent set by only sharing one vertex?	9
1.5	If Alice sends her highest degree vertex v (highlighted in yellow), Bob can remove the neighbours (left) and higher degree vertices (right) and determine what Alice's independent set is.	9
1.6	Given an independent set I , multiple steps of the process constructs a fingerprint F (the set of vertices colored in yellow) which can be used to construct a container $C(F)$ (colored in blue) by repeatedly removing the neighbours and higher degree vertices of the fingerprint vertices.	10
1.7	Encoding game for independent sets (left) vs sparse subgraphs (right). In the sparse subgraph setting, selecting the highest degree vertex in J and removing neighbours removes too much of J from the container.	19
5.1	An illustration of the two hyperedges formed from the constraint that the variables x_1 , x_2 , and x_3 must not all be equal.	52

7.1	An example of an independent set star. In the center is a core of vertices forming an independent. The vertices outside of the core have no edges to the core. There are no requirements on the edges between vertices on the outside.	93
-----	--	----

Chapter 1

Introduction

We begin with a simple problem about independent sets. An independent set in a graph is a set of vertices such that no two vertices are adjacent. Given a graph G on n vertices and a parameter $\rho \in (0, 1)$, can you determine if G has an independent set of size ρn ?

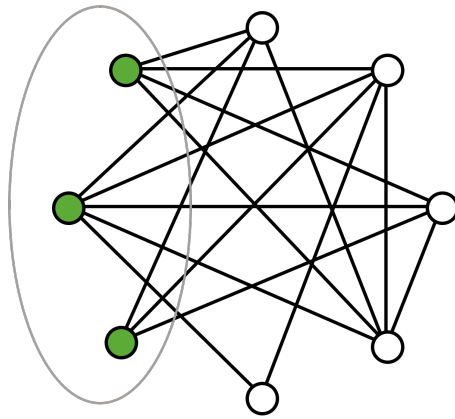


Figure 1.1: A Graph on 8 vertices with an independent set of size 3.

For small graphs, such as the example graph in [Fig. 1.1](#), you can simply do a brute force check of all possible subsets of vertices to find the largest independent set in the graph,

which in Fig. 1.1 gives the independent set of size 3 highlighted in green. However, for general values of ρ and n , this brute force algorithm requires exponential time in n . Is it possible to solve the problem more efficiently?

Unfortunately, it is believed that you cannot do much better than the brute force exponential time algorithm. In fact, the independent set problem is one of the classic NP-Complete problems studied by Karp [Kar72], and as such it is believed that it cannot be solved in time which is polynomial in n . Further, many approximations and restricted variants of the problem are also believed to be hard to solve. For example, it is NP-Hard to approximate the size of the maximum independent set up to factor $n^{1-\delta}$ for any constant $\delta > 0$ [Has96, Zuc07].¹ Even for the restricted setting where the algorithm must only distinguish between a random graph $G(n, 1/2)$ and a random graph with a planted ρn -independent set, no polynomial time algorithm is known when $\rho < 1/\sqrt{n}$ (see, e.g., [BHK⁺19, AKS98]). Despite little algorithmic progress in solving the independent set problem and its variants, there is one very natural notion of approximation that is very easy to solve: the property testing version.

Goldreich, Goldwasser and Ron first introduced the notion of graph property testing in 1998 [GGR98]. In that work they considered a relaxed version of the independent set problem: instead of distinguishing between graphs that have an independent set of size ρn and graphs that don't, the goal is to distinguish between graphs that have an independent set of size ρn and graphs where every induced subgraph on ρn vertices has at least ϵn^2 edges (i.e. at least a 2ϵ -fraction of the entries of the adjacency matrix must be modified to make the graph have a ρn independent set). Surprisingly, they showed that this problem can be solved by taking a random set $S \subseteq V$ of $\tilde{O}(\rho/\epsilon^4)$ and only inspecting the induced subgraph $G[S]$.²

Theorem 1.1 (Informal, [GGR98]). *Given a graph G , it is possible to distinguish, with probability at least $2/3$, between the case that G has an independent set of size ρn and the case that every induced subgraph on ρn vertices in G has at least ϵn^2 edges by taking a random set $S \subseteq V$ of $\tilde{O}(\rho/\epsilon^4)$ vertices and accepting if and only if the induced subgraph $G[S]$ has an independent set of size roughly $\rho|S|$.*

In other words, not only is the property testing version of the problem dramatically easier than the original independent set problem, it can be solved by looking at only a constant sized random portion of the graph!

¹Note that the independent set problem is equivalent to the clique problem by considering the complement graph. Here and throughout we phrase all results in terms of independent sets.

²Here and throughout we use $\tilde{O}(\cdot)$ and $\tilde{\Omega}(\cdot)$ to hide terms that are polylogarithmic in the argument.

Along with [Theorem 1.1](#), Goldreich, Goldwasser and Ron studied similar variants of a number of other classic graph problems, including bipartiteness, k -colorability, ρ -cut and many other graph partition properties, which initiated the study of graph property testing. Since then, a central goal in graph property testing has been to study the property testing variants of a number of natural graph problems, and determining what is the smallest size of an induced subgraph one needs to inspect in the graph in order to solve the property testing variant of the problem.

The central goal of this thesis is to develop a general framework for proving efficient property testers using the *graph and hypergraph container method*, which is a powerful combinatorial tool for characterizing independent sets in graphs and hypergraphs (see [Section 1.2](#) or the surveys [\[BMS18, Sam15\]](#) for an introduction to the container method). To do this, we establish a new connection between property testing and the container method. We show how this connection can be used to analyze the sample complexity of property testing algorithms, and illustrate the framework through three applications: testing the property of having a ρn independent set, testing satisfiability in constraint satisfaction problems, and tolerant testing of the property of having a ρn independent set.

In the rest of this chapter, we state the basic definitions in graph property testing (see [Chapter 2](#) for full definitions), then we discuss the container method and give some intuition on how it can be used for property testing, and then discuss our three applications.

1.1 Graph Property Testing

We start by giving some of the basic definitions in graph property testing (see [Chapter 2](#) for more details on the definitions and relevant prior work). Introduced by Goldreich, Goldwasser and Ron, graph property testing algorithms aim to distinguish graphs that have a specific property from graphs that are far from the property by inspecting a small fraction of the graph [\[GGR98\]](#). The definition of *far* depends on the model, and the most commonly studied model is the dense graph model. In this model, a graph G on n vertices is represented by an $n \times n$ adjacency matrix, and we say a (simple undirected) graph G on n vertices is ϵ -far from a graph property Π if we need to add or remove at least ϵn^2 edges from G to obtain a graph that has the property Π .

A *canonical ϵ -tester* for Π with sample cost s is a bounded-error³ randomized algorithm that samples a set S of s vertices from an unknown graph G , examines the induced subgraph

³By *bounded error* we mean there exist absolute constants $\delta_1 > \delta_2$ such that if G has property Π , then the algorithm accepts with probability at least δ_1 , and if G is ϵ -far from Π , then the algorithm accepts with probability at most δ_2 .

$G[S]$, and based only on this local view of the graph can distinguish between the case where G has property Π and the case where it is ϵ -far from Π . The *sample complexity* of Π , denoted $\mathcal{S}_{\Pi}(n, \epsilon)$, is the minimum value s for which there is a canonical ϵ -tester for Π with sample cost s .

Remarkably, many natural graph properties admit algorithms whose sample complexity depends only on the distance parameter ϵ , and is independent of the size of the graph (for example, [Theorem 1.1](#) for the ρ -INDEPSET property). A central goal in graph property testing is to determine the minimum sample complexity required to test natural graph properties. We approach this question using the graph container method.

We note that the *query complexity* is another commonly studied measure of the complexity of graph property testing algorithms. In general, a non-canonical tester can query arbitrary pairs of vertices $u, v \in V$ to determine if $(u, v) \in E$ or not, and the query complexity is the maximum number of edge queries the tester makes during its execution. For any graph property the query complexity and sample complexity are related by at most a quadratic factor [[GT03](#)], and so for this thesis we focus on sample complexity (with the exception of [Corollary 1.11](#)). See [Chapter 2](#) for further discussion of sample complexity and query complexity.

1.2 Container Method

In this section we describe the container method and give some intuition on how it can be applied to analyze graph property testing.

The container method is a combinatorial tool for characterizing independent sets in graphs and hypergraphs with specific properties. In general, a graph on n vertices can have up to 2^n independent sets. The container method can be used to show that, in many cases, while a graph may have many independent sets, the layout of the independent sets in the graph has some structure which can be useful for many applications. In particular, for many natural classes of graphs, such as regular graphs, nearly regular graphs, random graphs, or graphs that are sufficiently dense, graph container lemmas show that for any graph $G = (V, E)$ in the class, there exists a relatively small collection of *containers*, each of which is a subset of V and of size much smaller than $|V|$, such that every independent set is contained in at least one of these containers.

An Example. To illustrate how the container method works, consider the following simple example. The graph in [Fig. 1.2](#) has more than 20 independent sets, three of which

are highlighted. Despite having many independent sets, we can draw three *containers*, see the right hand side of Fig. 1.2, that cover all the independent sets. In particular, each container is a subset of vertices and we have that every independent set is a subset of one of these containers.

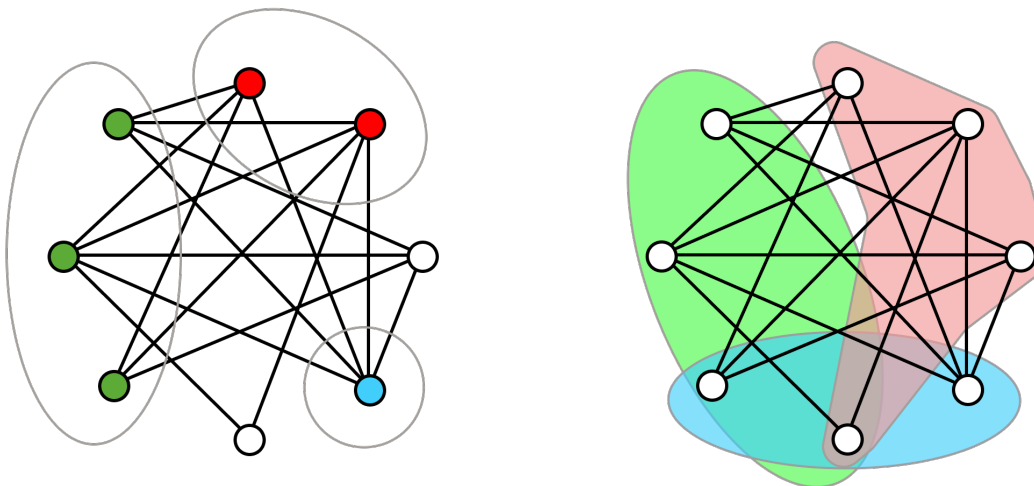


Figure 1.2: This graph has many independent sets, three are highlighted on the left side. However, all independent sets in this graph can be covered by three containers as shown on the right.

The container method is often applied in the form of a *container lemma*, which roughly says that if a graph satisfies specific properties then there exists a collection of containers such that every independent set is a subset of one of the containers. In order for such a container lemma to be meaningful the containers should each be of size strictly less than n (we can trivially cover all independent sets with the container V) and there should not be too many of them (we can trivially cover all independent sets using 2^n containers).

The first application of the container method was by Kleitman and Winston [KW82] to count the number of square-free graphs. In their work they showed the following container lemma, which roughly says that if all the ρn -induced subgraphs in a graph G are sufficiently dense then there is a relatively small collection of relatively small containers that cover all of the independent sets in G . Importantly, each container is of size strictly less than ρn (which is smaller than n in general), and the number of containers is at most $\binom{n}{2/\epsilon}$ (much

smaller than the trivial 2^n upper bound).⁴

Lemma 1.2 ([KW82]). *For any ϵ, ρ , let $G = (V, E)$ be a graph on n vertices such that every induced subgraph on ρn vertices has at least ϵn^2 edges. Then there exists a set $\mathcal{C} \subseteq 2^V$ of containers that satisfies.*⁵

1. $|\mathcal{C}| \leq \binom{n}{2/\epsilon}$,
2. for every $C \in \mathcal{C}$, $|C| \leq \left(\rho - \frac{\epsilon}{2\rho}\right)n$, and
3. for every independent set I in G , there exists $C \in \mathcal{C}$ such that $I \subseteq C$.

There are many combinatorial applications of graph and hypergraph container lemmas like the one stated above (see [Section 2.2](#) for some examples and discussion). This thesis focuses on the application to analyzing property testing algorithms.

How Containers can Apply to Property Testing. Suppose we want to test the property of having a ρn independent set, which we denote as ρ -INDEPSET. In other words, we want to distinguish between the case that the input graph G has a ρn independent set and the case that every ρn independent set in G has at least ϵn^2 edges. We will now describe how to use a container lemma to analyze a testing algorithm for this problem.

A natural idea for a testing algorithm is to take a random sample $S \subseteq V$ of vertices and check if $G[S]$ has an independent set of size $\rho|S|$. If the input graph has a ρn independent set then it's not too hard to show that $G[S]$ includes $\rho|S|$ vertices from the independent set with probability at least $1/2$ (as shown pictorially in [Fig. 1.3](#)). The challenging part is showing that if G is ϵ -far from ρ -INDEPSET then the probability that the sample includes an independent set of size $\rho|S|$ is very small (i.e. much smaller than $1/2$).

First consider the following naive approach to handling this challenge. Consider all the maximal⁶ independent sets in G . In order for $G[S]$ to have a $\rho|S|$ independent set, S must contain at least $\rho|S|$ vertices from one of the maximal independent sets. It's not too hard to show that, if G is ϵ -far from the ρ -independent set property, every independent set in G is smaller than $(\rho - \frac{\epsilon}{2\rho})n$. Hence, for every independent set I in G , the expected

⁴Note that Kleitman and Winston did not phrase it in this way as the notion of *containers* wasn't defined yet.

⁵We use 2^V to represent the power set of V .

⁶A *maximal* independent set is an independent set for which no additional vertex can be added while preserving independence. Every independent set is a subset of some maximal independent set.

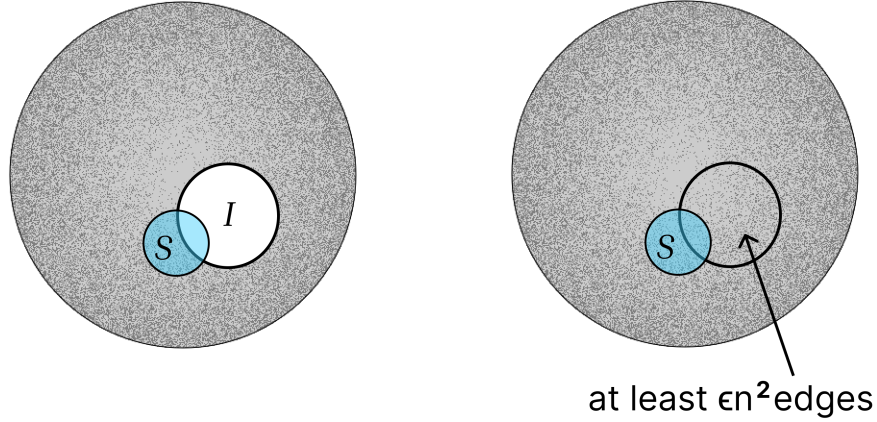


Figure 1.3: Testing the ρ -INDEPSET property. In the yes case (left), the sample S includes a sufficient number of vertices from the large independent set with reasonable probability. The challenging part is showing that in the no case (right), the sample does not include a $\rho|S|$ independent set.

number of vertices S includes from I is at most $(\rho - \frac{\epsilon}{2\rho})|S|$. So, we can upper bound the probability that S contains $\rho|S|$ vertices from a single maximal independent set using a Chernoff/Hoeffding bound, and then we can take a union bound over all possible $\binom{n}{\leq (\rho - \frac{\epsilon}{2\rho})n}$ maximal independent sets.

The main problem with this approach is the union bound: there are simply *too many* independent sets to take a union bound over. The container method can be used here to shortcut the union bound. In particular, [Lemma 1.2](#) says that, while there may be many independent sets, the independent sets can all be covered by a small number of containers. Hence, in order for $G[S]$ to have a $\rho|S|$ independent set, S must contain at least $\rho|S|$ vertices from one of the *containers*. So instead of taking a union bound over all maximal independent sets, we can take a union bound over the $\binom{n}{2/\epsilon}$ containers.

In more detail, using [Lemma 1.2](#) we can already prove a sample complexity upper bound of $O(\text{poly}(1/\epsilon) \cdot \log n)$ for testing the ρ -INDEPSET property. If G is ϵ -far from ρ -INDEPSET, [Lemma 1.2](#) gives a collection $\mathcal{C}$ of at most $\binom{n}{2/\epsilon}$ containers such that each container is of size at most $(\rho - \frac{\epsilon}{2\rho})n$ and every independent set is a subset of some container. Since every independent set is a subset of a container, then the probability that S has a $\rho|S|$ independent set is at most

$$\sum_{C \in \mathcal{C}} \Pr[S \text{ contains at least } \rho|S| \text{ vertices from } C].$$

For every $C \in \mathcal{C}$, the size of C is upper bounded by $|C| \leq (\rho - \frac{\epsilon}{2\rho})n$, and so the expected number of vertices in $S \cap C$ is at most $(\rho - \frac{\epsilon}{2\rho})|S|$. Using a Hoeffding/Chernoff bound, the probability that $S \cap C$ is at least $\rho|S|$ is less than $e^{-2\epsilon^2|S|}$. Hence, we find that

$$\sum_{C \in \mathcal{C}} \Pr[S \text{ contains at least } \rho|S| \text{ vertices from } C] \leq \sum_{C \in \mathcal{C}} e^{-2\epsilon^2|S|} \leq \binom{n}{2/\epsilon} e^{-2\epsilon^2|S|},$$

where the second inequality uses that $|\mathcal{C}| \leq \binom{n}{2/\epsilon}$ from the container lemma. Then, using the upper bound $\binom{n}{2/\epsilon} \leq n^{2/\epsilon}$, we find that if $|S|$ is at least roughly $\frac{\log(n)\log(1/\epsilon)}{\epsilon^3}$ then the probability is smaller than $1/3$.

It is worth noting that the bound above has a $\log n$ dependence. However, we often desire sample complexity bounds that do not depend on n at all. With a more careful analysis of the container lemma by Kleitman and Winston ([Lemma 1.2](#)), it is possible to remove the $\log n$ dependence from the bound above. See [Chapter 3](#) for the details, as well as the proof of the container lemma.

The above demonstrates that container lemmas provide a natural framework for studying canonical property testing algorithms. However, the standard container lemmas are not strong enough to obtain optimal sample complexity bounds. The central theme of this thesis is the development of new graph and hypergraph container lemmas tailored to property testing. Each of the main results introduces a new container lemma for a different property testing problem, each of which can then be used as a shortcut to the standard union bound to get stronger upper bounds on the sample complexity. To the best of our knowledge, our results are the first use of the container method in the analysis of graph property testing algorithms.

How to Prove Container Lemmas. Before stating our results we briefly describe the general approach to proving graph container lemmas. This discussion is intended to provide intuition for the proof techniques that recur throughout the thesis.

The approach to constructing containers can be motivated by a simple encoding game. Suppose two parties Alice and Bob can see a graph G , and Alice selects an independent set I in the graph and wishes to tell Bob her independent set. See [Fig. 1.4](#) for an example. Unfortunately, Alice doesn't have time to tell Bob all of her vertices, instead she just wants to tell Bob one of her vertices and hopes to tell Bob as much information about her independent set as possible. What can Alice do?

One natural idea is for Alice to tell Bob the vertex v in her independent set with highest degree in the graph. Why is this useful? By learning this vertex, Bob can immediately

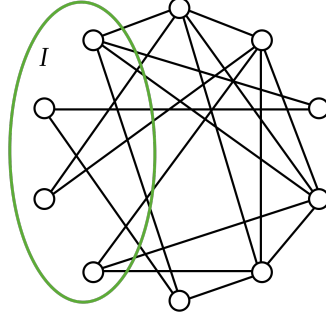


Figure 1.4: Alice is thinking about the independent set I . How can she tell Bob about her independent set by only sharing one vertex?

rule out all neighbours of v because they certainly cannot be in the independent set. In the example (see left hand side of Fig. 1.5), Alice tells Bob the yellow vertex, and Bob can immediately rule out 4 vertices and determine that Alice's independent set must be a subset of the remaining 6 vertices.

Further, there is more Bob learns from this vertex. Since Alice sent her *highest* degree vertex v , any vertex in the graph with higher degree than v also cannot be part of the independent set, because otherwise Alice would have sent it instead! Hence, Bob can remove two more vertices (circled in red in the right hand side of Fig. 1.5), and is able to determine Alice's independent set exactly.

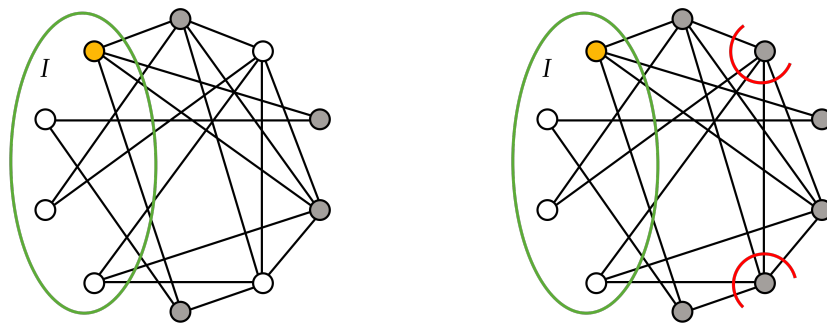


Figure 1.5: If Alice sends her highest degree vertex v (highlighted in yellow), Bob can remove the neighbours (left) and higher degree vertices (right) and determine what Alice's independent set is.

In larger graphs this process can be repeated many times. In each step, Alice selects the vertex of her independent set with highest degree in the current candidate set of vertices, sends this vertex to Bob and Bob can remove neighbours and higher degree vertices to construct a smaller candidate set of vertices. In the case of proving [Lemma 1.2](#), the process is repeated $2/\epsilon$ times (in general Bob cannot determine the independent set exactly, he only constructs a candidate set of vertices). The vertices that are selected as having highest degree in each step form the *fingerprint* of the independent set. In some sense, the fingerprint vertices can be viewed as the most *informative* vertices in the independent set. Using just the fingerprint, it is possible to construct a candidate set of vertices, called a *container*, by repeatedly removing the neighbours and higher degree vertices of each fingerprint vertex. See [Fig. 1.6](#) for an example of what this may look like after a number of steps.

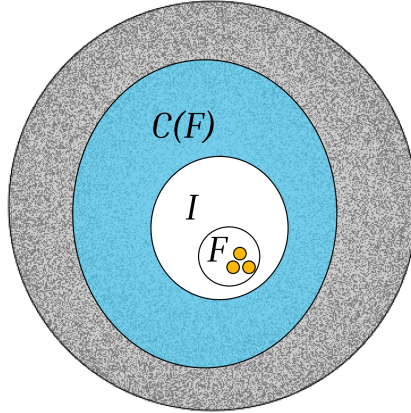


Figure 1.6: Given an independent set I , multiple steps of the process constructs a fingerprint F (the set of vertices colored in yellow) which can be used to construct a container $C(F)$ (colored in blue) by repeatedly removing the neighbours and higher degree vertices of the fingerprint vertices.

Next, we describe how to construct a *collection* of containers as described in [Lemma 1.2](#). For each independent set I in the graph, we run the iterative process described above $2/\epsilon$ times and add the resulting container to the collection. The key observation is that, while there are potentially an exponential number of independent sets, there are at most roughly $\binom{n}{2/\epsilon}$ possible fingerprints. This is because each fingerprint is a set of $2/\epsilon$ vertices from the graph. Since each fingerprint defines a container, there cannot be more than $\binom{n}{2/\epsilon}$ containers.

We note that this collection of containers can be constructed for any graph, not just graphs that are ϵ -far from ρ -INDEPSET. However, in order to prove the upper bound on the sizes of the containers (in [Lemma 1.2](#) each container is of size at most $(\rho - \frac{\epsilon}{2\rho})n$), we need that the graph is ϵ -far from ρ -INDEPSET. For further details on proving container lemmas, and why we run the procedure for $2/\epsilon$ steps, see the later chapters of this thesis ([Chapter 3](#) for example).

1.3 Testing the Independent Set Property

As was shown above, a standard graph container lemma immediately gives an upper bound on the sample complexity of ϵ -testing ρ -INDEPSET. In fact, with a more careful analysis of the container lemma from Kleitman and Winston, it is possible to achieve a $\tilde{O}(\rho^4/\epsilon^3)$ upper bound, reproducing the best known bound on ϵ -testing the ρ -INDEPSET property achieved by Feige, Langberg and Schechtman [[FLS04](#)] (see [Chapter 3](#) for the details on how to reproduce this bound using the container method). Can the container method be used to achieve the optimal bounds on testing ρ -INDEPSET? In particular, Feige, Langberg and Schechtman also gave a lower bound of $\mathcal{S}_{\rho\text{-INDEPSET}}(n, \epsilon) = \tilde{\Omega}(\rho^3/\epsilon^2)$, can the container method be used to prove a stronger upper bound to match this lower bound?

In our first result we show that the container method *can* be used to achieve the optimal upper bound, up to logarithmic factors, on the sample complexity of testing the ρ -INDEPSET property.

Theorem 1.3. *The sample complexity of ϵ -testing the ρ -INDEPSET property is $\tilde{O}(\frac{\rho^3}{\epsilon^2})$.*

We establish [Theorem 1.3](#) via a new container lemma that builds on [Lemma 1.2](#). Recall that [Lemma 1.2](#) says that, in a graph that is ϵ -far from ρ -INDEPSET, the independent sets can be covered by a collection of containers where there are at most roughly $\binom{n}{1/\epsilon}$ containers and each container is of size at most roughly $(\rho - \frac{\epsilon}{\rho})n$. We cannot improve these worst case bounds on the size of the collection or the size of the containers themselves, however, we give a more refined trade off in the number of containers of each particular size. For example, we show that there can only be at most $\binom{n}{O(1)}$ containers of size $(\rho - \tilde{\Theta}(\frac{\epsilon}{\rho}))n$. Further, if there are actually $\binom{n}{1/\epsilon}$ containers, then *most* of them should be of size $(\rho - \tilde{\Theta}(\rho))n$.

In order to formalize these trade-offs we need to rephrase the container lemma statement in terms of fingerprints. In particular, instead of having a collection of containers, our container lemma gives a collection of fingerprints $\mathcal{F} \subseteq 2^V$ and a container function $\mathcal{C}$:

$\mathcal{F} \rightarrow 2^V$ which can be applied to each fingerprint to construct the containers. Then, our main change is showing that for each fingerprint $F \subseteq V$, there is a trade off between the size of the fingerprint and the size of the associated container $\mathcal{C}(F)$. In other words, the larger the container, the smaller the fingerprint. Hence, there can't be too many large containers because there can't be too many small fingerprints.

Lemma 1.4. *Let $G = (V, E)$ be a graph on n vertices that is ϵ -far from the ρ -INDEPSET property. Then there exists a collection $\mathcal{F} \subseteq 2^V$ of fingerprints and a container function $\mathcal{C} : \mathcal{F} \rightarrow 2^V$ that satisfy the following two properties.*

1. *For every independent set I in G of cardinality $|I| \geq \frac{8\rho^2 \ln(\frac{2\rho}{\epsilon})}{\epsilon}$, there is a fingerprint $F \in \mathcal{F}$ such that $F \subseteq I \subseteq \mathcal{C}(F)$.*
2. *For every fingerprint $F \in \mathcal{F}$, the cardinality of F and of its associated container $\mathcal{C}(F)$ are bounded by*

$$|F| \leq \frac{8\rho^2 \ln(\frac{2\rho}{\epsilon})}{\epsilon} \quad \text{and} \quad |\mathcal{C}(F)| \leq \left(\rho - |F| \cdot \frac{\epsilon}{8\rho \ln(\frac{2\rho}{\epsilon})} \right) n.$$

Using our new container lemma, we prove a stronger upper bound on the sample complexity of testing the property of having a ρn independent set ([Theorem 1.3](#)). We do this by handling containers of each size separately in the union bound. The large containers have small fingerprints and so there are less fingerprints to union over, and small containers result in stronger concentration bounds when considering the probability that a sufficient number of vertices in the sample come from the container.

1.4 Testing Satisfiability

[Theorem 1.3](#) demonstrates that the container method framework applies very naturally to analyzing testers for the ρ -INDEPSET property. Is it possible to analyze testers for other properties using this framework? The property of being bipartite (and more generally the property of being k -colorable) is really the property of being able to partition the vertices into two (k in the case of k -colorability) independent sets. Hence, as the container method is well suited to independent sets, this seems like a natural next step. More generally, bipartiteness and k -colorability are both special cases of the satisfiability property for constraint satisfaction problems. Can we use the container method framework to get strong upper bounds on the sample complexity of testing these properties?

In our next result we establish a new hypergraph container lemma and show that it can be used to obtain new upper bounds on the sample complexity of satisfiability testing, which also gives new upper bounds on the sample complexity of testing k -colorability and recovers the best known bounds for bipartiteness testing.

In the satisfiability testing problem, which we denote by (q, k) -SAT, the input is a constraint satisfaction problem ϕ , which is a set of n variables that take values from an alphabet of size k , and a set of constraints, where each constraint is a predicate on a distinct set of q variables. The task of a tester is to distinguish between the case that ϕ is satisfiable and the case that at least $\epsilon \binom{n}{q}$ constraints must be removed from ϕ to make it satisfiable. The canonical tester samples a set S of variables and accepts if the restriction of ϕ to the set S of variables, denoted $\phi[S]$, is satisfiable. This tester has *one-sided error*, in that it always accepts when ϕ is satisfiable. We show that the canonical tester for satisfiability is efficient in a strong sense.

Theorem 1.5. *The sample complexity of ϵ -testing the (q, k) -SAT property is $\tilde{O}(\frac{kq^3}{\epsilon})$.*

[Theorem 1.5](#) improves on the best prior bound of roughly k^{3q}/ϵ by Sohler [[Soh12](#)], and is the first result to show that the sample complexity of testing (q, k) -SAT is polynomial in all three of q , k , and $1/\epsilon$. The best lower bound on testing satisfiability is $\Omega(\log k/\epsilon)$ coming from a lower bound on testing k -colorability [[KS26](#)].

Application to Testing Colorability. Sample complexity bounds for testing satisfiability can be used to obtain bounds on the sample complexity for testing k -colorability.

Corollary 1.6. *The sample complexity of ϵ -testing the k -COLOR property is $\tilde{O}(\frac{k}{\epsilon})$.*

[Corollary 1.6](#) improves on the prior incomparable bounds of $\tilde{O}(k^2/\epsilon^2)$ by Alon and Krivelevich and $\tilde{O}(k^6/\epsilon)$ by Sohler [[Soh12](#)]. In the case of bipartiteness ($k = 2$), [Corollary 1.6](#) recovers the best bound of $\tilde{O}(1/\epsilon)$ by Alon and Krivelevich [[AK02](#)]. For general k , there remains a gap between [Corollary 1.6](#) and the best lower bound of $\Omega(\log k/\epsilon)$ [[KS26](#)].

[Theorem 1.5](#) gives bounds on the sample complexity of testing k -colorability of hypergraphs as well [[AS03](#), [Soh12](#)]. In this problem, denoted (q, k) -COLOR, the input is a q -uniform hypergraph H on n vertices and the goal of the tester is to distinguish between the case that H is k -colorable and the case that at least $\epsilon \binom{n}{q}$ edges need to be removed to make H k -colorable. The sample complexity bound for testing satisfiability implies a corresponding bound for the sample complexity of the natural canonical tester for k -colorability, which improves on the prior best bounds of $(qk)^{O(q)}/\epsilon$ by Sohler [[Soh12](#)] and $q^{O(q)}k^2/\epsilon^2$ by Czumaj and Sohler [[CS01](#)].

Corollary 1.7. *The sample complexity of ϵ -testing the (q, k) -COLOR property is $\tilde{O}(\frac{kq^3}{\epsilon})$.*

Application to Testing Partition Properties. [Theorem 1.5](#) also implies new sample complexity bounds for testing all *semi-homogeneous partition properties*, a class of graph properties first studied by Nakar and Ron [[NR18](#)] (where they are called *0–1 graph partition properties*) and by Fiat and Ron [[FR21](#)]. This is the class of graph partition properties where between two parts the edge density is either 0, 1 or has no restrictions. (See [Chapter 2](#) for the formal definitions.) This is a natural class of graph properties, which includes k -colorability, being a biclique, and many other properties as well. This class is also interesting in its own right as it has connections to efficient distance approximation [[FR21](#)] and it is exactly the class of graph partition properties that can be tested with $\text{poly}(1/\epsilon)$ queries with one-sided error [[NR18](#)]. Let k -SHPP denote the set of semi-homogeneous partition properties with exactly k parts. [Theorem 1.5](#) implies the following bound on the sample complexity of the natural canonical tester for semi-homogeneous partition properties.

Corollary 1.8. *The sample complexity of ϵ -testing any property in k -SHPP is $\tilde{O}(\frac{k}{\epsilon})$.*

The bound in [Corollary 1.8](#) again recovers the best-known bound on the sample complexity for testing k -colorability of graphs. In general, it improves on both of the incomparable previous bounds of roughly $k \log k / \epsilon^2$ [[NR18](#)] and k^6 / ϵ [[Soh12](#)] on the sample complexity for testing k -SHPP properties.

A New Hypergraph Container Lemma. To prove [Theorem 1.5](#) we again want to prove that the natural canonical tester has high success probability even when it has small sample complexity. In the case of satisfiability, however, what is sampled by the canonical tester is a set of variables of the input CSP, and it accepts if and only if the CSP restricted to the sampled variables is satisfiable.

We wish to apply the container method framework to analyze the canonical tester for satisfiability. In this case, instead of a container lemma for independent sets in a graph that is ϵ -far from having a large independent set (like in [Lemma 1.4](#)), we desire a container lemma for the sets of variables S for which $\phi[S]$ is satisfiable in a CSP ϕ that is ϵ -far from satisfiable. Such a container lemma is useful because, when ϕ is ϵ -far from satisfiable, this new collection of containers can be used to upper bound the probability that the tester wrongly accepts. We prove such a container lemma.

Our new container lemma follows a very similar structure to [Lemma 1.4](#), with “independent sets” replaced by sets of variables S for which the restriction of the CSP $\phi[S]$ is

satisfiable. Omitting some of the technical details, our container lemma says the following (for the full version of the lemma see [Chapter 5](#)).

Lemma 1.9 (Informal). *Let ϕ be a CSP over variables $X = \{x_1, \dots, x_n\}$ that is ϵ -far from (q, k) -SAT. Then there exists a collection $\mathcal{F}$ of fingerprints (where each fingerprint is a subset of variables) and a container function $\mathcal{C} : \mathcal{F} \rightarrow 2^X$ that satisfy the following two properties.*

1. *For every $S \subseteq X$ for which $\phi[S]$ is satisfiable, there is a fingerprint $F \in \mathcal{F}$ such that $F \subseteq S \subseteq \mathcal{C}(F)$.*
2. *For every fingerprint $F \in \mathcal{F}$, the cardinality of F and of its associated container $\mathcal{C}(F)$ are bounded by*

$$|F| \leq 8 \frac{kq^2}{\epsilon} \quad \text{and} \quad |\mathcal{C}(F)| \leq \left(1 - |F| \cdot \frac{\epsilon}{4kq^3 \ln \frac{kq}{\epsilon}} \right)^n.$$

The key idea to proving the container lemma is to convert the CSP problem into a problem about independent sets in a hypergraph, and then apply the container method to the hypergraph. In particular, for any CSP ϕ , we use a standard transformation to construct a hypergraph H_ϕ such that satisfying assignments to the variables of ϕ correspond to independent sets in H_ϕ (for full details of the transformation see [Chapter 5](#)). Then, roughly speaking, a collection of containers for the independent sets in H_ϕ gives a set of containers for sets of variables S for which $\phi[S]$ is satisfiable.

Standard hypergraph container lemmas can be applied on the hypergraph H_ϕ , however, in order to prove the strong bounds in our container lemma, we utilize the fact that H_ϕ is not an arbitrary hypergraph, and rather H_ϕ has specific characteristics because of how it is constructed from ϕ .

1.5 Tolerant Testing the Independent Set Property

Our previous two theorems demonstrate that the container method framework can be used to give strong upper bounds on the sample complexity of testing a number of graph and hypergraph properties. For all of those properties, analyzing the canonical tester corresponds to analyzing the structure of independent sets in (hyper)graphs, and hence they are natural settings to apply the container method. In our next result we demonstrate that

the container method framework can also be used in a testing setting where the problem is about sparse subgraphs instead of independent sets. In particular, we will study the problem of *tolerant* testing ρ -INDEPSET. In this setting, while the property is about independent sets, analyzing the canonical tester corresponds to analyzing the structure of *sparse* induced subgraphs in graphs that are ϵ -far from ρ -INDEPSET.

Tolerant testing, introduced by Parnas, Ron and Rubinfeld [PRR06], is a natural generalization of testing where the goal is to distinguish between graphs that are ϵ_1 -close to some property Π and graphs that are ϵ_2 -far from Π , where ϵ_1 -close means that at most $\epsilon_1 n^2$ edges need to be added or removed from the graph to make it have the property. Such a tester is called an (ϵ_1, ϵ_2) -tolerant tester, and the goal is to find the minimum sample complexity of any such tester, as a function of ϵ_1, ϵ_2 and the input size. It is desirable for such a tester to apply for any $\epsilon_1 < \epsilon_2$, referred to as a fully tolerant tester, but this is not necessary. For example, if $\epsilon_1 = 0$ then this corresponds to the standard notion of testing. In general, ϵ_1 may be a function of ϵ_2 .

We now wish to extend the container method to the tolerant setting using the same approach as in the non-tolerant setting. Recall that for the non-tolerant version of testing ρ -INDEPSET, the natural tester takes a random sample $S \subseteq V$ of vertices from the input graph $G = (V, E)$ and checks if $G[S]$ has a $\rho|S|$ independent set. For tolerant testing, the natural tester instead checks if $G[S]$ has a set of $\rho|S|$ vertices that induces only a small number of edges. The main challenge with analyzing the tolerant tester is showing that if G is ϵ -far from ρ -INDEPSET then the probability that $G[S]$ includes a sparse subgraph of size $\rho|S|$ is small. We therefore desire a container lemma for sparse subgraphs instead of independent sets, however, the standard graph container approach does not naturally extend to sparse subgraphs.

To overcome this challenge, we introduce a new container lemma for sparse subgraphs, which leads to our new tolerant testing result. We first state our theorem and an application, and then discuss the new container lemma at the end of this section.

Theorem 1.10. *There exists a constant c such that for any $\epsilon > 0$ the sample complexity of $\left(\frac{\epsilon}{c \log^4(1/\epsilon)}, \epsilon\right)$ -tolerant testing the ρ -INDEPSET property is $\tilde{O}(\rho^3/\epsilon^2)$.*

Prior to Theorem 1.10, there were no direct results analyzing the tolerant tester for ρ -INDEPSET. Shown in [PRR06], any standard tester that takes a random sample S of vertices and inspects only $G[S]$ has a very small amount of tolerance by default based on the fact that a random subgraph won't find any violating edges in a graph that is very

close to the property,⁷ and so the tester of [Theorem 1.3](#) gives a (ϵ^4, ϵ) -tolerant tester with sample complexity $\tilde{O}(\rho^3/\epsilon^2)$. However, this is described by [\[PRR06\]](#) as “weak” tolerance, and the real question is: when does there exist an efficient tolerant tester beyond this weak amount of tolerance given by the standard tester?

The best prior results for tolerant testing ρ -INDEPSET comes from a general result for tolerant testing of general graph partition properties, which includes the ρ -INDEPSET property. Fiat and Ron show that for any $\epsilon_1 < \epsilon_2$ there is a (ϵ_1, ϵ_2) -tolerant tester for any graph partition property with sample complexity $\text{poly}(1/(\epsilon_2 - \epsilon_1))$ [\[FR21, Appendix C\]](#), however the polynomial dependence is much larger than [Theorem 1.10](#).

The sample complexity in [Theorem 1.10](#) matches the sample complexity, up to logarithmic factors, of the non-tolerant tester for ρ -INDEPSET of [Theorem 1.3](#), which itself is known to be optimal, up to logarithmic factors, based on a lower bound on the non-tolerant version by Feige, Langberg and Schechtman [\[FLS04\]](#). In other words, [Theorem 1.10](#) shows that, surprisingly, $(\tilde{\Omega}(\epsilon), \epsilon)$ -tolerant testing ρ -INDEPSET is no harder than the non-tolerant version.

In other settings, there are properties that demonstrate that tolerant testing is, in general, much harder than non-tolerant testing. In particular, in the bounded degree model, Goldreich and Wigderson gave a graph property for which there is a subexponential gap between the query complexity of non-tolerant testing and the query complexity of tolerant testing (i.e. the query complexity in the non-tolerant setting is q and the query complexity in the tolerant setting is $\exp(q^c)$ for some constant c) [\[GW22\]](#). Fischer and Fortnow show a similar separation for testing a property of boolean functions [\[FF05\]](#). It is not known whether such a strong separation exists in the dense graph setting.

Application to Non-Canonical Tester. All of our results stated thus far pertain to the sample complexity of canonical testers. Our tolerant tester of [Theorem 1.10](#) also has applications to analyzing the query complexity of non-canonical testers (i.e. testers that are able to query arbitrary edges in the adjacency matrix, see [Chapter 2](#) for definitions).

Corollary 1.11. *There exists a constant c such that for any $\epsilon > 0$ the query complexity of $(\frac{\epsilon}{c \log^4(1/\epsilon)}, \epsilon)$ -tolerant testing the ρ -INDEPSET property is $\tilde{O}(\rho^4/\epsilon^3)$.*

The canonical testers of [Theorem 1.10](#) (and [Theorem 1.3](#) for the non-tolerant setting) make $\tilde{O}(\rho^6/\epsilon^4)$ edge queries to the adjacency matrix. By definition, the problem of ϵ -testing

⁷They actually show this for testers which make edge queries to the input which are uniformly distributed, but a similar argument applies for testers where the vertex samples are uniformly distributed.

the ρ -INDEPSET property is non-trivial only when $\epsilon < \rho^2$, as otherwise there are no graphs that are ϵ -far from having a ρ -independent set. When $\epsilon = \Theta(\rho^2)$, the query complexity of both the canonical tester and the non-adaptive tester in the proof of [Corollary 1.11](#) is $O(1/\epsilon)$, which is optimal. And in all other non-trivial regimes where $\epsilon = o(\rho^2)$, the bound in [Corollary 1.11](#) is strictly better than the query complexity bound for the canonical tester.

It has been known for some time that the gap between the query complexity of the canonical tester and the optimal tester for any graph property is at most quadratic [[GT03](#)], and this holds even for adaptive testers, which are testers that can determine which edge to query next based on what the previous edge queries have revealed about the graph G . But in general there are few graph properties for which we know whether the canonical tester has optimal query complexity or not. [Corollary 1.11](#) show that the canonical tester for the ρ -INDEPSET property does not have optimal query complexity. Further, since our tester is non-adaptive, [Corollary 1.11](#) also shows that the ρ -INDEPSET property is one of the few graph properties for which we know that the canonical tester does not have the same query complexity as the best non-adaptive tester.

New Container Lemma for Sparse Subgraphs. In order to prove [Theorem 1.10](#) we wish to use the container method to analyze a tolerant tester for ρ -INDEPSET. However, as discussed prior to stating [Theorem 1.10](#), we now require a container lemma for sparse subgraphs⁸ instead of independent sets. The main challenge with proving such a lemma has to do with the standard way that the collection of fingerprints is constructed. In particular, as introduced in [Section 1.2](#), a fingerprint for an independent set I in $G = (V, E)$, and the associated container that contains I , is constructed through a greedy procedure as follows. After initiating the container to V , in each step the vertex $v \in I$ with the highest degree in the current container is added to the fingerprint, and the neighbours of v and the vertices with higher degree than v are removed from the current container. In this way, the independent set is always a subset of the current container. However, this fails dramatically when dealing with sparse subgraphs: running this procedure on a set J for which $G[J]$ is a sparse subgraph would not guarantee that J remains in the current container in each step because there may be vertices in J that neighbour the selected vertex. Even if we relax the condition so that we only require that *most* of J is contained in the container, which is actually all that we need to prove [Theorem 1.10](#), the fingerprint procedure still fails because all of J may be removed by a single vertex selected to the fingerprint that neighbours all vertices in J .

To illustrate this challenge, consider the example in [Fig. 1.7](#). On the left we have the

⁸Here and throughout we say “subgraph” to mean induced subgraph.

independent set version, selecting the highest degree vertex and removing neighbours and higher degree vertices does not remove any vertices from the independent set. However, on the right, this process fails; selecting the highest degree vertex in J and removing neighbours removes too much of J .

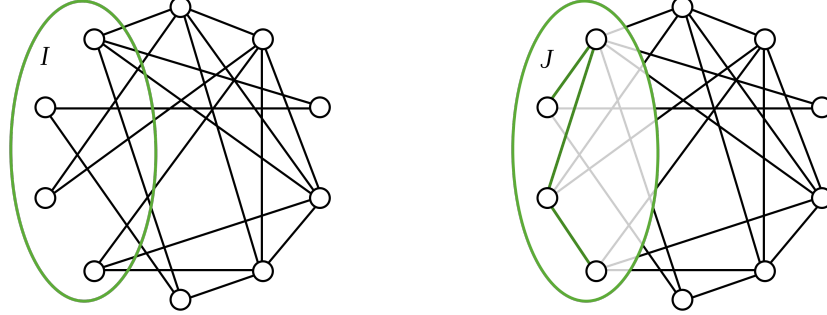


Figure 1.7: Encoding game for independent sets (left) vs sparse subgraphs (right). In the sparse subgraph setting, selecting the highest degree vertex in J and removing neighbours removes too much of J from the container.

Due to this challenge, nearly all of the prior work on the graph and hypergraph container method applies to independent sets and not sparse subgraphs. Container lemmas for sparse subgraphs have only been studied in the work of Saxton and Thomason [ST15], as well as a very recent work by Nenadov [Nen25]. In [ST15] the results apply to sparse subgraphs $G[J]$ but there is a trade-off between the allowable density of $G[J]$ and the size of fingerprints. The parameters in their general container theorem are not sufficient for our tolerant testing application, where very small fingerprints are required. In [Nen25] the results apply to subgraphs $G[J]$ for which the maximum degree is much less than $\frac{\epsilon}{\rho^2}|J|$ (in graphs that are ϵ -far from ρ -INDEPSET). For our application we require a container lemma for a much larger range of sparse subgraphs. In particular, we require a lemma that applies for sparse subgraphs with edge density up to roughly $\frac{\epsilon}{\rho^2}$, which is the minimum edge density of the entire graph provided it is ϵ -far from ρ -INDEPSET, and for sparse subgraphs J that may have maximum degree up to $|J|$. Neither of these prior results gives the parameters required for our setting, and it is not even clear whether such a container lemma should exist.

We address these challenges and give a new container lemma for sparse subgraphs in graphs that are ϵ -far from ρ -INDEPSET. Informally, our new container lemma says that if G is ϵ -far from having a ρn independent set then there exists a set of fingerprints $\mathcal{F} \subseteq 2^V$ and a function $C : \mathcal{F} \rightarrow 2^V$ such that for every $J \subseteq V$ for which $G[J]$ has fewer than

$\tilde{O}(\frac{\epsilon}{\rho^2}|J|^2)$ edges, there exists a fingerprint $F \in \mathcal{F}$ such that $C(F)$ contains *most* of J . In addition, we have similar trade offs on the fingerprint and container sizes as was seen in our previous container lemmas. For the full formal statement of the container lemma see [Chapter 6](#).

1.6 Structure of this Thesis

[Chapter 2](#) covers the basic definitions of property testing, relevant prior work, and some useful bounds that will be used throughout the thesis. [Chapter 3](#) gives a warm-up application of the container method to testing the independent set property which achieves weaker bounds than [Theorem 1.3](#) but has a much simpler analysis. In particular, we use a simple container procedure to prove a graph container lemma, and use the graph container lemma to analyze the canonical tester. Each of the following three chapters follows a similar structure, with modified container procedures and stronger container lemmas. [Chapter 4](#) proves our new container lemma for independent sets that in graphs that are far from having a ρn independent set, and uses it to upper bound on the sample complexity of testing the ρ -INDEPSET property ([Theorem 1.3](#)). [Chapter 5](#) proves our new hypergraph container lemma for the hypergraph corresponding to the constraint satisfaction problem, and uses it to upper bound on the sample complexity of testing the (q, k) -SAT property ([Theorem 1.5](#)). [Chapter 6](#) proves our new container lemma for sparse subgraphs in graphs that are far from having a ρn independent set, and uses it to upper bound on the sample complexity of tolerant testing the ρ -INDEPSET property ([Theorem 1.10](#)).

[Chapter 7](#) gives some concluding remarks and discusses a number of future directions of this work.

Chapter 2

Background and Prior Work

This chapter covers the definitions and relevant prior work for this thesis. Throughout this thesis we work on simple undirected graphs and hypergraphs and use standard notation to represent them (see, for example, [Die12]). In particular, we represent a graph by $G = (V, E)$, where V is the vertex set and E is the edge set, and our graphs do not have loops or multiedges. For a subset of the vertices $S \subseteq V$, we use $G[S]$ to denote the induced subgraph, and whenever we say “subgraph” we are referring to induced subgraph. Throughout we use $\deg(v)$ to denote the degree of v in G , and for $S \subseteq V$ we use $\deg_{G[S]}(v)$ to denote the degree of v in $G[S]$. For $S, T \subseteq V$, we use $E(S)$ to denote the edges of G with both endpoints in S , and we use $E(S, T)$ to denote the set of edges in G with one endpoint in S and one endpoint in T .

2.1 Graph Property Testing

Graph property testing was introduced by Goldreich, Goldwasser, and Ron [GGR98] and has since developed into a broad area of research. We refer the reader to the books by Goldreich [Gol17], and Bhattacharyya and Yoshida [BY22], and the survey by Ron [Ron10], for comprehensive introductions to the area. Graph property testing algorithms aim to distinguish between graphs that have a specific property from graphs that are far from the property by inspecting a small fraction of the graph. The definition of *far* depends on the model, and the most commonly studied model is the dense graph model. In this model, a graph G on n vertices is represented by an $n \times n$ adjacency matrix, and we say a graph G on n vertices is ϵ -far from a graph property Π if we need to add or remove at least ϵn^2 edges from G to obtain a graph that has the property Π . In other words, a graph is ϵ -far

from the property if at least an 2ϵ fraction of the entries of the adjacency matrix must be changed to make the graph satisfy the property. A testing algorithm accesses the input graph G via queries to the adjacency matrix and random samples from the vertex set.

A randomized algorithm T is an ϵ -tester for the property Π of graphs if there are absolute constants δ_1, δ_2 satisfying $0 \leq \delta_2 < \delta_1 \leq 1$ such that T accepts all graphs that have property Π with probability at least δ_1 and accepts all graphs that are ϵ -far from Π with probability at most δ_2 . (The definition of testers often fixes $\delta_2 = \frac{1}{3}$ and $\delta_1 = \frac{2}{3}$ but standard success amplification methods show that the two definitions result in complexity measures that are asymptotically equivalent so we use the more general definition for convenience.) If $\delta_1 = 1$ then T is said to have *one-sided error*, and otherwise T has *two-sided error*.

We are interested in property testers that examine only a sublinear fraction of the graph during their execution. The most basic such tester is the *canonical tester* that samples a set $S \subseteq V$ of vertices uniformly at random, queries the adjacency matrix on all pairs of vertices in S , and accepts if and only if the induced subgraph $G[S]$ on the set S of vertices satisfies some fixed property (which may or may not be the same as the property being tested). The *sample complexity* of the canonical tester is the size $|S|$ of the set of sampled vertices.

A more general class of graph property testers is the class of algorithms that query arbitrary pairs of vertices $u, v \in V$ to determine if $(u, v) \in E$ or not. A tester that fixes all of the edge queries it wishes to make to the unknown graph G before obtaining the answer to any of these queries is called *non-adaptive*. An *adaptive* tester can, by contrast, determine which edge to query next based on what the previous edge queries have revealed about the graph G . The *query complexity* of a non-adaptive or adaptive graph property tester is the maximum number of edge queries it may make during its execution.

Property testers give two natural measures of complexity for graph properties:

$\mathcal{S}_\Pi$: The *sample complexity* of Π , denoted $\mathcal{S}_\Pi(n, \epsilon)$, is the minimum sample complexity of a canonical ϵ -tester for Π on graphs over n vertices.

$\mathcal{Q}_\Pi$: The *query complexity* of Π , denoted $\mathcal{Q}_\Pi(n, \epsilon)$, is the minimum query complexity of an ϵ -tester for Π on graphs over n vertices.

For any graph property Π , a canonical tester with sample complexity s makes $\binom{s}{2}$ queries, and so $\mathcal{Q}_\Pi(n, \epsilon) \leq \mathcal{S}_\Pi(n, \epsilon)^2$. Further, Goldreich and Trevisan showed that any tester (in the dense graph model), even an adaptive one, can be converted into a canonical

tester with at most a quadratic overhead in the number of queries made to the adjacency matrix [GT03], which can be rephrased as

$$\mathcal{Q}_{\Pi}(n, \epsilon) = \Omega(\mathcal{S}_{\Pi}(n, \epsilon)).$$

Due to this relationship between sample and query complexity, the majority of the work in graph property testing has focused on canonical testers and their associated sample complexities. As such, canonical testers will be the main focus of this thesis.

Much of the work done in graph property testing fits under two umbrellas. The first is characterizing graph properties that are testable with sample complexity that only depends on ϵ . Having a sample complexity which only depends on ϵ is desirable because, if ϵ is a constant, even as the number of vertices in the graph grows the algorithm still only takes a constant sized sample S of vertices and runs a constant-time algorithm on the induced subgraph $G[S]$. Much of the work in this direction relies heavily on the regularity lemma. This line of research culminated in qualitative characterizations of the graph properties that are testable with sample complexity depending only on ϵ , both with one-sided and two-sided error [AFNS09, AS08]. The resulting characterizations show exactly which graph properties are testable in this sense, however, due to the use of the regularity lemma, the dependence on $1/\epsilon$ in the sample complexity can be tower type.

The second main direction of work in graph property testing is finding the optimal sample complexity, as a function of ϵ , for various natural graph properties. This line of work seeks tight bounds on the sample complexity of testing specific natural properties such as graph partition properties, H -freeness, and spectral properties (see the textbooks [BY22, Gol17] for more on the various properties that have been studied).

The results in this thesis fits under the second umbrella. The remainder of this section reviews previous work on the specific problems studied in this thesis.

Testing the Independent Set Property. The ρ -INDEPSET property is the set of all graphs on n vertices that contain a independent set on ρn vertices.

What is the sample complexity $\mathcal{S}_{\rho\text{-INDEPSET}}(n, \epsilon)$ for ϵ -testing the ρ -INDEPSET property? Goldreich, Goldwasser, and Ron [GGR98] were the first to consider this question and showed that $\mathcal{S}_{\rho\text{-INDEPSET}}(n, \epsilon) = \tilde{O}(\rho/\epsilon^4)$. Improved bounds on the sample complexity for testing the ρ -INDEPSET property were obtained by Feige, Langberg, and Schechtman [FLS04]. With a new direct analysis of the canonical tester for testing independent sets, they showed that $\mathcal{S}_{\rho\text{-INDEPSET}}(n, \epsilon) = \tilde{O}(\rho^4/\epsilon^3)$. And by considering the restricted problem of distinguishing graphs that consist of a single ρn -independent set from those that

consist of a single $(\rho - \frac{\epsilon}{\rho})n$ -independent set, they showed that $\mathcal{S}_{\rho\text{-INDEPSET}}(n, \epsilon) = \tilde{\Omega}(\rho^3/\epsilon^2)$. Note that the problem is non-trivial only when $\epsilon < \rho^2$, since we can always remove at most $\binom{\rho n}{2} < \rho^2 n^2$ edges to any graph to form a ρn -independent set. Hence the $\tilde{\Omega}(\rho^3/\epsilon^2)$ lower bound in [FLS04] is always strictly smaller than their upper bound.

Testing k -Colorability. The k -COLOR property is the set of all graphs that are k -colorable. In particular, a graph is k -colorable if there is a way to color the vertices with k colors such that no two adjacent vertices have the same color. The case where $k = 2$ corresponds to bipartiteness.

The study of the sample complexity of the k -COLOR property has a long history that predates the definition of property testing itself. Rödl and Duke [RD85], building on prior work of Bollobás, Erdős, Simonovits, and Szemerédi [BESS78], showed that $\mathcal{S}_{k\text{-COLOR}}(n, \epsilon)$ is a constant independent of n when both k and ϵ are constant. However, this result relies on the regularity lemma and so the bounds obtained on the sample complexity grow as a tower of height polynomial in $1/\epsilon$.

Goldreich, Goldwasser, and Ron [GGR98] obtained much better bounds for the k -COLOR property. They showed that $\mathcal{S}_{k\text{-COLOR}}(n, \epsilon) = \tilde{O}(k^2/\epsilon^3)$ for all $k \geq 3$. Alon and Krivelevich [AK02] further improved the bounds for testing k -colorability. They showed that $\mathcal{S}_{k\text{-COLOR}}(n, \epsilon) = \tilde{O}(k/\epsilon^2)$ for all $k \geq 3$. Sohler [Soh12] obtained another improvement in the ϵ -dependence showing that $\mathcal{S}_{k\text{-COLOR}}(n, \epsilon) = \tilde{O}(k^6/\epsilon)$ and, in particular, when k is a constant the sample complexity of testing k -colorability is $\tilde{\Theta}(1/\epsilon)$. But the bound of Alon and Krivelevich remains stronger whenever $\epsilon = \omega(1/k^5)$.

Note that for testing bipartiteness, nearly optimal bounds $\mathcal{S}_{2\text{-COLOR}}(n, \epsilon) = \tilde{\Theta}(1/\epsilon)$ are known [GGR98, AK02]. For k -COLOR, the best lower bound is $\Omega(\log k/\epsilon)$ [KS26].

Testing k -Colorability in Hypergraphs. For any $q \geq 2$, a q -uniform hypergraph $H = (V, E)$ is the natural generalization of a graph where elements of E are sets of q distinct elements from V . (When $q = 2$, this definition corresponds exactly to that of a graph.) All the definitions from the last section carry over to properties of hypergraphs, with one important difference: for any fixed property Π of q -uniform hypergraphs, the hypergraph H is ϵ -far from Π if we have to add or remove $\epsilon \binom{n}{q}$ edges from the hypergraph to obtain a hypergraph H' that has property Π . This modification ensures that we have that H is ϵ -far from a property Π if and only if an ϵ fraction of all possible edges in a q -uniform hypergraph must be modified to get a hypergraph with the property.

A q -uniform hypergraph $H = (V, E)$ is k -colorable if there exists a coloring of the vertices using k colors such that no hyperedge is monochromatic. For q -uniform hypergraphs, Alon and Shapira [AS03] showed that the sample complexity of the canonical k -colorability tester is at most $(qk)^{O(q)}/\epsilon^2$. This bound was improved to $(qk)^{O(q)}/\epsilon$ by Sohler [Soh12] and to $q^{O(q)}k^2/\epsilon^2$ by Czumaj and Sohler [CS01].

Remark 2.1. Much of the prior work on testing hypergraphs uses a slightly different definition of distance. Namely, a q -uniform hypergraph is often defined to be ϵ -far from a property if at least ϵn^q edges (instead of $\epsilon \binom{n}{q}$ as above) need to be added or removed to obtain a hypergraph with the property. This is often slightly more convenient for analysis (and is in fact why we use this version of distance for graph testing), however the two notions are asymptotically equivalent only when q is a fixed constant. In the setting that particularly interests us here where we care about the asymptotic dependence on q as well as on other parameters in the property definition, our notion of distance is strictly more restrictive and more appropriate to the study of (simple) hypergraphs where all hyperedges are sets of distinct vertices. In particular, a sample complexity with polynomial dependence on q under our notion of distance also implies a matching polynomial dependence on q under the more relaxed notion of distance, but the converse is not true.

We also note that, in either notion, a tester with sample complexity that is polynomial in q, k and $1/\epsilon$ still queries a number of hyperedges of H that is exponential in q . This is due to the fact that, for a sample S of vertices, the induced subgraph $H[S]$ has up to $\binom{|S|}{q}$ potential hyperedges.

Testing satisfiability. An instance of a *constraint satisfaction problem* (CSP) on a set of n variables, that can each take one of k possible values, is a set of *constraints*, where each constraint is a predicate on a distinct set of variables. Throughout this thesis, we consider only q -uniform CSPs, in which every constraint is a predicate on exactly q variables. In particular, each set of q variables supports at most one constraint. This convention allows us to view a q -uniform CSP with a labeled q -uniform hypergraph, where each hyperedge carries a constraint. This differs slightly from some previous formulations of CSP testing, which allow multiple constraints on the same set of variables.

The CSP ϕ is *satisfiable* if and only if there is an assignment of values to the variables such that all of the constraints are satisfied. The (q, k) -SAT property is the set of all q -uniform CSPs on n variables that each can take one of k possible values. A q -uniform CSP ϕ on n variables is ϵ -far from (q, k) -SAT (or ϵ -far from satisfiable) if and only if at least $\epsilon \binom{n}{q}$ constraints need to be removed to turn it into a satisfiable CSP.

For any set S of variables, we write $\phi[S]$ to denote the restriction of the CSP ϕ to the set S . The restriction $\phi[S]$ consists of the set of constraints whose variables are all present in S .

Satisfiability testing was first studied by Alon and Shapira [AS03] who showed that the sample complexity of the canonical tester for (q, k) -SAT is independent of n and polynomial in $1/\epsilon$. Specifically, they showed that its sample complexity is roughly $2^{k^{2q}}/\epsilon^2$. Sohler [Soh12] improved the bounds to show that the sample complexity of the canonical tester is roughly k^{3q}/ϵ , and in particular has only a singly exponential dependence on q and a linear dependence on $1/\epsilon$.

Semi-homogeneous graph partition properties. Graph partition properties have played a central role in property testing ever since the original work of Goldreich, Goldwasser, and Ron [GGR98]. A graph property Π is a k -partition property if it can be described by a matrix of $2k^2$ parameters $\ell_{1,1}, \dots, \ell_{k,k}, u_{1,1}, \dots, u_{k,k}$ where $0 \leq \ell_{i,j} \leq u_{i,j} \leq 1$ for each $i, j \in [k]$ and where a graph $G = (V, E)$ has property Π if and only if there is a partition of the vertices of G into the sets $V_1, \dots, V_k$ for which the number of edges $|E(V_i, V_j)|$ between vertices in V_i and V_j satisfies

$$\ell_{i,j} \leq \frac{|E(V_i, V_j)|}{|V_i| \cdot |V_j|} \leq u_{i,j}$$

for all $i \neq j \in [k]$, and the number of edges $|E(V_i)|$ in the induced subgraph $G[V_i]$ satisfies

$$\ell_{i,i} \leq \frac{|E(V_i)|}{\binom{|V_i|}{2}} \leq u_{i,i}$$

for all $i \in [k]$. (Note that there are different classes of partition properties in which one can add other types of constraints, such as absolute bounds on the sizes of the parts and/or on the number of edges between different parts. In this paper, however, we focus our attention on the class of partition properties described above.)

An interesting subclass of partition properties is that of *homogeneous partition properties*, where for every $i, j \in [k]$ the density bounds satisfy $\ell_{i,j} = u_{i,j} \in \{0, 1\}$. That is, for every pair of parts V_i and V_j and any graph that satisfies a homogeneous property, all vertices in V_i must be connected to every vertex in V_j (if $\ell_{i,j} = u_{i,j} = 1$), or none of those edges must be present in G (if $\ell_{i,j} = u_{i,j} = 0$). Examples of homogeneous graph properties include the property of being an empty graph (which is a 1-partition property) and the property of being a collection of k isolated cliques.

The class of *semi-homogeneous partition properties* is a superclass of the set of all homogeneous partition properties obtained by relaxing the constraint on the densities to require $\ell_{i,j}, u_{i,j} \in \{0, 1\}$ but not that $\ell_{i,j} = u_{i,j}$. With this definition, the required density between some parts can be 0 or 1, as with homogeneous properties, but for some other pairs of parts we can have no restrictions. The class of semi-homogeneous partition properties is large and includes many natural properties that are not homogeneous, including most notably k -colorability.

This class of semi-homogeneous partition properties has connections to efficient distance approximation [FR21] and it is exactly the class of graph partition properties that can be tested with $\text{poly}(1/\epsilon)$ queries with one-sided error [NR18].

The previous best bounds on the sample complexity of testing k -SHPP are two incomparable bounds on the order of $k \log k / \epsilon^2$ [NR18] and k^6 / ϵ [Soh12].

Tolerant Testing. Tolerant testing, introduced by Parnas, Ron and Rubinfeld [PRR06], is a natural generalization of testing where the goal is to distinguish between graphs that are ϵ_1 -close to some property Π from graphs that are ϵ_2 -far from Π , where ϵ_1 -close means that at most $\epsilon_1 n^2$ edges need to be added or removed from the graph to make it have the property. Such a tester is called an (ϵ_1, ϵ_2) -tolerant tester, and the goal is to find the minimum sample complexity of any such tester, as a function of ϵ_1, ϵ_2 and the input size. It is desirable for such a tester to apply for any $\epsilon_1 < \epsilon_2$, referred to as a fully tolerant tester, but this is not necessary. For example, if $\epsilon_1 = 0$ then this corresponds to the standard notion of testing. In general, ϵ_1 may be a function of ϵ_2 .

Shown in [PRR06], any tester that takes a random sample S of vertices and inspects only $G[S]$ has a very small amount of tolerance by default based on the fact that a random subgraph won't find any violating edges in a graph that is very close to the property.¹ For example, the tester of Theorem 1.3 gives a (ϵ^4, ϵ) -tolerant tester with sample complexity $\tilde{O}(\rho^3 / \epsilon^2)$. However, this is described by [PRR06] as “weak” tolerance, and the real question is: when does there exist an efficient tolerant tester beyond this weak amount of tolerance given by the standard tester?

There have been no direct results analyzing this question for tolerantly testing ρ -INDEPSET. The best prior result comes from a general result for tolerant testing of general graph partition properties, which is a class of properties for which each property corresponds to graphs that can be partitioned into k parts with specific bounds on the edge

¹They actually show this for testers which make edge queries to the input which are uniformly distributed, but a similar argument applies for testers where the vertex samples are uniformly distributed.

densities between parts. This class of properties includes k -colorability, ρ -INDEPSET (with $k = 2$), and many others. Fiat and Ron show that for any $\epsilon_1 < \epsilon_2$ there is a (ϵ_1, ϵ_2) -tolerant tester for any graph partition property with sample complexity $\text{poly}(k, 1/(\epsilon_2 - \epsilon_1))$ [FR21, Appendix C]. These testers work by making multiple calls to a standard tester for general graph partition properties ([GGR98], [FMS10] or [SS24], for example) and as a result have large polynomial dependence on $1/\epsilon$. There is also a general result by Fischer and Newman saying that any graph property that has an ϵ -tester with sample complexity $s(\epsilon)$ has an (ϵ_1, ϵ_2) -tolerant tester, for any $\epsilon_1 < \epsilon_2$, with sample complexity that is tower type in $s(\epsilon_2 - \epsilon_1)$ [FN07], and this was recently improved by Shapira, Kushnir and Gishboliner to a doubly exponential bound in $s(\epsilon_2 - \epsilon_1)$ [SKG24].

2.2 Brief History of the Container Method

The container method is a combinatorial tool for characterizing independent sets in graphs and hypergraphs with specific properties. In particular, for many natural classes of graphs, such as regular graphs, nearly regular graphs, random graphs, or graphs that are sufficiently dense, graph container lemmas show that for any graph $G = (V, E)$ in the class, there exists a relatively small collection of *containers*, each of which is a subset of V and of size smaller than $|V|$, such that every independent set is contained in at least one of these containers. (See Section 1.2 for a simple example of containers, as well as more discussion on container lemmas and how to prove them.)

The graph container method was first used by Kleitman and Winston [KW82] to bound the number of square-free graphs. Later, Sapozhenko (see [Sap05] and the references therein) coined the term “containers”, and showed that the graph container method can also be used to resolve many other enumeration problems, such as counting the number of independent sets in d -regular graphs, counting the number of sum-free sets of integers and counting the number of antichains in posets. In 2015, Balogh, Morris, and Samotij [BMS15] and Saxton and Thomason [ST15] extended the container method to hypergraphs. The resulting hypergraph container method has since been used to obtain a number of striking results throughout combinatorics. (See the survey [BMS18] for an introduction to the method and its various applications.) More recently, there have been a number of improvements and variations of hypergraph container lemmas [NP25, Nen24, CS26]. The graph and hypergraph container methods have also recently found uses in algorithmic settings for counting independent sets in bipartite graphs [JPP23], for obtaining faster exact algorithms for many NP-complete problems on almost-regular graphs [Zam23], and for faster algorithms for satisfiability when the input formula has a “spread” property [HLMZ26].

In this section, we review some of the classic combinatorial applications of the graph and hypergraph container method and discuss some similarities with our application to property testing.

Counting Independent Sets in Regular Graphs. One of the most natural applications of the graph container method is to counting independent sets in d -regular graphs. Proving a conjecture of Granville, Alon first showed that the number of independent sets in a d -regular graph is at most $2^{\frac{n}{2}(1+O(d^{-0.1}))}$, and conjectured that the graph with $\frac{n}{2d}$ disjoint copies of $K_{d,d}$, which has $(2^{d+1} - 1)^{\frac{n}{2d}}$ independent sets, has the maximum number of independent sets [Alo91]. Sapozhenko improved the upper bound by Alon to $2^{\frac{n}{2}(1+O(\sqrt{\log(d)/d}))}$ using the graph container method [Sap01].² Sapozhenko's proof follows similar ideas to the Kleitman and Winston approach [KW82], and we outline the ideas below (note that our summary follows closely the presentation in the survey of Samotij on the topic [Sam15]).

The argument of Sapozhenko goes roughly as follows. We first observe that any d -regular graph is ϵ -far from ρ -INDEPSET, for appropriate values of ϵ and ρ . In particular, let $G = (V, E)$ be d -regular, then we claim that for any $\epsilon < \frac{d}{2n}$, every subset $U \subseteq V$ of size $\frac{n}{2} + \frac{\epsilon n^2}{d}$ has at least ϵn^2 edges. This follows from the fact that the vertices in $V \setminus U$ contribute to at most $d \left(\frac{n}{2} - \frac{\epsilon n^2}{d} \right)$ edges in the graph, and since there are exactly $\frac{dn}{2}$ edges in G , then

$$|E(U)| \geq \frac{dn}{2} - d \left(\frac{n}{2} - \frac{\epsilon n^2}{d} \right) \geq \epsilon n^2.$$

Hence, for any $\epsilon < \frac{d}{2n}$, a d -regular graph is ϵ -far from ρ -INDEPSET with $\rho = \frac{1}{2} + \frac{\epsilon n}{d}$. We will select $\epsilon = \Theta \left(\frac{\sqrt{d \log(d)}}{n} \right)$.

Now, we can apply the Kleitman/Winston container lemma, restated below, to get an upper bound on the number of independent sets in a d -regular graph. (See Section 1.2 for more discussion on Lemma 1.2.)

Lemma 1.2 ([KW82]). *For any ϵ, ρ , let $G = (V, E)$ be a graph on n vertices such that every induced subgraph on ρn vertices has at least ϵn^2 edges. Then there exists a set $\mathcal{C} \subseteq 2^V$ of containers that satisfies:*

1. $|\mathcal{C}| \leq \binom{n}{2/\epsilon},$

²Since then, Alon's conjecture has been proven, first by Kahn for bipartite graphs [Kah01], and then by Zhao for general graphs [Zha10].

2. for every $C \in \mathcal{C}$, $|C| \leq \left(\rho - \frac{\epsilon}{2\rho}\right)n$, and
3. for every independent set I in G , there exists $C \in \mathcal{C}$ such that $I \subseteq C$.

Using [Lemma 1.2](#), for any graph G that is ϵ -far from ρ -INDEPSET, we can upper bound the number of independent sets in G by the sum of the number of possible subsets of each container. In other words, the number of independent sets in G is at most

$$\sum_{C \in \mathcal{C}} 2^{|C|} \leq \binom{n}{2/\epsilon} \cdot 2^{\rho n} \leq 2^{\frac{2}{\epsilon} \log(en/2) + \rho n},$$

where the first inequality uses that $|C| \leq \binom{n}{2/\epsilon}$ and $|C| \leq \rho n$ for every $C \in \mathcal{C}$. Finally, by substituting $\epsilon \sim \frac{\sqrt{d \log(d)}}{n}$ and $\rho = \frac{1}{2} + \frac{en}{d} = \frac{1}{2} + \Theta(\sqrt{\log(d)/d})$ and simplifying, we find that the number of independent sets in a d -regular graph G is at most

$$2^{\frac{n}{2} \left(1 + O\left(\sqrt{\frac{\log(d)}{d}}\right)\right)}.$$

More Counting Applications. There are many other counting applications of the container method. In particular, many counting problems can be translated into problems about counting independent sets in an appropriately constructed graph (or hypergraph), and then the ideas described above can be used to give an upper bound on the number of independent sets. In fact, the original application of Kleitman and Winston to counting square-free graphs can be viewed as following this general approach [\[KW82, Sam15\]](#). To illustrate this type of application we briefly explain three examples of problems that can be converted into problems about independent sets.

1. Kleitman and Winston used the container method to count the number of square-free graphs [\[KW82\]](#). The approach is roughly the following. Consider the process of constructing a C_4 -free graph G_n on n vertices one vertex at a time, where in each step the new vertex added has (roughly) the lowest degree in the new graph (such an ordering always exists). We are going to count the number of C_4 -free graphs by counting the number of ways to construct a graph in this way. Suppose G_{n-1} is a C_4 -free graph on $n-1$ vertices. Let H_{n-1} be the graph corresponding to 2-paths in G_{n-1} . In other words, H_{n-1} is on the same $n-1$ vertices as G_{n-1} , and the edges of H_{n-1} correspond to paths of length 2 in G_{n-1} . Then observe that if a new vertex v with degree d is added to G_{n-1} to construct G_n , G_n is C_4 -free if and only if the d

neighbours of v form an independent set in H_{n-1} (otherwise there would be a copy of C_4 in G_n). Hence, we can upper bound the number of ways to add vertex v with degree d by the number of independent sets of size d in H_{n-1} .

In order to show that H_{n-1} has sufficiently high density they use that G_{n-1} has large minimum degree (compared to the degree of the new vertex added), and so H_{n-1} should also be sufficiently dense. See [KW82] or the survey [Sam15] for more details.

2. The graph container method can also be used to count the number of sum-free subsets of $[n]$ [Sap08, Sam15]. We outline one idea below of how to convert the problem of counting sum-free subsets into a problem about counting independent sets in an auxiliary graph. First, a sum-free subset of $[n]$ is defined to be a set $A \subseteq [n]$ for which there is no triple $x, y, z \in A$ with $x + y = z$. Let A be a subset of $[n]$, and let $S \subseteq A$ be a subset of A . Construct a graph G_S on vertex set $[n]$, and add an edge for every pair $(x, y) \in [n]$ such that $x + s = y$ or $x + y = s$ for some $s \in S$. The key observation is that if A is a sum-free subset then $A \setminus S$ forms an independent set in G_S .

With an appropriate way of selecting S from A , we can upper bound the number of sum-free subsets by counting the number of ways to select S , and then, for each fixed S , upper bounding the number of choices for $A \setminus S$ by the number of independent sets in G_S . See [Sap08] or [Sam15] for additional details.

3. A standard application of the *hypergraph* container method is counting F -free graphs for some fixed forbidden graph F (see, for example, the survey [BMS18]). As with the examples above, the key idea is to encode the problem into a problem about independent sets. For example, for counting triangle-free graphs, let H be the 3-uniform hypergraph with vertex set corresponding to the edges of K_n (so there are $N = \binom{n}{2}$ vertices in H) and an edge for every triple $e_1, e_2, e_3 \in V(H)$ such that the edges e_1, e_2, e_3 form a triangle in K_n . Hence, observe that a triangle-free graph corresponds to an independent set in H .

In order to apply the container method we need that the hypergraph is sufficiently dense, and this requires proving a “supersaturation lemma”. For triangle-freeness the supersaturation lemma says that for any δ , there exists ϵ such that if G is a graph on n vertices with at least roughly $(\frac{1}{2} + \delta)\binom{n}{2}$ edges then G has at least ϵn^3 triangles. This supersaturation lemma translates to H being $\frac{\epsilon}{N^{3/2}}$ -far from ρ -INDEPSET,³ with $\rho \approx (\frac{1}{2} + \delta)$. Using this supersaturation lemma, it is possible to prove a container

³Here ρ -INDEPSET corresponds to the *hypergraph* property of having an independent set of size ρN .

lemma (like [Lemma 1.2](#)) for covering the independent sets in H , where each container is of size at most ρN .

A similar approach as above works for any F , and in general H will be a $|E(F)|$ -uniform hypergraph.

Other Applications of the Container Method The applications mentioned above correspond to counting problems, however there are also many applications of the container method that are not explicit counting problems. We outline one classic example below: a random analogue of Mantel's theorem.

Mantel's theorem says that every graph on n vertices with more than $n^2/4$ edges contains a triangle [\[Man07\]](#). A random analogue of this was first studied by Frankl and Rödl [\[FR86\]](#), where they showed that, with high probability, for any $\alpha > 0$ and p at least roughly $1/\sqrt{n}$, every subgraph⁴ $G \subset G(n, p)$ on at least $p(1 + \alpha)n^2/4$ edges has a triangle.⁵ A similar result can be shown using the container method (see [\[BMS18\]](#)). The approach is as follows.

As with the application to counting triangle-free graphs described above, we start with constructing a 3-uniform hypergraph H on $N = \binom{n}{2}$ vertices (corresponding to the edges of K_n) and whose edges correspond to the triangles of K_n . Triangle-free graphs correspond to independent sets in H . Again using the supersaturation lemma described above, it is possible to prove a container lemma (like [Lemma 1.2](#)) for covering the independent sets in H , where each container is of size at most ρN , with $\rho \approx (\frac{1}{2} + \delta)$ for some constant δ .

Now, we wish to upper bound the probability that $G(n, p)$ contains a triangle-free subgraph with more than $p(1 + \alpha)n^2/4 \approx \frac{p(1+\alpha)}{2} \binom{n}{2}$ edges. First observe that the random graph $G(n, p)$ corresponds to taking a random set of vertices in H where each vertex is selected with probability p . Hence, since any triangle-free subgraph in $G(n, p)$ corresponds to an independent set in H and all independent sets are covered by the containers of H , then the probability that $G(n, p)$ has a large triangle-free subgraph can be upper bounded by the probability that a random vertex sample $S \subseteq V(H)$ contains a large independent set. For each container C of H , we know the container size is at most $\rho N \approx (\frac{1}{2} + \delta) \binom{n}{2}$, and so the expected number of vertices in $S \cap C$ is $p \cdot (\frac{1}{2} + \delta) \binom{n}{2}$. Hence, the probability that $S \cap C$ is at least $\frac{p(1+\alpha)}{2} \binom{n}{2}$ can be upper bounded using a Chernoff bound (for appropriate selection of $\delta < \alpha$). Then applying a union bound over the collection of containers completes the upper bound on the probability that S has a large independent set. For full details see [\[BMS18\]](#).

⁴Here we use subgraph to mean a subset of edges, *not* an induced subgraph.

⁵ $G(n, p)$ is the random graph on n vertices where each edge occurs with probability p .

Similarities with Our Application to Property Testing. Our application of the container method to property testing has many similarities with the classic combinatorial applications of the container method outlined above. First, while our theorems are stated in terms of the sample complexity of testing various graph and hypergraph properties, the key technical results can be viewed as counting results. For example, the key technical result in [Theorem 1.3](#) says that if the input graph G is ϵ -far from ρ -INDEPSET then the probability that a random induced subgraph on roughly $s \sim \rho^3/\epsilon^2$ vertices satisfies ρ -INDEPSET is at most $1/3$. In other words, [Theorem 1.3](#) gives an upper bound of $\frac{1}{3}\binom{n}{s}$ on the number of induced subgraphs of size s that satisfy ρ -INDEPSET.

A second connection has to do with the characteristics of the graphs and hypergraphs constructed in the above examples. The combinatorics literature on the container method does not usually use the terminology of “ ϵ -far.” However, in many cases this is exactly what is shown for the constructed graphs and hypergraphs. For example, in the example of counting independent sets in d -regular graphs, one of the steps is showing that d -regular graphs are ϵ -far from ρ -INDEPSET with $\rho = \frac{1}{2} + \frac{\epsilon n}{d}$. Further, in the applications to counting triangle-free graphs and the application to the randomized analogue of Mantel’s theorem, the supersaturation lemma can be interpreted as saying that the constructed hypergraph H is far from having a large independent set.

Finally, the analysis of the sampling in the application to the randomized analogue of Mantel’s theorem is very similar to the analysis of our sampling algorithms for property testing. In more detail, in the proof of the randomized analogue of Mantel’s theorem, a random graph $G(n, p)$ can be viewed as taking a random set S of vertices in the constructed hypergraph H , where each vertex is sampled with probability p . Hence, the probability that $G(n, p)$ has a large triangle-free subgraph can be upper bounded by the probability that a random sample of vertices from H contains a large independent set. This is exactly what we analyze in our proof of [Theorem 1.3](#): in a graph that is ϵ -far from ρ -INDEPSET, we upper bound the probability that a random sample of vertices has a large independent set.

2.3 Useful Bounds

Throughout this thesis we use the following inequalities to simplify expressions.

Exponential Approximation: $1 + x \leq e^x$ for any $x \in \mathbb{R}$.

Binomial Approximation: $\left(\frac{n}{\ell}\right)^\ell \leq \binom{n}{\ell} \leq \left(\frac{en}{\ell}\right)^\ell$ for any $1 \leq \ell \leq n$. We sometimes use the simpler upper bound of $\binom{n}{\ell} \leq n^\ell$.

Markov's Inequality: If X is a non-negative random variable and $a > 0$ then

$$\Pr[X \geq a] \leq \frac{\mathbb{E}[X]}{a}.$$

We also use the following form of Chernoff's bound for hypergeometric distributions.

Lemma 2.2 (Chernoff's Bound). *Let X be drawn from the hypergeometric distribution $H(N, K, m)$ where m elements are drawn without replacement from a population of N elements, K of which are marked, and X represents the number of marked elements that were drawn. Then for any $\vartheta \geq \mathbb{E}[X]$,*

$$\Pr[X \geq \vartheta] \leq \exp\left(-\frac{(\vartheta - \mathbb{E}[X])^2}{\vartheta + \mathbb{E}[X]}\right).$$

Proof. A standard multiplicative form of Chernoff's bound states that for all $\delta > 0$, the random variable X satisfies $\Pr[X \geq (1 + \delta)\mathbb{E}[X]] \leq \exp\left(-\frac{\delta^2 \mathbb{E}[X]}{2 + \delta}\right)$. (This standard bound is usually stated for sums of independent variables, but the same bound also applies to hypergeometric random variables as well. See, e.g., [Mul18]). Using the identity $\vartheta = \left(1 + \frac{\vartheta - \mathbb{E}[X]}{\mathbb{E}[X]}\right) \mathbb{E}[X]$, we can apply this inequality with the parameter $\delta = \frac{\vartheta - \mathbb{E}[X]}{\mathbb{E}[X]}$ and simplify. $\square$

Chapter 3

A Warmup Application of the Container Method

In this chapter we use the container method to prove the following theorem on testing the ρ -INDEPSET property, reproducing the prior best bound by Feige, Langberg and Schetchman [FLS04].

Theorem 3.1. *The sample complexity of the ρ -INDEPSET property is*

$$\mathcal{S}_{\rho\text{-INDEPSET}}(n, \epsilon) = O\left(\frac{\rho^4}{\epsilon^3} \ln\left(\frac{1}{\epsilon}\right)\right).$$

The proof of [Theorem 3.1](#) demonstrates the basic container method framework and is significantly simpler than the proofs in [Chapters 4 to 6](#).

3.1 Container Procedure and Container Lemma

We utilize the following procedure ([Algorithm 1](#)), first used by Kleitman and Winston [KW82], that generates a fingerprint and container pair, denoted $F(I)$ and $C(I)$, given an independent set I . The procedure works by greedily selecting the vertex v from the independent set with highest degree in the current “candidate” container, and adding this vertex to the fingerprint (see [Figs. 1.4 and 1.5](#) from [Chapter 1](#) for an example and motivation of this process). Then, all vertices that neighbour v and all vertices that had degree higher than v are removed from the container because they cannot be part of the independent set. The process is repeated for $2\rho/\epsilon$ iterations.

Algorithm 1: SIMPLE FINGERPRINT & CONTAINER GENERATOR

Input: A graph $G = (V, E)$, an independent set $I \subseteq V$, and parameters ρ, ϵ .

```
1 Initialize  $F \leftarrow \emptyset$  and  $C \leftarrow V$ ;  
2 for  $t = 1, 2, \dots, 2\rho/\epsilon$  do  
3    $v \leftarrow$  the vertex in  $I$  with largest degree in  $G[C]$ ;  
   // Add  $v$  to the fingerprint1  
4    $F \leftarrow F \cup \{v\}$ ;  
   // Remove all the neighbours of  $v$  and all vertices with higher  
   // degree than  $v$  from the container  
5    $C \leftarrow C \setminus \{w \in C : (v, w) \in E \text{ or } \deg_{G[C]}(w) > \deg_{G[C]}(v)\}$ ;  
6 Return  $F, C$ ;
```

By construction, the fingerprint $F(I)$ and container $C(I)$ of an independent set I satisfy a number of useful properties. For instance, we have that $F(I) \subseteq I \subseteq C(I)$ based on the fact that every vertex added to F comes from I , and vertices are only removed from C if they are not contained in the independent set. Further, while the container $C(I)$ is produced on input I , it is uniquely defined by the fingerprint $F(I)$. In other words, $C(F(I)) = C(I)$ because if we instead run the container procedure on input $F(I)$, the procedure will select the same vertices to the new fingerprint in each iteration.

The above observations apply for any graph and can be used to construct a collection of containers by running the container procedure on every independent set in the graph. While there could be up to 2^n independent sets in the graph, using the observation that the containers are uniquely defined by the much smaller fingerprints, we can get a much better bound on the size of the collection of containers.

For our application to property testing, we also desire an upper bound on the size of the individual containers. For this, we consider the case that G is ϵ -far from having a ρn independent set, and prove the following graph container lemma. (Note that the following lemma is a slight reformulation of [Lemma 1.2](#). The lemma is phrased in terms of a collection of fingerprints instead of a collection of containers, but a collection of containers can be constructed by running the container function on every fingerprint.)

Lemma 3.2. *Let $G = (V, E)$ be a graph on n vertices that is ϵ -far from the ρ -INDEPSET*

¹Note that in [Algorithm 1](#) we allow the same vertex to be selected to the fingerprint multiple times. This is done to make the analysis simpler and is only relevant when the independent set is smaller than $2\rho/\epsilon$.

property. Then there exists a collection $\mathcal{F} \subseteq 2^V$ of fingerprints and a container function $\mathcal{C} : \mathcal{F} \rightarrow 2^V$ that satisfy the following two properties.

1. For every independent set I in G there exists $F \in \mathcal{F}$ such that $F \subseteq I \subseteq \mathcal{C}(F)$.
2. For every fingerprint $F \in \mathcal{F}$, the cardinality of F and of its associated container $\mathcal{C}(F)$ are bounded by

$$|F| \leq \frac{2\rho}{\epsilon} \quad \text{and} \quad |\mathcal{C}(F)| \leq \left(\rho - \frac{\epsilon}{2\rho}\right)n.$$

Proof. Consider the collection of fingerprints produced by running the procedure [Algorithm 1](#) on every independent set in G :

$$\mathcal{F} = \{F(I) : I \text{ is an independent set in } G\}.$$

Let the container function $\mathcal{C}(\cdot)$ be the container function $C(\cdot)$ from [Algorithm 1](#).

First observe that for any independent set I in G , there exists $F \in \mathcal{F}$ such that $F \subseteq I \subseteq \mathcal{C}(F)$ based on the construction of $\mathcal{F}$ and the observations described above. Next, observe that for any $F \in \mathcal{F}$, the bound $|F| \leq 2\rho/\epsilon$ follows from the fact that the procedure is terminated after $2\rho/\epsilon$ steps.

All that remains is showing the upper bound on the size of $\mathcal{C}(F)$, and for this we require the fact that G is ϵ -far from the the ρ -INDEPSET property. First, we claim that the maximum degree in $G[\mathcal{C}(F)]$ is at most $\frac{\epsilon}{2\rho}n$. This can be shown by observing that the candidate container is initialized to V , of size n , and so for at least one of the $2\rho/\epsilon$ iterations the vertex selected during the iteration has degree at most $\frac{\epsilon}{2\rho}n$, otherwise we would be left with a final container of negative size. During this iteration, all vertices with higher degree are removed, and hence the final container satisfies $\Delta(G[\mathcal{C}(F)]) \leq \frac{\epsilon}{2\rho}n$.

Now, for sake of contradiction consider the case that $|\mathcal{C}(F)| > \left(\rho - \frac{\epsilon}{2\rho}\right)n$. By the claim above, the vertices in $G[\mathcal{C}(F)]$ have degree at most $\frac{\epsilon}{2\rho}n$, and so by selecting at least $\left(\rho - \frac{\epsilon}{2\rho}\right)n$ vertices from $\mathcal{C}(F)$ and at most $\frac{\epsilon}{2\rho}n$ additional arbitrary vertices, we find a set of ρn vertices that has strictly less than ϵn^2 edges in G . This contradicts the fact that G is ϵ -far from the the ρ -INDEPSET property, and hence $|\mathcal{C}(F)| \leq \left(\rho - \frac{\epsilon}{2\rho}\right)n$. $\square$

3.2 Proof of Theorem

Now, using [Lemma 3.2](#) and a Chernoff Bound ([Lemma 2.2](#)), we prove [Theorem 3.1](#). The tester is the natural tester: take a random sample S of vertices and check if it has a large independent set. The challenging part of the proof is showing that when G is ϵ -far from ρ -INDEPSET then the sample contains a large independent set with low probability.

The main idea is to use the collection $\mathcal{F}$ of fingerprints and associated containers from [Lemma 3.2](#). Using that every independent set is contained in at least one container, a first attempt might be to upper bound the probability that S has at least ρs vertices from a single container C , and take the union bound over all containers in $\mathcal{C}$. Unfortunately this isn't strong enough to prove the desired bound (as seen in [Chapter 1](#), this results in a $\log n$ factor in the sample complexity). Instead, we use the associated fingerprints, and, in particular, the observation that for every independent set I in G there exists $F \in \mathcal{F}$ such that $F \subseteq I \subseteq \mathcal{C}(F)$. Hence, S contains an independent set of size ρs only when there exists $F \in \mathcal{F}$ such that S contains at least ρs vertices from $\mathcal{C}(F)$ **and** S contains F . The requirement that S contains F is what allows us to remove the $\log n$ factor.²

Proof. (of [Theorem 3.1](#)) Let S be a random set of $s = c \cdot \frac{\rho^4}{\epsilon^3} \ln\left(\frac{1}{\epsilon}\right)$ vertices drawn uniformly at random from V without replacement, where c is a large enough constant.³

If G contains a ρn independent set, then S contains at least ρs vertices from this independent set with probability at least $\frac{1}{2}$, since the number of such vertices follows a hypergeometric distribution, and the median of this distribution is at least ρs [[Neu70](#), [KB80](#)].

In the remainder of the proof we upper bound the probability that S contains a ρs -independent set when G is ϵ -far from containing a ρn -independent set. We consider the collection of fingerprints $\mathcal{F} \subseteq 2^V$ and container function $\mathcal{C} : \mathcal{F} \rightarrow 2^V$ from [Lemma 3.2](#). By the lemma, for every independent set I in G there exists $F \in \mathcal{F}$ such that $F \subseteq I \subseteq \mathcal{C}(F)$. Hence, S contains an independent set of size ρs only when there exists $F \in \mathcal{F}$ such that S contains at least ρs vertices from $\mathcal{C}(F)$ **and** S contains F . In other words, the probability that S contains a ρs independent set is at most

$$\sum_{F \in \mathcal{F}} \Pr[F \subseteq S] \cdot \Pr[|\mathcal{C}(F) \cap S| \geq \rho s \mid F \subseteq S].$$

²An alternate way of viewing this is the following: instead of union bounding over all possible containers, we union bound over containers with fingerprints coming from the $\binom{|S|}{2\rho/\epsilon}$ possible fingerprint locations within the sampled vertices.

³Note that here and throughout we ignore all integer rounding issues as they do not affect the asymptotics of the final results.

For a specific $F \in \mathcal{F}$, the probability that S contains F is at most

$$\frac{\binom{n-|F|}{s-|F|}}{\binom{n}{s}} = \frac{\binom{s}{|F|}}{\binom{n}{|F|}},$$

because there are at most $\binom{n}{s}$ ways to choose S , and $\binom{n-|F|}{s-|F|}$ ways to choose an S that includes F . The probability that S contains at least ρs vertices from $\mathcal{C}(F)$, conditioned on the fact that S contains F , can be upper bounded using a Chernoff bound. In particular, let X denote the number of vertices among the other $s - |F|$ sampled vertices that belong to $\mathcal{C}(F)$. Using that $|\mathcal{C}(F)| \leq \left(\rho - \frac{\epsilon}{2\rho}\right)n$, the expected value of X is

$$\mathbb{E}[X] \leq \left(\rho - \frac{\epsilon}{2\rho}\right)(s - |F|) < \rho s - \frac{\epsilon s}{2\rho} \leq \rho s - |F| - \frac{\epsilon s}{4\rho},$$

where the last inequality uses the fact that $s = c \cdot \frac{\rho^4}{\epsilon^3} \ln(\frac{1}{\epsilon})$ for a large enough constant c and that the problem is non-trivial only when $\epsilon < \rho^2$.

By the Chernoff bound in [Lemma 2.2](#), the probability that we draw at least $\rho s - |F|$ vertices from C in the other $s - |F|$ vertices drawn to form S is

$$\Pr[X \geq \rho s - |F|] \leq \exp\left(\frac{-(\rho s - |F| - \mathbb{E}[X])^2}{\rho s - |F| + \mathbb{E}[X]}\right) \leq \exp\left(-\frac{\epsilon^2 s}{32\rho^3}\right).$$

Substituting back into the probability bound above, the probability that S contains a ρs independent set is at most

$$\sum_{F \in \mathcal{F}} \frac{\binom{s}{|F|}}{\binom{n}{|F|}} \cdot \exp\left(-\frac{\epsilon^2 s}{32\rho^3}\right).$$

We then split up the sum based on the size of the fingerprints to find

$$\sum_{F \in \mathcal{F}} \frac{\binom{s}{|F|}}{\binom{n}{|F|}} \cdot \exp\left(-\frac{\epsilon^2 s}{32\rho^3}\right) \leq \sum_{j=1}^{2\rho/\epsilon} \sum_{\substack{F \in \mathcal{F} \\ |F|=j}} \frac{\binom{s}{j}}{\binom{n}{j}} \cdot \exp\left(-\frac{\epsilon^2 s}{32\rho^3}\right).$$

Finally, we use that the number of fingerprints of size j is at most $\binom{n}{j}$ to upper bound the above expression by

$$\sum_{j=1}^{2\rho/\epsilon} \binom{s}{j} \exp\left(-\frac{\epsilon^2 s}{32\rho^3}\right) \leq \sum_{j=1}^{2\rho/\epsilon} \exp\left(j \ln s - \frac{\epsilon^2 s}{32\rho^3}\right) \leq \frac{2\rho}{\epsilon} \exp\left(\frac{2\rho}{\epsilon} \ln s - \frac{\epsilon^2 s}{32\rho^3}\right) < 1/3$$

where the first inequality uses the upper bound $\binom{s}{j} \leq s^j$, the second inequality uses that $j \leq 2\rho/\epsilon$, and the final inequality uses that $s = c \cdot \frac{\rho^4}{\epsilon^3} \ln(\frac{1}{\epsilon})$ for a large enough constant c . Hence, the probability that S contains an independent set of size at least ρs is less than $1/3$. $\square$

Chapter 4

Testing Independent Sets

In this section we prove [Theorem 1.3](#), restated below, by giving a new upper bound on the sample complexity of testing the ρ -INDEPSET property.

Theorem 1.3. *The sample complexity of ϵ -testing the ρ -INDEPSET property is $\tilde{O}(\frac{\rho^3}{\epsilon^2})$.*

4.1 Proof Overview

The proof of [Theorem 1.3](#) follows the same structure as the proof of [Theorem 3.1](#) from [Chapter 3](#). In particular, we again analyze the natural tester that checks whether the sampled induced subgraph $G[S]$ has an independent set of size $\rho|S|$. In the case G has an independent set of size ρn , this tester accepts with probability at least $\frac{1}{2}$.¹ The challenge in the analysis, as is usually the case in property testing, is in showing that the tester rejects all graphs that are far from ρ -INDEPSET with sufficiently large probability. We again address this challenge with the use of the graph container lemma, however this time we prove a stronger graph container lemma, restated below.

Lemma 1.4. *Let $G = (V, E)$ be a graph on n vertices that is ϵ -far from the ρ -INDEPSET property. Then there exists a collection $\mathcal{F} \subseteq 2^V$ of fingerprints and a container function $\mathcal{C} : \mathcal{F} \rightarrow 2^V$ that satisfy the following two properties.*

1. *For every independent set I in G of cardinality $|I| \geq \frac{8\rho^2 \ln(\frac{2\rho}{\epsilon})}{\epsilon}$, there is a fingerprint $F \in \mathcal{F}$ such that $F \subseteq I \subseteq \mathcal{C}(F)$.*

¹See the [Chapter 2](#) for a discussion related to this choice of acceptance probability.

2. For every fingerprint $F \in \mathcal{F}$, the cardinality of F and of its associated container $\mathcal{C}(F)$ are bounded by

$$|F| \leq \frac{8\rho^2 \ln(\frac{2\rho}{\epsilon})}{\epsilon} \quad \text{and} \quad |\mathcal{C}(F)| \leq \left(\rho - |F| \cdot \frac{\epsilon}{8\rho \ln(\frac{2\rho}{\epsilon})} \right) n.$$

The key difference between [Lemma 1.4](#) and the simpler graph container lemma from [Chapter 3](#) is that, for each fingerprint $F \subseteq V$, there is a trade off between the size of the fingerprint and the size of the associated container $\mathcal{C}(F)$. In other words, the larger the container, the smaller the fingerprint. Hence, there can't be too many large containers because there can't be too many small fingerprints.

4.2 The Fingerprint and Container Procedure

The fingerprint and container procedure used to prove [Lemma 1.4](#), written in [Algorithm 2](#), is nearly identical to the procedure used in [Chapter 3](#) with the exception of the stopping condition. In particular, given a graph G and independent set I , the first fingerprint of an independent set I is simply the vertex v contained in I with the highest degree in G , where ties are broken based on an assumed ordering of the vertices V . Since I is an independent set, any neighbour of v must not be contained in I and, furthermore, by the fact that v has the highest degree in G of all vertices in I , any vertex with higher degree in G cannot be in I . The associated container is constructed by removing all vertices from these two collections. The procedure is repeated until every vertex from I has been selected to the fingerprint.

The greedy algorithm for generating fingerprints and containers was originally defined with an appropriate stopping condition so that each independent set generates a single fingerprint and container. For our proofs, however, it will be more convenient to work directly with the sequence of fingerprints $F_1, F_2, \dots, F_{|I|}$ and their corresponding containers $C_1, C_2, \dots, C_{|I|}$ generated by the algorithm for the independent set I . Furthermore, it is also convenient to extend the definition with $C_t = F_t = I$ for all $t > |I|$. We refer to F_t and C_t as the t -th fingerprint and t -th container of I , respectively.

When we consider the fingerprints or containers of multiple independent sets, we write $F_t(I)$ and $C_t(I)$ to denote the t -th fingerprint and container of the independent set I . (When the current context specifies a single independent set, we will continue to write only F_t and C_t to keep the notation lighter.)

Algorithm 2: FINGERPRINT & CONTAINER GENERATOR

Input: A graph $G = (V, E)$ and an independent set $I \subseteq V$

```
1 Initialize  $F_0 \leftarrow \emptyset$  and  $C_0 \leftarrow V$ ;  
2 for  $t = 1, 2, \dots, |I|$  do  
3    $v_t \leftarrow$  the vertex in  $I \setminus F_{t-1}$  with largest degree in  $G[C_{t-1}]$ ;  
   // Add  $v_t$  to the fingerprint  
4    $F_t \leftarrow F_{t-1} \cup \{v_t\}$ ;  
   // Remove all the neighbours of  $v_t$  from the container  
5    $C_t \leftarrow C_{t-1} \setminus \{w \in C_{t-1} : (v_t, w) \in E\}$ ;  
   // And remove all vertices with higher degree than  $v_t$  in  $G[C_{t-1}]$   
6    $C_t \leftarrow C_t \setminus \{w \in C_{t-1} : \deg_{G[C_{t-1}]}(w) > \deg_{G[C_{t-1}]}(v_t)\}$ ;  
7 Return  $F_1, \dots, F_{|I|}$  and  $C_1, \dots, C_{|I|}$ ;
```

By construction, the sequences of fingerprints and containers of an independent set satisfy a number of useful properties. For instance, we have

$$F_1 \subseteq F_2 \subseteq \dots \subseteq F_{|I|} = I \subseteq C_{|I|} \subseteq C_{|I|-1} \subseteq \dots \subseteq C_2 \subseteq C_1 \quad (4.1)$$

so that for each $t = 1, 2, \dots, |I|$, $F_t \subseteq I \subseteq C_t$.

The construction of the algorithm also guarantees that the t -th container of an independent set is the same as the t -th container of its t -th fingerprint.

Proposition 4.1. *For any graph $G = (V, E)$, any independent set I in G , and any t , the fingerprint $F_t(I)$ and container $C_t(I)$ of I satisfy*

$$C_t(F_t(I)) = C_t(I).$$

Proof. If $t > |I|$, then $F_t(I) = I = C_t(I)$ and the equality holds. Now consider the case that $t \leq |I|$. If $v_1, \dots, v_t$ are the vertices selected in the first t rounds of the greedy algorithm on input I , then the algorithm selects the same vertices and forms the same container C_t when provided with input $F_t(I)$ instead of I . $\square$

4.3 Container Shrinking Lemma

The main tool in the proof of the Graph Container Lemma for testing independent sets, [Lemma 1.4](#), is a shrinking lemma that, informally speaking, says that many vertices are

removed in each round of [Algorithm 2](#).

To gain some intuition for this shrinking lemma, observe that if $|C_t| \geq \rho n$, then $G[C_t]$ has edge density at least $\frac{2\epsilon}{\rho^2}$. Then, it is not hard to show that at least roughly $\frac{\epsilon}{2\rho^2}|C_t|$ vertices are removed when constructing C_{t+1} , which means that $|C_{t+1}| \leq \left(1 - \frac{\epsilon}{2\rho^2}\right)|C_t|$. This is because in the t -th step of [Algorithm 2](#) either v_t has degree larger than $\frac{\epsilon}{2\rho^2}|C_t|$ in $G[C_t]$, or all vertices with degree larger than $\frac{\epsilon}{2\rho^2}|C_t|$ in $G[C_t]$ are removed when constructing C_{t+1} , and there are at least $\frac{\epsilon}{2\rho^2}|C_t|$ such vertices.

The following lemma shows that if $G[C_t]$ contains a large sparse subgraph, then the container, or more specifically the number of vertices in the container outside of the sparse subgraph, shrinks by much more than a factor of $(1 - \frac{\epsilon}{2\rho^2})$.

Lemma 4.2 (Container Shrinking Lemma). *Let $G = (V, E)$ be a graph on n vertices that is ϵ -far from the ρ -INDEPSET property, let I be an independent set in G , and let $t < |I|$ be an index for which the t -th container C_t of I has cardinality $|C_t| \geq \rho n$. For any $\alpha > 0$, if there exists a subset $C \subseteq C_{t+1}$ of $(\rho - \alpha)n$ vertices where $G[C]$ contains at most $\frac{\epsilon}{4}n^2$ edges, then*

$$|C_{t+1} \setminus C| \leq \left(1 - \frac{\epsilon}{4\rho\alpha}\right) |C_t \setminus C|.$$

Proof. First recall that $C \subseteq C_{t+1} \subseteq C_t$, and so it holds that $|C_{t+1} \setminus C| = |C_t \setminus C| - |C_t \setminus C_{t+1}|$. To prove the lemma we will give a lower bound on $|C_t \setminus C_{t+1}|$. First suppose that C contains a vertex v with degree at least $\frac{\epsilon}{4\rho\alpha}|C_t \setminus C|$ in the graph $G[C_t]$. Then, when constructing C_{t+1} from C_t , at least this number of vertices are removed from C_t since the vertex v_{t+1} selected by the greedy algorithm has degree at least as large as that of v in $G[C_t]$, otherwise v would not have been in C_{t+1} (and so therefore would not be in C).

Consider now the setting where all the vertices in C have degree at most $\frac{\epsilon}{4\rho\alpha}|C_t \setminus C|$ in $G[C_t]$. We claim that $G[C_t \setminus C]$ contains at least $\frac{\epsilon}{4\rho\alpha}|C_t \setminus C|^2$ edges. This claim suffices to complete the proof of the lemma, since it implies that at least $\frac{\epsilon}{4\rho\alpha}|C_t \setminus C|$ vertices in $C_t \setminus C$ have degree at least $\frac{\epsilon}{4\rho\alpha}|C_t \setminus C|$ in $G[C_t \setminus C]$, and thus that at least this many vertices are removed from C_t when constructing C_{t+1} by [Algorithm 2](#).

We now complete the proof of the claim that $G[C_t \setminus C]$ contains at least $\frac{\epsilon}{4\rho\alpha}|C_t \setminus C|^2$ edges. Let R be chosen uniformly at random from all subsets of $C_t \setminus C$ of size αn . Since G is ϵ -far from having independent sets of size ρn , the induced subgraph $G[R \cup C]$ contains at least ϵn^2 edges. By the lemma hypothesis, $G[C]$ contains at most $\frac{\epsilon}{4}n^2$ edges. Finally, by the maximum degree of the vertices of C in $G[C_t]$, the number of edges with one endpoint

in C and one endpoint in $C_t \setminus C$ is at most $|C| \cdot \frac{\epsilon}{4\rho\alpha} |C_t \setminus C| \leq \frac{\epsilon}{4\alpha} |C_t \setminus C| n$. Each such edge is included in $G[R \cup C]$ with probability $|R|/|C_t \setminus C|$, and so the expected number of edges between C and R is at most $\frac{\epsilon}{4\alpha} |R| n = \frac{\epsilon}{4} n^2$. Therefore, the expected number of edges in $G[R]$ is at least

$$\frac{\epsilon}{2} n^2 = \frac{\epsilon}{2\alpha^2} |R|^2 \geq \frac{\epsilon}{\alpha^2} \binom{|R|}{2}.$$

In other words, the random graph R chosen uniformly at random from the subgraphs of $C_t \setminus C$ of cardinality αn has expected density at least $\frac{\epsilon}{\alpha^2}$. This implies that $C_t \setminus C$ also has density at least $\frac{\epsilon}{\alpha^2}$. So it contains at least $\frac{\epsilon}{\alpha^2} \binom{|C_t \setminus C|}{2} \geq \frac{\epsilon}{\rho\alpha} \binom{|C_t \setminus C|}{2} \geq \frac{\epsilon}{4\rho\alpha} |C_t \setminus C|^2$ edges, as we wanted to show. $\square$

4.4 Proof of Graph Container Lemma

In this section, we prove the Graph Container Lemma for testing independent sets. We first use the Container Shrinking Lemma to prove the following alternative formulation of the graph container lemma for testing independent sets. We then show how this alternative graph container lemma implies [Lemma 1.4](#).

Lemma 4.3. *Let $G = (V, E)$ be a graph on n vertices that is ϵ -far from the ρ -INDEPSET property. For any independent set I in G , there exists an index $t \leq \frac{8\rho^2}{\epsilon} \ln(\frac{2\rho}{\epsilon})$ such that the size of the t -th container of I is bounded by*

$$|C_t| \leq \left(\rho - t \cdot \frac{\epsilon}{8\rho \ln(\frac{2\rho}{\epsilon})} \right) n \quad (4.2)$$

and $G[C_t]$ contains at most $\epsilon n^2/4$ edges.

Proof. Let C denote the first container in the sequence $C_1, C_2, \dots$ that contains at most $\frac{\epsilon}{4} n^2$ edges. The existence of C is guaranteed because $C_{|I|+1}$, which by definition is equal to I , is an independent set and has no edges. Define α such that $|C| = (\rho - \alpha)n$. The value of α is bounded by $\frac{\epsilon}{2\rho} \leq \alpha \leq \rho$, where the lower bound on α is due to the fact that G is ϵ -far from the ρ -INDEPSET property.

Define t^* to be the largest index for which $|C_{t^*}| \geq \rho n$. First observe that $t^* \leq |I|$ because otherwise C_{t^*} is an independent set, which contradicts the fact that G is ϵ -far from the ρ -INDEPSET property. Hence, by [Lemma 4.2](#) applied to all values of $t = 0, 1, 2, \dots, t^* - 1$,

$$|C_{t^*} \setminus C| \leq \left(1 - \frac{\epsilon}{4\rho\alpha} \right)^{t^*} n$$

and since $|C_{t^*} \setminus C| \geq \alpha n$, we conclude that $t^* \leq \frac{4\rho\alpha}{\epsilon} \ln(1/\alpha) \leq \frac{4\rho\alpha}{\epsilon} \ln(2\rho/\epsilon)$.

Consider now values $t > t^*$. When $G[C_t]$ contains more than $\frac{\epsilon}{4}n^2$ edges, then the vertices in C_t have average degree at least $\frac{\epsilon}{4\rho}n$ in this graph. This means that at least $\frac{\epsilon}{4\rho}n$ vertices in C_t have degree at least $\frac{\epsilon}{4\rho}n$ in $G[C_t]$, so at least $\frac{\epsilon}{4\rho}n$ vertices are removed in the t -th iteration of [Algorithm 2](#). Since $|C_{t^*+1} \setminus C| \leq \alpha n$, there can be at most $\frac{4\rho\alpha}{\epsilon}$ such iterations before we reach C . Hence, in total there are at most $\frac{4\rho\alpha}{\epsilon} \ln(2\rho/\epsilon) + \frac{4\rho\alpha}{\epsilon} \leq \frac{8\rho\alpha}{\epsilon} \ln(2\rho/\epsilon)$ iterations before we reach C , and rearranging for α shows that the container C satisfies the conclusion of the lemma. $\square$

[Lemma 1.4](#), restated below, follows almost immediately from [Lemma 4.3](#).

Lemma 1.4. *Let $G = (V, E)$ be a graph on n vertices that is ϵ -far from the ρ -INDEPSET property. Then there exists a collection $\mathcal{F} \subseteq 2^V$ of fingerprints and a container function $\mathcal{C} : \mathcal{F} \rightarrow 2^V$ that satisfy the following two properties.*

1. *For every independent set I in G of cardinality $|I| \geq \frac{8\rho^2 \ln(\frac{2\rho}{\epsilon})}{\epsilon}$, there is a fingerprint $F \in \mathcal{F}$ such that $F \subseteq I \subseteq \mathcal{C}(F)$.*
2. *For every fingerprint $F \in \mathcal{F}$, the cardinality of F and of its associated container $\mathcal{C}(F)$ are bounded by*

$$|F| \leq \frac{8\rho^2 \ln(\frac{2\rho}{\epsilon})}{\epsilon} \quad \text{and} \quad |\mathcal{C}(F)| \leq \left(\rho - |F| \cdot \frac{\epsilon}{8\rho \ln(\frac{2\rho}{\epsilon})} \right) n.$$

Proof. For any independent set I in G , let t_I denote an index that satisfies the container cardinality bound in [Lemma 4.3](#). Let

$$\mathcal{F} = \left\{ F_{t_I}(I) : I \text{ is an independent set in } G \text{ and } |I| \geq \frac{8\rho^2}{\epsilon} \ln\left(\frac{2\rho}{\epsilon}\right) \right\},$$

and let the container function $\mathcal{C} : \mathcal{F} \rightarrow 2^V$ be the function that takes a fingerprint $F \in \mathcal{F}$ and returns $C_{|F|}(F)$, where $C_t(\cdot)$ and $F_t(\cdot)$ are the functions defined by [Algorithm 2](#).

Before proving the conclusions of the lemma, first observe that for any independent set I with $|I| \geq \frac{8\rho^2}{\epsilon} \ln(\frac{2\rho}{\epsilon})$, we have that $|F_{t_I}(I)| = t_I$. This holds because the upper bound on t_I from [Lemma 4.3](#) and the fact that exactly one new vertex is added to the fingerprint in each step of the procedure. Using this, we have that

$$\mathcal{C}(F_{t_I}(I)) = C_{|F_{t_I}(I)|}(F_{t_I}(I)) = C_{t_I}(F_{t_I}(I)) = C_{t_I}(I),$$

where the third equality uses [Proposition 4.1](#).

Now, to show the first conclusion of the lemma, let I be any independent set. By the basic properties of containers we have that $F_{t_I}(I) \subseteq I \subseteq C_{t_I}(I)$, and since $\mathcal{C}(F_{t_I}(I)) = C_{t_I}(I)$ then $F_{t_I}(I) \subseteq I \subseteq \mathcal{C}(F_{t_I}(I))$. Finally, by the definition of $\mathcal{F}$, we have that $F_{t_I}(I) \in \mathcal{F}$, which completes the proof of the first conclusion of the lemma.

We now show the desired upper bounds on the size of every fingerprint $F \in \mathcal{F}$ and associated container $\mathcal{C}(F)$. Since $|F_{t_I}(I)| = t_I$ then by the upper bound of $t_I \leq \frac{8\rho^2}{\epsilon} \ln\left(\frac{2\rho}{\epsilon}\right)$ from [Lemma 4.3](#) we get the upper bound on $|F_{t_I}(I)|$, and by the upper bound on $|C_{t_I}(I)|$ in [Lemma 4.3](#) we get the desired upper bound on $|\mathcal{C}(F_{t_I}(I))|$. $\square$

Remark 4.4. Since [Lemma 4.3](#) applies for any independent set it is possible to prove a version of [Lemma 1.4](#) so that the first conclusion of the lemma holds for any independent set, instead of just independent sets of size at least $\frac{8\rho^2}{\epsilon} \ln\left(\frac{2\rho}{\epsilon}\right)$. However this requires a slight variation on the definition of a fingerprint or a slight variation of the container procedure, and for [Theorem 1.3](#) we only need that [Lemma 1.4](#) applies for independent sets of size at least roughly $\frac{\rho^2 \ln^3(1/\epsilon)}{\epsilon}$. For example, one such variation is including the step t as part of the fingerprint, so that now the fingerprint is a tuple containing a set of vertices and a stopping point t for the procedure.

4.5 Proof of [Theorem 1.3](#)

We are now ready to use [Lemma 1.4](#) to complete the proof of [Theorem 1.3](#), restated below.

Theorem 1.3. *The sample complexity of ϵ -testing the ρ -INDEPSET property is $\tilde{O}\left(\frac{\rho^3}{\epsilon^2}\right)$.*

Proof. Let S be a random set of $s = c \cdot \frac{\rho^3}{\epsilon^2} \ln^3\left(\frac{1}{\epsilon}\right)$ vertices drawn uniformly at random from V without replacement, where c is a large enough constant. Note that for clarity of presentation, in the rest of the proof we ignore all integer rounding issues as they do not affect the asymptotics of the final result.

If G contains a ρn independent set, then S contains at least ρs vertices from this independent set with probability at least $\frac{1}{2}$, since the number of such vertices follows a hypergeometric distribution, and the median of this distribution is at least ρs [[Neu70](#), [KB80](#)].

In the remainder of the proof we upper bound the probability that S contains a ρs -independent set when G is ϵ -far from containing a ρn -independent set. We consider the

collection of fingerprints $\mathcal{F}$ and the container function $\mathcal{C} : \mathcal{F} \rightarrow 2^V$ from [Lemma 1.4](#) which states that for every independent set I in G with $|I| > \frac{8\rho^2}{\epsilon} \ln\left(\frac{2\rho}{\epsilon}\right)$ there exists $F \in \mathcal{F}$ such that $F \subseteq I \subseteq \mathcal{C}(F)$. Hence, S contains an independent set of size $\rho s > \frac{8\rho^2}{\epsilon} \ln\left(\frac{2\rho}{\epsilon}\right)$ only when there exists $F \in \mathcal{F}$ such that S contains F **and** S contains at least ρs vertices from $\mathcal{C}(F)$. In other words, the probability that S contains a ρs independent set is at most

$$\sum_{F \in \mathcal{F}} \Pr[F \subseteq S] \cdot \Pr[|\mathcal{C}(F) \cap S| \geq \rho s \mid F \subseteq S].$$

For a specific $F \in \mathcal{F}$, the probability that S contains F is at most

$$\frac{\binom{n-|F|}{s-|F|}}{\binom{n}{s}} = \frac{\binom{s}{|F|}}{\binom{n}{|F|}},$$

because there are at most $\binom{n}{s}$ ways to choose S , and $\binom{n-|F|}{s-|F|}$ ways to choose an S that includes F . The probability that S contains at least ρs vertices from $\mathcal{C}(F)$, conditioned on the fact that S contains F , can be upper bounded using Chernoff's bound as explained next.

Fix any fingerprint F and set $t = |F|$. Let S be a sample of s vertices that contains F . Let X denote the number of vertices among the other $s - t$ sampled vertices that belong to $\mathcal{C}(F)$. By the upper bound on the cardinality of $\mathcal{C}(F)$ from the second property of [Lemma 1.4](#), the expected value of X is bounded above by

$$\mathbb{E}[X] \leq \left(\rho - \frac{t\epsilon}{8\rho \ln(\frac{2\rho}{\epsilon})} \right) (s - t) < \rho s - \frac{t\epsilon s}{8\rho \ln(\frac{2\rho}{\epsilon})} \leq \rho s - t - \frac{t\epsilon s}{16\rho \ln(\frac{2\rho}{\epsilon})},$$

where the last inequality uses the fact that $s = c \cdot \frac{\rho^3}{\epsilon^2} \ln^3(\frac{1}{\epsilon})$ for a large enough constant c and that the problem is non-trivial only when $\epsilon < \rho^2$.

By Chernoff's bound in [Lemma 2.2](#), the probability that we draw at least $\rho s - t$ vertices from $\mathcal{C}(F)$ in the additional $s - t$ vertices drawn to form S is

$$\begin{aligned} \Pr[X \geq \rho s - t] &\leq \exp\left(\frac{-(\rho s - t - \mathbb{E}[X])^2}{\rho s - t + \mathbb{E}[X]}\right) \\ &\leq \exp\left(\frac{-\left(\frac{t\epsilon s}{16\rho \ln(\frac{2\rho}{\epsilon})}\right)^2}{2\rho s}\right) \\ &\leq \exp\left(-\frac{t^2 \epsilon^2 s}{512\rho^3 \ln^2(\frac{2\rho}{\epsilon})}\right). \end{aligned}$$

Therefore, by applying a union bound over all values $t \leq \frac{8\rho^2 \ln(\frac{2\rho}{\epsilon})}{\epsilon}$ and all possible choices of $F \in \mathcal{F}$ of size t , the probability that S contains an independent set of size ρs is at most

$$\begin{aligned} \sum_t \sum_{\substack{F \in \mathcal{F} \\ |F|=t}} \frac{\binom{s}{|F|}}{\binom{n}{|F|}} \cdot \Pr[|\mathcal{C}(F) \cap S| \geq \rho s \mid F \subseteq S] &\leq \sum_t \binom{s}{t} \exp\left(-\frac{t^2 \epsilon^2 s}{512 \rho^3 \ln^2(\frac{2\rho}{\epsilon})}\right) \\ &\leq \sum_t \exp\left(t \ln s - \frac{t^2 \epsilon^2 s}{512 \rho^3 \ln^2(\frac{2\rho}{\epsilon})}\right) < \frac{1}{3} \end{aligned}$$

where the second inequality uses the upper bound $\binom{s}{t} \leq s^t$, and the last inequality again uses the fact that $s = c \cdot \frac{\rho^3}{\epsilon^2} \ln^3(\frac{1}{\epsilon})$ for a large enough constant c . Hence, the probability that S contains an independent set of size at least ρs is less than $1/3$. $\square$

Remark 4.5. The completeness guarantee in the proof of [Theorem 1.3](#) states that the algorithm correctly accepts graphs that have a ρn -independent set with probability at least $\frac{1}{2}$, instead of the usual $\frac{2}{3}$ bound typically used in property testing. As discussed in [Chapter 2](#), since the soundness guarantee is that the algorithm accepts graphs that are ϵ -far from the ρ -INDEPSET property with probability at most $\frac{1}{3}$, these guarantees still enable us to apply standard error reduction techniques to obtain algorithms that err with probability at most $\frac{1}{3}$ with the same asymptotic sample complexity.

Alternatively, it is also possible to obtain the $\frac{2}{3}$ completeness guarantee directly by changing the algorithm slightly to check if the induced subgraph has the $(\rho - \tau)$ -INDEPSET property for some appropriate gap parameter $\tau := \tau(\rho, \epsilon)$.

Chapter 5

Testing Satisfiability

In this chapter we prove [Theorem 1.5](#), restated below.

Theorem 1.5. *The sample complexity of ϵ -testing the (q, k) -SAT property is $\tilde{O}(\frac{kq^3}{\epsilon})$.*

We note that throughout this section we assume that $q = o(n)$. We can do this because the bound on the sample complexity in [Theorem 1.5](#) is trivial when $q = \Omega(n)$.

5.1 Proof Overview

We are given a CSP ϕ , with n variables, exactly q variables per constraint and an alphabet of size k , and wish to distinguish between the case that ϕ is satisfiable and the case that ϕ is ϵ -far from satisfiable. We analyze the canonical tester that takes a sample S of variables selected uniformly at random, and checks whether there exists an assignment to the variables in S that satisfies $\phi[S]$, where $\phi[S]$ is the set of constraints whose variables are all present in S . If ϕ is satisfiable, then $\phi[S]$ is certainly satisfiable. The main challenge of the proof is showing that if ϕ is ϵ -far from satisfiable, then $\phi[S]$ is satisfiable with only low probability.

We wish to apply the container method to handle this challenge. In the case of satisfiability testing, what is sampled by the canonical tester is a set of *variables* of the input CSP (as opposed to a set of vertices as seen in [Theorem 1.3](#)), and it accepts if and only if the CSP restricted to the sampled variables is satisfiable. As a result, instead of a container lemma for independent sets in a graph that is ϵ -far from having a large independent set

(like in [Lemma 1.4](#)), we desire a container lemma for the sets of variables S for which $\phi[S]$ is satisfiable in a CSP ϕ that is ϵ -far from satisfiable. Such a container lemma is useful because it gives a set of containers for the sets of variables S for which $\phi[S]$ is satisfiable, and, when ϕ is ϵ -far from satisfiable, this collection of containers can be used to upper bound the probability that the tester wrongly accepts. We prove such a container lemma, restated formally below. Our new container lemma follows a very similar structure to [Lemma 1.4](#), with “independent sets” replaced by sets of variables S for which the restriction of the CSP $\phi[S]$ is satisfiable.

Note that in the container lemma below each fingerprint $F \in \mathcal{F}$ is a subset of the set $k \times X$. In other words, a fingerprint can be viewed as a set of single variable assignments. We use the notation $\text{vars}(F)$ to denote the set of variables appearing in F . In particular, if $|\text{vars}(F)| = |F|$ then F corresponds to a set of single variable assignments such that each variable is assigned at most once.

Lemma 5.1. *Let ϕ be a CSP over variables $X = \{x_1, \dots, x_n\}$ that is ϵ -far from (q, k) -SAT. Then there exists a collection $\mathcal{F} \subseteq 2^{kX}$ of fingerprints and a container function $\mathcal{C} : \mathcal{F} \rightarrow 2^X$ that satisfy the following two properties.*

1. *For every $S \subseteq X$ for which $|S| \geq \frac{8kq^2}{\epsilon}$ and $\phi[S]$ is satisfiable, there is a fingerprint $F \in \mathcal{F}$ such that $\text{vars}(F) \subseteq S \subseteq \mathcal{C}(F) \cup \text{vars}(F)$.*
2. *For every fingerprint $F \in \mathcal{F}$, $|\text{vars}(F)| = |F|$ and the cardinality of F and of its associated container $\mathcal{C}(F)$ are bounded by*

$$|F| \leq 8 \frac{kq^2}{\epsilon} \quad \text{and} \quad |\mathcal{C}(F)| \leq \left(1 - |F| \cdot \frac{\epsilon}{4kq^3 \ln \frac{kq}{\epsilon}} \right) n.$$

In order to prove the container lemma we first observe that the problem of satisfiability can be expressed as a problem about independent sets in hypergraphs. For any CSP ϕ , we construct its associated hypergraph H_ϕ in the following standard way. The hypergraph H_ϕ has nk vertices. We associate each vertex with a possible assignment to a variable of ϕ . (So each variable of ϕ is associated with k vertices in H_ϕ .) We then add a q -edge to the hypergraph H_ϕ for each assignment to q -variables that falsifies a constraint in ϕ . See [Fig. 5.1](#) for an example of an edge coming from the constraint that evaluates to false only when the variables x_1, x_2, x_3 are all equal. With this construction, a satisfying assignment to ϕ corresponds to an independent set on n vertices in H_ϕ where each vertex in this independent set corresponds to a unique variable. And when ϕ is ϵ -far from satisfiable,

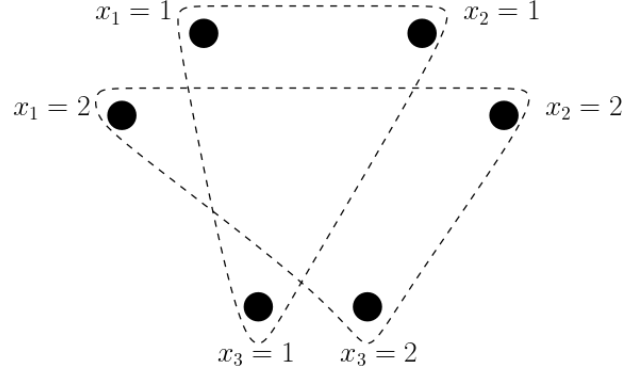


Figure 5.1: An illustration of the two hyperedges formed from the constraint that the variables x_1 , x_2 , and x_3 must not all be equal.

every induced subhypergraph of H_ϕ that corresponds to an assignment to all n variables contains at least $\epsilon \binom{n}{q}$ edges.

With this construction, a collection of containers for the independent sets in H_ϕ gives a collection of containers for the subsets $S \subseteq X$ of variables for which $\phi[S]$ is satisfiable. Hence, an initial approach to prove [Lemma 5.1](#) is to apply a standard hypergraph container lemma (such as a hypergraph container lemma seen in [\[BMS15\]](#) or [\[ST15\]](#)) to find containers for the independent sets in H_ϕ . However, when converting these containers for independent sets to containers for subsets $S \subseteq X$ of variables we do not get the desired upper bounds on the size of each container. Instead, we design a new container procedure that uses the fact that the hypergraphs constructed from CSPs are not arbitrary, but rather have a specific structure. In particular, if ϕ is ϵ -far from satisfiable, then all n -subgraphs in H_ϕ that correspond to assignments to n different variables have at least $\epsilon \binom{n}{q}$ edges. This specific characteristic of H_ϕ , which itself is very different than any characteristics utilized by hypergraph container methods in the past, can be exploited when constructing the container/fingerprint pairs.

Further, as was the case with [Lemma 1.4](#), a main challenge is showing that there is a trade-off between the size of fingerprints and their corresponding containers. This trade-off is what allows us to prove our strong upper bounds on the analysis of the canonical tester.

5.2 Container Procedure

We now describe our hypergraph container procedure ([Algorithm 3](#)). The container procedure takes as input a q -uniform hypergraph $H = (V, E)$, a parameter $n < |V|$, and an independent set $I \subseteq V$, and produces a sequence of fingerprints and containers. The procedure constructs this sequence iteratively. After setting the initial container C_0 to be V and the initial fingerprint F_0 to be empty, the procedure runs for $|I|/(q-1)$ iterations,¹ where in each iteration the procedure selects $(q-1)$ distinct fingerprint vertices from the independent set and reduces the size of the container accordingly. These fingerprint vertices are selected in a greedy fashion to try and remove as many vertices from the container as possible.

The basic idea for selecting these fingerprint vertices and updating the containers is as follows. Suppose the current container is C , and suppose v is the vertex contained in I with highest degree in $H[C]$. First observe that all vertices with degree higher than v in $H[C]$ are certainly not in the independent set, and so they are removed from the container. Further, every q -edge in $H[C]$ can be viewed as a set of q vertices that cannot all be contained in the independent set. By constructing a $(q-1)$ -uniform hypergraph with vertex set $C \setminus v$ and with an edge set that is all formed by removing v from all q -edges containing v , we have that each $(q-1)$ -edge in this new hypergraph can be viewed as $q-1$ vertices that cannot all be in the independent set. And so we recurse on the new $(q-1)$ -uniform hypergraph.

In order to optimize the procedure for hypergraphs constructed from CSPs, we modify this basic approach as follows. Since our procedure will run on a hypergraph H_ϕ constructed from a CSP ϕ on n variables that is ϵ -far from satisfiable, and this hypergraph has the important property that every n -subgraph corresponding to unique variables is dense (see [Section 5.4](#) for specific details), it is useful to consider the degree of a vertex in induced subgraphs on at most n vertices.

In more detail, the first fingerprint vertex selected in each iteration will be the vertex which achieves the highest degree in some induced subgraph on at most n vertices, which we call a $(\leq n)$ -subgraph, of the current container. For example, if the current container is $C \subseteq V(H)$, then for each vertex v we calculate the quantity

$$\deg_{H[C]}^{\leq n}(v) := \max_{D \in \binom{C}{\leq n}} (\deg_{H[D]}(v))$$

and select the fingerprint vertex to be the vertex of I achieving the highest value, where ties are broken based on an assumed ordering of the vertices $V[H]$.

¹Note that for clarity of presentation, here and throughout we ignore all integer rounding issues as they do not affect the asymptotics of the final result.

After selecting the fingerprint vertex $v_{t,q}$, where t denotes that we are in the t -th iteration and the q denotes the fact that the vertex was selected from a q -uniform hypergraph, the rest of the iteration consists of running an inner loop that begins with the subgraph that achieves this maximum. In particular, construct a $(q-1)$ -uniform hypergraph $\mathcal{D}_{t,q-1}$ with vertex set

$$V(\mathcal{D}_{t,q-1}) = \arg \max_{D \in \binom{C}{\leq n}} (\deg_{H[D]}(v_{t,q})) \setminus v_{t,q},$$

where if there are multiple D sets reaching the maximum then an arbitrary maximal set is selected. The edge set of $\mathcal{D}_{t,q-1}$ is constructed by removing $v_{t,q}$ from all q -edges of $H[C]$ that contain $v_{t,q}$. The edge set corresponds to sets of vertices that cannot all be part of the independent set.

The inner loop follows the basic approach described above. In particular, the loop runs from $\ell = q-1, \dots, 2$, and in each step it selects the vertex $v_{t,\ell}$ in I with highest degree in the current hypergraph $\mathcal{D}_{t,\ell}$ and adds it to the fingerprint.² It then constructs a new $(\ell-1)$ -uniform hypergraph $\mathcal{D}_{t,\ell-1}$ that has edge set consisting of all edges of $\mathcal{D}_{t,\ell}$ containing $v_{t,\ell}$, with $v_{t,\ell}$ removed. In each step, we also keep track of the vertices with higher degree than the vertex selected, and these vertices are removed after the inner loop finishes because they cannot be part of the independent set.

Finally, after the inner loop is completed, we are left with a 1-uniform hypergraph $\mathcal{D}_{t,1}$. Since, in each step, the edge set corresponds to sets of vertices that cannot all be part of the independent set, we have that all the 1-edges in $\mathcal{D}_{t,1}$ correspond to vertices that cannot be part of the independent set. So these vertices are also removed. This completes one iteration, and this is repeated $|I|/(q-1)$ times until all vertices of I have been selected.

The procedure outputs the sequence of fingerprints $F_1, F_2, \dots, F_{|I|/(q-1)}$ and their corresponding containers $C_1, C_2, \dots, C_{|I|/(q-1)}$ generated by the algorithm for the independent set I . We refer to F_t and C_t as the t -th *fingerprint* and t -th *container* of I , respectively. When we consider the fingerprints or containers of multiple independent sets, we write $F_t(I)$ and $C_t(I)$ to denote the t -th fingerprint and container of the independent set I . (When the current context specifies a single independent set, we will continue to write only F_t and C_t to keep the notation lighter.) We note that it is also convenient to extend the definition with $C_t = F_t = I$ for all $t > |I|/(q-1)$.

²Note that I may not be a subset of $V(\mathcal{D}_{t,\ell})$, and so we define $\deg_{\mathcal{D}_{t,\ell}}(v)$ for a vertex $v \notin \mathcal{D}_{t,\ell}$ as zero.

Algorithm 3: FINGERPRINT & CONTAINER GENERATOR

Input: A q -uniform hypergraph $H = (V, E)$, a parameter $n < |V|$, and an independent set $I \subseteq V$

```

1 Initialize  $F_0 \leftarrow \emptyset$  and  $C_0 \leftarrow V(H)$ ;
2 for  $t = 1, 2, \dots, |I|/(q-1)$  do
3    $v_{t,q} \leftarrow$  the vertex in  $I \cap C_{t-1}$  that has the largest value of  $\deg_{H[C_{t-1}]}^{\leq n}(v)$ ;
   // Keep track of all vertices with higher  $q$ -degree than  $v_{t,q}$  to be
   removed later
4    $X_q \leftarrow \{w \in C_{t-1} : \deg_{H[C_{t-1}]}^{\leq n}(w) > \deg_{H[C_{t-1}]}^{\leq n}(v_{t,q})\}$ ;
   // Construct  $(q-1)$ -uniform hypergraph  $\mathcal{D}_{t,q-1}$  on the set vertices
   of  $C_{t-1}$ , of size at most  $n$ , that  $v_{t,q}$  has highest degree in. The
   edge set of  $\mathcal{D}_{t,q-1}$  is all edges incident to  $v_{t,q}$ , with  $v_{t,q}$  removed
5    $V(\mathcal{D}_{t,q-1}) \leftarrow \left( \arg \max_{D \in \binom{C_{t-1}}{\leq n}} \deg_{H[D]}(v_{t,q}) \right) \setminus v_{t,q}$ ;
6    $E(\mathcal{D}_{t,q-1}) \leftarrow \{e \setminus v_{t,q} : e \in E(H[C_{t-1}]) \text{ and } v_{t,q} \in e\}$ ;
7   for  $\ell = q-1, \dots, 2$  do
8      $v_{t,\ell} \leftarrow$  the vertex in  $I \setminus (F_{t-1} \cup \{v_{t,\ell+1}, \dots, v_{t,q-1}\})$  with largest  $\ell$ -degree in
        $\mathcal{D}_{t,\ell}$ ;
     // Keep track of all vertices with higher  $\ell$ -degree than  $v_{t,\ell}$  in
      $\mathcal{D}_t$  to remove later
9      $X_\ell \leftarrow \{w \in V(\mathcal{D}_{t,\ell}) : \deg_{\mathcal{D}_{t,\ell}}(w) > \deg_{\mathcal{D}_{t,\ell}}(v_{t,\ell})\}$ ;
     // Construct  $\mathcal{D}_{t,\ell-1}$  on the vertices of  $\mathcal{D}_{t,\ell}$  minus  $v_{t,\ell}$ , and
     containing all edges incident to  $v_{t,\ell}$ , with  $v_{t,\ell}$  removed
10     $V(\mathcal{D}_{t,\ell-1}) \leftarrow V(\mathcal{D}_{t,\ell}) \setminus v_{t,\ell}$ ;
11     $E(\mathcal{D}_{t,\ell-1}) \leftarrow \{e \setminus v_{t,\ell} : e \in E(\mathcal{D}_{t,\ell}) \text{ and } v_{t,\ell} \in e\}$ ;
    // Add  $v_{t,2}, \dots, v_{t,q}$  to the fingerprint
12     $F_t \leftarrow F_{t-1} \cup \{v_{t,2}, \dots, v_{t,q}\}$ ;
    // Update container set by removing all  $X_\ell$  sets, all vertices
    with a 1-edge, and the new fingerprint vertices
13     $C_t \leftarrow C_{t-1} \setminus (\cup_{\ell=2}^q X_\ell \cup \{v \in C_{t-1} : \{v\} \in E(\mathcal{D}_{t,1})\} \cup \{v_{t,2}, \dots, v_{t,q}\})$ ;
14 Return  $F_1, \dots, F_{|I|/q}$  and  $C_1, \dots, C_{|I|/q}$ ;

```

5.3 Basic Properties

By construction, the sequence of fingerprints and containers of an independent set I satisfy a number of useful properties. For instance, we have $F_1 \subseteq F_2 \subseteq \dots \subseteq F_{|I|/(q-1)} \subseteq I$ and $I \setminus F_{|I|/(q-1)} \subseteq C_{|I|/(q-1)} \subseteq C_{|I|/(q-1)-1} \subseteq \dots \subseteq C_2 \subseteq C_1$, so that for each t , $F_t \subseteq I$ and $I \setminus F_t \subseteq C_t$.

The construction of the algorithm also guarantees that the t -th container of an independent set is the same as the t -th container of its t -th fingerprint. In other words, even for a large independent set, the t -th container is uniquely defined by the t -th fingerprint of size $t \cdot (q - 1)$, which may be much smaller than the size of the independent set. This can be used to show an upper bound on the number of possible fingerprints/containers, which will be useful later in the proof of [Theorem 1.5](#).

Proposition 5.2. *For any hypergraph $H = (V, E)$, any independent set I in H , and any t , the fingerprint $F_t(I)$ and container $C_t(I)$ of I satisfy*

$$C_t(F_t(I)) = C_t(I).$$

Proof. If $t > |I|/(q - 1)$ then $F_t(I) = I = C_t(I)$ and the equality holds. Now consider the case that $t \leq |I|/(q - 1)$. If $v_{1,q}, \dots, v_{1,2}, v_{2,q}, \dots, v_{2,2}, \dots, v_{t,q}, \dots, v_{t,2}$ are the vertices selected in the first t iterations of the container algorithm on input I , then the algorithm selects the same vertices and forms the same container C_t when provided with input $F_t(I)$ instead of I . $\square$

Next, we demonstrate that the container procedure makes progress in every step of the procedure by finding sparser and sparser containers. In order to show this, we first require a basic property about the number of vertices with nearly average degree in a hypergraph. The average degree in a ℓ -uniform hypergraph on $|E|$ edges is $\ell|E|/|V|$. In the following proposition, we show that if $|E|$ is sufficiently large, then a large number of vertices have degree at least $(\ell - 1)|E|/|V|$.

Proposition 5.3. *Let $H = (V, E)$ be a ℓ -uniform hypergraph, with $\ell \geq 2$. Then there are at least $|E|/\binom{|V|-1}{\ell-1}$ vertices in H with degree larger than $(\ell - 1)|E|/|V|$.*

Proof. Suppose not. In other words, suppose that there are strictly less than $|E|/\binom{|V|-1}{\ell-1}$ vertices with degree larger than $(\ell - 1)|E|/|V|$. These vertices have degree at most $\binom{|V|-1}{\ell-1}$. The remaining vertices, of which there are at most $|V|$, have degree at most $(\ell - 1)|E|/|V|$.

So the total number of edges is less than $\frac{1}{\ell} \left(\frac{|E|}{\binom{|V|-1}{\ell-1}} \cdot \binom{|V|-1}{\ell-1} + |V| \cdot \frac{(\ell-1)|E|}{|V|} \right) = |E|$, a contradiction. $\square$

We can now state and prove the property that demonstrates that the container procedure makes progress in every iteration. In particular, we show that as t increases, we get a stronger upper bound on the maximum degree of every $(\leq n)$ -subgraph of $H[C_t]$.

Proposition 5.4. *Let $H = (V, E)$ be a q -uniform hypergraph with $|V| = kn$, let I be an independent set in H , and let $t \leq |I|/(q-1)$. Then the maximum q -degree of every $(\leq n)$ -subgraph of $H[C_t]$ is at most $\frac{2kq}{t} \binom{n-1}{q-1}$.*

Proof. Fix any $t \leq |I|/(q-1)$. Since the containers satisfy the containment $V = C_0 \supseteq C_1 \supseteq \dots \supseteq C_t$, then

$$kn \geq |C_0| - |C_t| = \sum_{i=1}^t (|C_{i-1}| - |C_i|) = \sum_{i=1}^t |C_{i-1} \setminus C_i|$$

and there must be an index $j \in [t]$ for which $|C_{j-1} \setminus C_j| \leq kn/t$.

We claim that for each step of the inner loop of the j th iteration, the number of ℓ -edges in $\mathcal{D}_{j,\ell}$ is at most $\frac{2k\ell}{t} \binom{n-(q-\ell)}{\ell}$. First we show that this claim suffices to prove the proposition. Using the claim, the number of $(q-1)$ -edges in $\mathcal{D}_{j,q-1}$ is at most $\frac{2k(q-1)}{t} \binom{n-1}{q-1}$, and so the fingerprint vertex $v_{j,q}$ has q -degree at most $\frac{2k(q-1)}{t} \binom{n-1}{q-1}$ in every induced $(\leq n)$ -subgraph of $H[C_{j-1}]$. Hence, all vertices that had higher degree in some $(\leq n)$ -subgraph of $H[C_{j-1}]$ were excluded from C_j , so the maximum degree in any $(\leq n)$ -subgraph of $H[C_j]$ is at most $\frac{2k(q-1)}{t} \binom{n-1}{q-1}$. Since $C_t \subseteq C_j$, the maximum degree of any $(\leq n)$ -subgraph of $H[C_t]$ can also be upper bounded by $\frac{2k(q-1)}{t} \binom{n-1}{q-1}$.

We now proceed to prove the claim by induction. The base case is the largest ℓ such that $|E(\mathcal{D}_{j,\ell})| = 0$, and if no such ℓ exists then $\ell = 1$ is the base case.³ The claim holds trivially if $|E(\mathcal{D}_{j,\ell})| = 0$. Otherwise, in the $\ell = 1$ case, the vertices corresponding to 1-edges in $\mathcal{D}_{j,1}$ are removed from C_j . Since $|C_{j-1} \setminus C_j| \leq \frac{kn}{t}$ then the number of 1-edges is at most $\frac{kn}{t}$, which is at most $\frac{2k(n-q+1)}{t}$ since $q = o(n)$.

³We require $|E(\mathcal{D}_{j,\ell})| = 0$ as a base case to handle the case that a vertex outside of $\mathcal{D}_{j,\ell}$ is added to the fingerprint, because when that happens $|V[\mathcal{D}_{j,\ell}]|$ does not shrink by one, and so the proof in the inductive step does apply as is. If such a vertex is selected then $|E(\mathcal{D}_{j,\ell})|$ becomes 0.

In the inductive case $\ell \leq q - 1$, suppose that the number of ℓ -edges in $\mathcal{D}_{j,\ell}$ is more than $\frac{2k\ell}{t} \binom{n-(q-\ell)}{\ell}$. If $v_{j,\ell}$ has degree larger than $\frac{2k(\ell-1)}{t} \binom{n-(q-\ell)-1}{\ell-1}$, then, by the construction of $\mathcal{D}_{j,\ell-1}$, the number of $(\ell-1)$ -edges in $\mathcal{D}_{j,\ell-1}$ is larger than $\frac{2k(\ell-1)}{t} \binom{n-(q-\ell)-1}{\ell-1}$, a contradiction with the inductive hypothesis. Otherwise, by [Proposition 5.3](#), there are at least

$$\frac{|E(\mathcal{D}_{j,\ell})|}{\binom{|V(\mathcal{D}_{j,\ell})|-1}{\ell-1}} \geq \frac{\frac{2k\ell}{t} \binom{n-(q-\ell)}{\ell}}{\binom{n-(q-\ell)-1}{\ell-1}} = \frac{2k(n-(q-\ell))}{t} > \frac{kn}{t}$$

vertices with degree larger than $\frac{(\ell-1)|E(\mathcal{D}_{j,\ell})|}{|V(\mathcal{D}_{j,\ell})|} \geq \frac{2(\ell-1)k\ell \binom{n-(q-\ell)}{\ell}}{t(n-(q-\ell))} = \frac{2k(\ell-1)}{t} \binom{n-(q-\ell)-1}{\ell-1}$ in $\mathcal{D}_{j,\ell}$, where we use that $|V(\mathcal{D}_{j,\ell})| \leq n - (q - \ell)$, which follows from the fact that the fingerprint vertex is removed from $V(\mathcal{D}_{j,\ell})$ in each step, in both inequalities and that $q = o(n)$. These high degree vertices are all removed from C_j because they have higher degree than $v_{j,\ell}$. This results in $|C_{j-1} \setminus C_j| > \frac{kn}{t}$, a contradiction, which completes the proof of the claim. $\square$

5.4 Hypergraph Container Lemma

Our results on testing satisfiability make use of a standard transformation of CSPs to hypergraphs that we define here.

Definition 5.5. Let ϕ be a q -uniform CSP on n variables that each can take k possible values. The *hypergraph representation* of ϕ , denoted H_ϕ , is defined to be a q -uniform hypergraph on $k \cdot n$ vertices, where each vertex corresponds to a single variable assignment. For each constraint of ϕ , we add one edge for each assignment to the q variables that falsifies the constraint.

See [Figure 5.1](#) for an example of an edge coming from the constraint that evaluates to false only when the variables x_1, x_2, x_3 are all equal. It's worth noting that edges are not added to sets of vertices that correspond to assignments to repeated variables. Further, without loss of generality we may assume that for any set of q variables there is at most one constraint on those variables in ϕ , and so H_ϕ does not have multi-edges. Throughout this section we utilize the fact that each vertex corresponds to a specific single variable assignment and each q -edge can be viewed as an assignment to q variables.

Using this construction, observe that for any subset $U \subseteq V(H_\phi)$ that only contains vertices corresponding to distinct variables, the edges of $H_\phi[U]$ correspond to distinct falsified constraints. In other words, we have the following property.

Observation 1. *Let $U \subseteq V$ be a subset of vertices of H_ϕ such that the labels of vertices in U are to distinct variables and let A be the $|U|$ -variable assignment corresponding to the labels of U . Then the number of edges in $H_\phi[U]$ is exactly the number of constraints, on variables of U , that are falsified by A .*

In particular, we have that ϕ is satisfiable if and only if there exists an independent set I in H_ϕ with $|I| = n$ and where the vertices of I correspond to distinct variables. Further, if ϕ is ϵ -far from satisfiable, then every induced subgraph of H_ϕ that corresponds to an assignment to all n variables contains at least $\epsilon \binom{n}{q}$ edges.

Using this construction, we first prove the following alternate formulation of our container lemma, which we will then use to prove [Lemma 5.1](#).

Lemma 5.6. *Let ϕ be an instance of (q, k) -SAT on n variables. Let H_ϕ be the hypergraph representation of ϕ , and let I be an independent set in H_ϕ . If ϕ is ϵ -far from satisfiable then there exists $t \leq 8kq/\epsilon$ such that the number of distinct variables appearing in labels of $C_t(I)$, denoted $\text{vars}(C_t(I))$, is bounded by*

$$\text{vars}(C_t(I)) \leq \left(1 - \frac{\epsilon t}{4kq^2 \ln(kq/\epsilon)}\right) n.$$

Proof. We prove the contrapositive. That is, assume that for all $t \leq 8kq/\epsilon$, the number of distinct variables in labels of $C_t(I)$ is bounded by

$$\text{vars}(C_t(I)) > \left(1 - \frac{\epsilon t}{4kq^2 \ln(kq/\epsilon)}\right) n. \tag{5.1}$$

We will construct an assignment to the n variables of ϕ which shows that ϕ is not ϵ -far from satisfiable.

Start with an assignment to as many distinct variables as possible coming from the labels of $C_{8kq/\epsilon}(I)$. In other words, let Z be the set of variables appearing in labels of $C_{8kq/\epsilon}(I)$, and construct $U \subseteq C_{8kq/\epsilon}(I)$ by removing vertices from $C_{8kq/\epsilon}(I)$ until each variable in Z appears in exactly one label in U .

For each $t = 1, \dots, 8kq/\epsilon$, let Z_t be the set of variables that appear in at least one label of a vertex in C_{t-1} , but do not appear in any labels of vertices in C_t . Further, for each $t = 1, \dots, 8kq/\epsilon$, construct $V_t \subset V$ as follows. For each variable $z \in Z_t$, add a single vertex from C_{t-1} to V_t that is labelled with an assignment of z . Note that if a variable in Z_t appears in the labels of multiple vertices C_{t-1} , then we select one of the vertices arbitrarily and add it to V_t .

First observe that $Z \cup Z_1 \cup \dots \cup Z_{8kq/\epsilon}$ form a partitioning of the n variables of ϕ . Hence, $U \cup V_1 \cup \dots \cup V_{8kq/\epsilon}$ contains n distinct vertices, with the labels corresponding to a complete assignment A to the n variables of ϕ .

We claim that $H_\phi[U \cup V_1 \cup \dots \cup V_{8kq/\epsilon}]$ contains less than $\epsilon \binom{n}{q}$ edges. Before proving the claim, observe that this is sufficient for completing the proof of the lemma because, by [Observation 1](#), the number of constraints falsified by A is less than $\epsilon \binom{n}{q}$. Since A is an assignment to the n variables of ϕ , then ϕ is not ϵ -far from satisfiable.

We now proceed to prove the claim. Since $U \subseteq C_{8kq/\epsilon}(I)$ and U is of size at most n , then, by [Proposition 5.4](#), the maximum degree in the subgraph $H[U]$ is at most $\frac{2kq \binom{n-1}{q-1}}{8kq/\epsilon} = \epsilon \binom{n-1}{q-1}/4$, and so $H[U]$ has at most $\frac{\epsilon \binom{n}{q}}{4}$ q -edges. In order to count the additional edges in $H_\phi[U \cup V_1 \cup \dots \cup V_{8kq/\epsilon}]$, consider adding the vertices from $V_1, \dots, V_{8kq/\epsilon}$ to U in reverse order (starting at vertices in $V_{8kq/\epsilon}$ and going to V_1). In this way, for any $t \leq 8kq/\epsilon$, when V_t is added, it holds that $(U \cup V_{8kq/\epsilon} \cup \dots \cup V_t) \subseteq C_{t-1}(I)$. Hence, by [Proposition 5.4](#), each $v \in V_t$ contributes at most $\frac{2kq \binom{n-1}{q-1}}{t-1}$ additional edges (or $\binom{n-1}{q-1}$ edges if $t = 1$) because v is contained in $C_{t-1}(I)$.

Then, the sum of the edges in $H_\phi[U \cup V_1 \cup \dots \cup V_{8kq/\epsilon}]$ can be upper bounded by

$$\frac{\epsilon \binom{n}{q}}{4} + |V_1| \cdot \binom{n-1}{q-1} + \sum_{t=2}^{8kq/\epsilon} |V_t| \frac{2kq \binom{n-1}{q-1}}{t-1}. \quad (5.2)$$

For any $t \leq 8kq/\epsilon$, $\bigcup_{j=1}^t Z_j$ is the set of variables not appearing in labels of $C_t(I)$, and so [\(5.1\)](#) implies that $\sum_{j=1}^t |Z_j| \leq \frac{t\epsilon n}{4kq^2 \ln(kq/\epsilon)}$. Since $|Z_j| = |V_j|$ for all j , then $\sum_{j=1}^t |V_j| \leq \frac{t\epsilon n}{4kq^2 \ln(kq/\epsilon)}$. In the sum [\(5.2\)](#), the contribution from $|V_t|$ goes down as t increases, and so the sum is maximized when $|V_t| = \frac{\epsilon n}{4kq^2 \ln(kq/\epsilon)}$ for all t . Hence, the sum of the edges in $H_\phi[U \cup V_1 \cup \dots \cup V_{8kq/\epsilon}]$ can be upper bounded by

$$\frac{\epsilon \binom{n}{q}}{4} + \frac{\epsilon \binom{n}{q}}{4kq \ln(kq/\epsilon)} + \frac{\epsilon \binom{n}{q}}{2 \ln(kq/\epsilon)} \sum_{t=2}^{8kq/\epsilon} \frac{1}{t-1} < \epsilon \binom{n}{q},$$

where the last inequality uses the upper bound $H(m) \leq \ln(m) + 1$ on the harmonic series and the bound $\epsilon < e^{-8}$ that we can apply without loss of generality. $\square$

We now prove our container lemma [Lemma 5.1](#), restated below.

Lemma 5.1. *Let ϕ be a CSP over variables $X = \{x_1, \dots, x_n\}$ that is ϵ -far from (q, k) -SAT. Then there exists a collection $\mathcal{F} \subseteq 2^{kX}$ of fingerprints and a container function $\mathcal{C} : \mathcal{F} \rightarrow 2^X$ that satisfy the following two properties.*

1. *For every $S \subseteq X$ for which $|S| \geq \frac{8kq^2}{\epsilon}$ and $\phi[S]$ is satisfiable, there is a fingerprint $F \in \mathcal{F}$ such that $\text{vars}(F) \subseteq S \subseteq \mathcal{C}(F) \cup \text{vars}(F)$.*
2. *For every fingerprint $F \in \mathcal{F}$, $|\text{vars}(F)| = |F|$ and the cardinality of F and of its associated container $\mathcal{C}(F)$ are bounded by*

$$|F| \leq 8 \frac{kq^2}{\epsilon} \quad \text{and} \quad |\mathcal{C}(F)| \leq \left(1 - |F| \cdot \frac{\epsilon}{4kq^3 \ln \frac{kq}{\epsilon}} \right) n.$$

Proof. For any set of variables $S \subseteq X$ for which $\phi[S]$ is satisfiable, let I_S denote an independent set in H_ϕ corresponding to an assignment to the variables in S that satisfies $\phi[S]$. Let t_S denote an index that satisfies the container cardinality bound in [Lemma 5.6](#). Let

$$\mathcal{F} = \left\{ F_{t_S}(I_S) : S \subseteq X, \phi[S] \text{ is satisfiable and } |S| \geq \frac{8kq^2}{\epsilon} \right\},$$

and let the container function $\mathcal{C} : \mathcal{F} \rightarrow 2^X$ be the function that takes a fingerprint $F \in \mathcal{F}$ and returns $\text{vars}(C_{|F|/(q-1)}(F))$, where $C_t(\cdot)$ and $F_t(\cdot)$ are the functions defined by [Algorithm 3](#), as discussed following the algorithm.⁴

Before proving the conclusions of the lemma, first observe that for any set of variables S with $|S| \geq \frac{8kq^2}{\epsilon}$ we have that $|F_{t_S}(I_S)| = t_S \cdot (q-1)$. This holds because the upper bound on t_S from [Lemma 5.6](#) and the fact that exactly $(q-1)$ new vertices are added to the fingerprint in each step of the procedure. Using this, we have that

$$\mathcal{C}(F_{t_S}(I_S)) = \text{vars}(C_{|F_{t_S}(I_S)|/(q-1)}(F_{t_S}(I_S))) = \text{vars}(C_{t_S}(F_{t_S}(I_S))) = \text{vars}(C_{t_S}(I_S)),$$

where the third equality uses [Proposition 5.2](#).

Now, to show the first conclusion of the lemma, let S be any set of variables with $|S| \geq \frac{8kq^2}{\epsilon}$. By the basic properties of containers we have that $F_{t_S}(I_S) \subseteq I_S \subseteq C_{t_S}(I_S) \cup F_{t_S}(I_S)$ and so $\text{vars}(F_{t_S}(I_S)) \subseteq S \subseteq \text{vars}(C_{t_S}(I_S)) \cup \text{vars}(F_{t_S}(I_S))$. Using $\mathcal{C}(F_{t_S}(I_S)) = \text{vars}(C_{t_S}(I_S))$ we have that $\text{vars}(F_{t_S}(I_S)) \subseteq S \subseteq \mathcal{C}(F_{t_S}(I_S)) \cup \text{vars}(F_{t_S}(I_S))$, and using that $F_{t_S}(I_S) \in \mathcal{F}$ completes the proof of the first conclusion of the lemma.

⁴Note that by abuse of notation we go back and forth between an assignment to a set of variables and the set of vertices in H_ϕ corresponding to that assignment.

We now show the desired upper bounds on the size of every fingerprint $F \in \mathcal{F}$ and associated container $\mathcal{C}(F)$. Since $|F_{t_S}(I)| = t_S \cdot (q - 1)$ then by the upper bound of $t_S \leq \frac{8kq}{\epsilon}$ from [Lemma 5.6](#) we get the upper bound on $|F_{t_S}(I_S)|$, and by the upper bound on $|\text{vars}(C_{t_S}(I_S))|$ in [Lemma 5.6](#) we get the desired upper bound on $|\mathcal{C}(F_{t_S}(I_S))|$.

Finally observe that $|\text{vars}(F)| = |F|$ since I_S corresponds to an independent set with only one assignment per variable. $\square$

5.5 Proof of [Theorem 1.5](#)

We can now complete the proof of [Theorem 1.5](#), restated below.

Theorem 1.5. *The sample complexity of ϵ -testing the (q, k) -SAT property is $\tilde{O}(\frac{kq^3}{\epsilon})$.*

Proof. Let S be a random set of $s = c \frac{kq}{\epsilon} \ln^2(kq/\epsilon)$ variables drawn uniformly at random without replacement from the variable set of ϕ , where c is a large enough constant. The tester accepts a (q, k) -SAT instance ϕ if and only if $\phi[S]$ is satisfiable, where $\phi[S]$ represents the constraints of ϕ restricted to S . When ϕ is satisfiable, then $\phi[S]$ is also satisfiable and so the tester always accepts.

In the remainder of the proof, we upper bound the probability that $\phi[S]$ is satisfiable when ϕ is ϵ -far from satisfiable. We consider the collection $\mathcal{F}$ of fingerprints and the container function $\mathcal{C} : \mathcal{F} \rightarrow 2^X$ from [Lemma 5.1](#) which states that for every $S \subseteq X$ for which $|S| \geq \frac{8kq^2}{\epsilon}$ and $\phi[S]$ is satisfiable there exists $F \in \mathcal{F}$ such that $\text{vars}(F) \subseteq S \subseteq \mathcal{C}(F) \cup \text{vars}(F)$. Hence, $\phi[S]$ is satisfiable only when there exists $F \in \mathcal{F}$ such that S contains $\text{vars}(F)$ **and** the remaining vertices of S are contained in $\mathcal{C}(F)$. In other words, the probability that $\phi[S]$ is satisfiable is at most

$$\sum_{F \in \mathcal{F}} \Pr[\text{vars}(F) \subseteq S] \cdot \Pr[S \setminus \text{vars}(F) \subseteq \mathcal{C}(F) \mid \text{vars}(F) \subseteq S].$$

For a specific $F \in \mathcal{F}$, using that $|\text{vars}(F)| = |F|$, the probability that S contains $\text{vars}(F)$ is at most

$$\frac{\binom{n-|F|}{s-|F|}}{\binom{n}{s}} = \frac{\binom{s}{|F|}}{\binom{n}{|F|}},$$

because there are at most $\binom{n}{s}$ ways to choose S , and $\binom{n-|F|}{s-|F|}$ ways to choose an S that includes $\text{vars}(F)$. The probability that the remaining $s - |F|$ variables in S come from

$\mathcal{C}(F)$, conditioned on the fact that S contains $\text{vars}(F)$, can be upper bounded by

$$\left(\frac{|\mathcal{C}(F)|}{n}\right)^{s-|F|} \leq \left(1 - \frac{\epsilon |F|}{4kq^3 \ln(kq/\epsilon)}\right)^{s-|F|}.$$

Therefore, by applying a union bound over all values $j \leq \frac{8kq^2}{\epsilon}$ and all possible choices of $F \in \mathcal{F}$ of size j , the probability that $\phi[S]$ is satisfiable is at most

$$\begin{aligned} & \sum_{j=1}^{8kq^2/\epsilon} \sum_{\substack{F \in \mathcal{F} \\ |F|=j}} \frac{\binom{s}{|F|}}{\binom{n}{|F|}} \cdot \Pr[S \setminus \text{vars}(F) \subseteq \mathcal{C}(F) \mid \text{vars}(F) \subseteq S] \\ & \leq \sum_{j=1}^{8kq^2/\epsilon} \binom{nk}{j} \frac{\binom{s}{j}}{\binom{n}{j}} \exp\left(-\frac{\epsilon j}{4kq^3 \ln(kq/\epsilon)} \cdot (s - |F|)\right) \\ & \leq \sum_{j=1}^{8kq^2/\epsilon} \exp\left(j \ln(ske) - \frac{\epsilon j}{8kq^3 \ln(kq/\epsilon)} \cdot s\right) \end{aligned}$$

where the first inequality uses that there are at most $\binom{nk}{j}$ possible fingerprints of size j and the second inequality uses the bounds on the binomial $\left(\frac{m}{\ell}\right)^\ell \leq \binom{m}{\ell} \leq \left(\frac{m\epsilon}{\ell}\right)^\ell$ and that $s > 2|F|$. Finally, using the fact that $s = c \frac{kq^3}{\epsilon} \ln^2(kq/\epsilon)$ for a large enough constant c , the above expression can be upper bounded by $1/3$, and so the probability that $\phi[S]$ is satisfiable is at most $1/3$. $\square$

5.6 Proofs of Corollary 1.7 and Corollary 1.8

We complete this section by proving Corollaries 1.7 and 1.8, restated below, using Theorem 1.5. We prove Corollary 1.7 by showing that testing whether a q -uniform hypergraph is k -colorable is a special case of testing (q, k) -SAT. Similarly, we prove Corollary 1.8 by showing that testing whether a graph G satisfies a property in k -SHPP is a special case of testing $(2, k)$ -SAT.

Corollary 1.7. *The sample complexity of ϵ -testing the (q, k) -COLOR property is $\tilde{O}(\frac{kq^3}{\epsilon})$.*

Proof. Let H be any q -uniform hypergraph on n vertices. We wish to distinguish between the case that H is k -colorable and the case that H is ϵ -far from k -colorable. Let ϕ be a CSP on n variables, one for each vertex, that can each take values from $[k]$. The constraints of ϕ correspond to the hyperedges of H . In particular, for every hyperedge in H , there is a constraint to the variables associated with the vertices of the edge so that the variables cannot all be assigned the same value from $[k]$. Using that edges of H correspond exactly to constraints of ϕ , then ϕ is satisfiable if and only if H is k -colorable, and ϕ is ϵ -far from satisfiable if and only if H is ϵ -far from k -colorable.

We can test whether H is k -colorable by simulating the tester from [Theorem 1.5](#) on ϕ . In particular, a sample of variables of ϕ corresponds exactly to a sample of vertices of H , and when the tester asks to inspect $\phi[S]$, it can be simulated by inspecting $H[\hat{S}]$, where $\hat{S}$ is the vertices associated with the variables of S . $\square$

Corollary 1.8. *The sample complexity of ϵ -testing any property in k -SHPP is $\tilde{O}\left(\frac{k}{\epsilon}\right)$.*

Proof. Let Π be property in k -SHPP with parameters $\ell_{1,1}, \dots, \ell_{k,k}, u_{1,1}, \dots, u_{k,k}$ where $\ell_{i,j}, u_{i,j} \in \{0, 1\}$ for each $i, j \in [k]$. Let $G = (V, E)$ be any graph on n vertices. We wish to distinguish between the case that G is in property Π , and the case that G is ϵ -far from Π .

Let ϕ be a CSP on n variables $x_1, \dots, x_n$, one for each vertex $v_1, \dots, v_n \in V$, that can each take values from $[k]$. In order to define the constraints of ϕ , we define the following two sets $\Pi_0, \Pi_1 \subseteq [k]^2$. Let $\Pi_0 = \{(i, j) : i, j \in [k] \text{ and } \ell_{i,j} = 0\}$ and let $\Pi_1 = \{(i, j) : i, j \in [k] \text{ and } u_{i,j} = 1\}$. In other words, Π_0 corresponds to possible partition assignments (from $[k]$) to pairs of vertices such that if the vertices are not neighbours then they are consistent with the desired edge densities of Π . Similarly, Π_1 corresponds to assignments (from $[k]$) to pairs of vertices such that if the vertices *are* neighbours then they are consistent with the desired edge densities of Π . In this way, if it's possible to construct a partition of V where every pair of non-adjacent vertices in G is assigned to pairs of parts from Π_0 , and every pair of adjacent vertices in G is assigned to pairs of parts from Π_1 , then G is in k -SHPP.

Using Π_0 and Π_1 , we define the constraints as follows. For every pair of non-adjacent vertices (v_i, v_j) in G , ϕ has a constraint on x_i, x_j that evaluates to true if and only if x_i and x_j take pairs of assignments from Π_0 . Similarly, for every pair of adjacent vertices (v_i, v_j) in G , ϕ has a constraint on x_i, x_j that evaluates to true if and only if x_i and x_j take pairs of assignments from Π_1 .

First observe that ϕ is satisfiable if and only if G is in k -SHPP. Further, if G is ϵ -far from satisfiable, then, for any partitioning of G , at least ϵn^2 pairs of vertices violate the desired edge densities of Π . This corresponds to exactly $\epsilon n^2 > \epsilon \binom{n}{2}$ constraints of ϕ

evaluating to false for any assignment to the n variables, which means that ϕ is ϵ -far from satisfiable.

We can test whether G is in k -SHPP by simulating the tester from [Theorem 1.5](#) on ϕ . In particular, a sample of variables of ϕ corresponds exactly to a sample of vertices of G , and when the tester asks to inspect $\phi[S]$, it can be simulated by inspecting $G[\hat{S}]$, where $\hat{S}$ is the vertices associated with the variables of S . $\square$

Chapter 6

Tolerant Testing Independent Sets

In this section we prove [Theorem 1.10](#) on tolerant testing the ρ -INDEPSET property, restated below.

Theorem 1.10. *There exists a constant c such that for any $\epsilon > 0$ the sample complexity of $\left(\frac{\epsilon}{c \log^4(1/\epsilon)}, \epsilon\right)$ -tolerant testing the ρ -INDEPSET property is $\tilde{O}(\rho^3/\epsilon^2)$.*

First we formally state the container lemma and give a proof overview. Then, in [Section 6.3](#) we give the new fingerprint and container generating procedures. In [Sections 6.4](#) and [6.5](#) we prove a number of lemmas that we need to prove our new graph container lemma on sparse subgraphs, before finally proving the lemma in [Section 6.6](#). Then we prove [Theorem 1.10](#) using our new graph container lemma for sparse subgraphs. In [Section 6.8](#) we prove the application to non-canonical testers ([Corollary 1.11](#)).

6.1 Container Lemma Statement

In order to state our lemma, we require a new definition of fingerprints.

Definition 6.1. Let $U \subseteq V$ be a subset of vertices in a graph $G = (V, E)$. A *fingerprint* on U is a pair (F, R) where F is a sequence $(f_1, f_2, \dots, f_{|F|})$ where each $f_i \in U \times \{\uparrow, \downarrow\}$ and $R \in ([|F|] \times U) \cup \{\perp\}$. F is called the fingerprint sequence, and R is called the revision. The $\perp$ is used to denote that there is no revision. We use $\mathcal{F}(U)$ to denote the set of all possible fingerprints on U .

We can now state our new graph container lemma for sparse subgraphs.

Lemma 6.2. *Let $\epsilon < \rho^2/2$ and let $\ell \geq 1024 \log^4(1/\epsilon)$. If $G = (V, E)$ is ϵ -far from having a ρn independent set then there exists a set of fingerprints $\mathcal{F} \subseteq \mathcal{F}(V)$ and a function $C : \mathcal{F} \rightarrow 2^V$ that satisfies the following:*

1. *For any $(F, R) \in \mathcal{F}$, $|C(F, R)| \leq \left(1 - \frac{\epsilon}{2\rho^2}\right) \rho n$. Further, letting $\alpha_{(F,R)}$ denote the value in $[\frac{\epsilon}{2\rho^2}, 1]$ that satisfies $|C(F, R)| = (1 - \alpha_{(F,R)}) \rho n$, then*
2. *For any $(F, R) \in \mathcal{F}$, $|F| \leq \frac{32\alpha_{(F,R)}\rho^2 \log^2(8\rho/\epsilon)}{\epsilon}$, and*
3. *For any $J \subseteq V$ for which $G[J]$ has fewer than $\frac{\epsilon}{\ell\rho^2}|J|^2$ edges, there exists $(F, R) \in \mathcal{F}(J) \cap \mathcal{F}$ such that $|J \cap C(F, R)| \geq \left(1 - \frac{32\alpha_{(F,R)} \log^2(8\rho/\epsilon)}{\sqrt{\ell}}\right) |J|$.*

To prove [Lemma 6.2](#) we modify the standard container approach in three ways, which we describe briefly below (a more detailed overview can be found in [Section 6.2](#)). First, as mentioned above, it is useful to change the definition of a fingerprint to decouple the operations of removing the neighbours of the selected vertex and removing the higher degree vertices of the selected vertex. This is represented by the $\uparrow$ or $\downarrow$ that is paired with each vertex in the fingerprint sequence, where the $\uparrow$ corresponds to the operation of “remove higher degree vertices”, and the $\downarrow$ corresponds to the operation of “remove neighbours”. In other words, instead of viewing a fingerprint as a set of the most informative vertices, we view a fingerprint as a sequence of informative vertices along with instructions of $\uparrow$ or $\downarrow$ to denote how to construct the container using each fingerprint vertex. The second change has to do with how fingerprint vertices are selected. In the case of constructing a fingerprint for an independent set, at each step the most “informative” vertex is the vertex that tells the most information about the independent set by revealing lots of vertices that are *not* in the independent set. For a sparse subgraph J , the most informative vertex is the vertex that can eliminate many vertices from the current set of candidates, without eliminating too many vertices from J , and so the greedy procedure now selects the vertex that maximizes this ratio. The final change is due to the fact that, even with the above modifications, the fingerprint/container procedure still may not reach a container that contains a sufficiently high fraction of the sparse subgraph. As a result, we do an additional step that makes a “revision” to the container. The revision is represented by an index of the fingerprint sequence and a vertex in the sparse subgraph, and can be viewed as an additional instruction on how to revise the container to meet the desired containment bounds.

We note that there is an important trade off in [Lemma 6.2](#) between the size of the container and the size of the fingerprint, and this trade off is balanced with the fraction of J that is contained in the container. In particular, for sufficiently large ℓ (at least roughly $2056 \log^4(1/\epsilon)$), for any sparse subgraph $G[J]$ there exists a fingerprint $(F, R) \in \mathcal{F}$ such that $|C(F, R)| = (1-\alpha)\rho n$ and at least $(1-\alpha/2)|J|$ of J is contained in the container. These trade offs are necessary in order to prove the optimal bounds on the sample complexity in [Theorem 1.10](#).

Notations. Throughout this section we use the following notations. Let $G = (V, E)$ be a graph. For $S \subseteq V$ we use the notation $S_{\uparrow v}$ to denote all vertices in S with more neighbours in S than v . In other words,

$$S_{\uparrow v} = \{u \in S : |N(u) \cap S| > |N(v) \cap S|\}.$$

We define the edge density of a subgraph $S \subseteq V$ as $\frac{|E(S)|}{\binom{|S|}{2}}$. Throughout, we say *subgraph* to refer to the *induced subgraph* $G[S]$. We use $\log(\cdot)$ to denote $\log_2(\cdot)$.

6.2 Proof Overview

For a graph $G = (V, E)$ that is ϵ -far from ρ -INDEPSET, we prove [Lemma 6.2](#) by constructing a set $\mathcal{F} \subseteq \mathcal{F}(V)$ of fingerprints, and showing that the collection satisfies the desired conclusions. In particular, we give a container function $C : \mathcal{F}(V) \rightarrow 2^V$, and for every set J such that $G[J]$ is sparse we describe how to construct a fingerprint (F, R) such that $C(F, R)$ satisfies the conclusions of the lemma: $|C(F, R)|$ is not too large, $|F|$ is not too large, and $C(F, R)$ contains a relatively large amount of J . Then the collection $\mathcal{F}$ is the set of all fingerprints formed in this way. In what follows we briefly describe the construction of F and R , what the container function is, and the key ideas towards showing that the constructions satisfy the desired conclusions of the lemma. Before doing that, first recall that we define a fingerprint as a pair (F, R) such that F is a sequence of vertices along with instructions from $\{\uparrow, \downarrow\}$, and R is a pair (i, v) where $i \in [|F|]$ and v is a vertex ([Definition 6.1](#)). We refer to F as the fingerprint sequence and R as the revision.

Let $\ell = \text{polylog}(1/\epsilon)$. For any subgraph $G[J]$ with edge density at most $\frac{\epsilon}{\ell \rho^2}$ (note that if G is ϵ -far from ρ -INDEPSET then the edge density of G is at least $\frac{\epsilon}{\rho^2}$), we construct F through an iterative greedy procedure ([Algorithm 4](#)). First, initialize a container to V , then, at each step, select a vertex $v \in J$ and an operation from $\uparrow$ or $\downarrow$, where $\uparrow$ corresponds

to removing higher degree vertices than v and $\downarrow$ corresponds to removing neighbours of v , such that running the operation on the current container removes a large number of vertices in the container and removes a relatively small number of vertices from J in the process. In particular, in each step the procedure selects the vertex and operation that *maximizes* the ratio of vertices removed from the container divided by the number of vertices of J removed from the container. We add the pair containing the vertex v and the operation from $\{\uparrow, \downarrow\}$ to the fingerprint sequence F , run the operation on the container, and repeat this process until we reach a container with fewer than $\epsilon n^2/4$ edges. This final container, which for now we call C_F , can be viewed as a first “guess” at a container for J .

Our first lemma, [Lemma 6.5](#), shows that C_F already satisfies some useful properties. In particular, we show that at each step of the procedure described above, the ratio of vertices removed from the current container divided by the vertices removed from J is at least roughly $\sqrt{\ell} \cdot \frac{n}{|J|}$.¹ In other words, if $\frac{1}{\sqrt{\ell}}|J|$ vertices of J are removed from the container, then roughly n vertices are removed from the container, and since the initial container size is of size n , we have that C_F contains at least roughly $(1 - \frac{1}{\sqrt{\ell}})|J|$ of J .

To develop some intuition on why this is the case consider the following example. Let $G = G(n, \frac{4\epsilon}{\rho^2})$ be the random graph on n vertices where each edge occurs with probability $\frac{4\epsilon}{\rho^2}$. With high probability this graph is ϵ -far from having a ρn independent set. Let J be a subset of vertices such that the edge density of $G[J]$ is at most $\frac{\epsilon}{\ell\rho^2}$. Consider the first step of the procedure described above. Observe that the average degree in $G[J]$ is less than $\frac{\epsilon}{\ell\rho^2}|J|$, hence, the procedure can select a vertex $v \in J$ with at most the average degree in $G[J]$, and the operation $\downarrow$ to remove neighbours, and doing so removes at least roughly $\frac{4\epsilon}{\rho^2}n$ vertices from the container while removing fewer than $\frac{\epsilon}{\ell\rho^2}|J|$ vertices of J . This gives a removal ratio of roughly $4\ell \cdot \frac{n}{|J|}$.

In order to show the $\sqrt{\ell} \cdot \frac{n}{|J|}$ ratio in general, we require a general lemma ([Lemma 6.4](#)) about the degrees of vertices in a dense graph. This lemma shows that for any subset $U \subseteq V$ there exists a degree threshold d such that at least roughly $|E(U)|/d$ vertices in U have degree larger than d in U . Hence, in each step of the procedure, at least roughly $\frac{1}{d} \cdot \frac{\epsilon}{\rho^2}n^2$ vertices in the current container have degree at least d . If there exists a vertex $v \in J$ with degree less than $\frac{1}{\sqrt{\ell}} \frac{d}{n}|J|$ in $G[J]$ and degree more than d in the current container, then selecting it and operation $\downarrow$ removes $\sqrt{\ell} \cdot \frac{n}{|J|}$ times more vertices from the container than from J . Otherwise, there are only at most $\frac{\epsilon}{\sqrt{\ell}\rho^2} \frac{n}{d}|J|$ vertices in J with degree larger than d in the current container based on the sparsity of $G[J]$. Hence, selecting the vertex in J

¹Note that in the full proof this ratio depends on the size of the current container instead of n , but for simplicity here we assume that the current container is of size roughly n .

with maximum degree in the current container that is less than d , and operation $\uparrow$, again removes roughly $\sqrt{\ell} \cdot \frac{n}{|J|}$ times more vertices from the container than from J .

In order to prove the stronger containment properties of [Lemma 6.2](#), we start by letting $r \cdot \frac{n}{|J|}$ be the smallest ratio achieved throughout the running of the procedure (by [Lemma 6.5](#) we have that $r \geq \sqrt{\ell}$, but in many cases it may be much larger). Generalizing the argument above, we can show that C_F contains at least roughly $(1 - \frac{1}{r})|J|$ of J . Further, we can also show that the maximum size of the fingerprint is roughly $\frac{\sqrt{\ell}}{r} \frac{\rho^2}{\epsilon}$ (note: in order to show this we require that the greedy procedure described above only selects fingerprint vertices that remove some minimum threshold of vertices in the container). This means that if $|C_F|$ is smaller than roughly $(1 - \frac{\sqrt{\ell}}{r})\rho n$ then we have the desired bounds in the conclusions of the [Lemma 6.2](#) and so no revision is necessary and we set $R = \perp$.

In the case that $|C_F|$ is larger than $(1 - \frac{\sqrt{\ell}}{r})\rho n$, we show (in [Lemma 6.6](#)) that the container C_F can be revised to C'_F by selecting a vertex $v \in J$ and removing all vertices from C_F that had higher degree than v at some time step i of the procedure, and importantly, this revised container C'_F satisfies the desired conclusions of the lemma. We give more details below on how to select such a vertex, but before that we show how this implies [Lemma 6.2](#). We make two important observations. First, even though C_F is constructed through a procedure that depends on J , C_F can be reconstructed solely from F : initialize a container as V and then iteratively run the operations from F . Also, C'_F can be constructed solely from F and R : construct C_F and then, if $R = (i, v)$, revise C_F by removing all vertices that had higher degree than v at step i . This is exactly what our container function ([Algorithm 5](#)) does on input (F, R) . To summarize, for any sparse subgraph $G[J]$ we construct a fingerprint (F, R) such that $C(F, R)$ satisfies the conclusions of the lemma, and the collection $\mathcal{F}$ is the set of all fingerprints formed in this way.

Selecting a Revision Vertex. Consider the time step i of the procedure that achieved the smallest ratio $r \cdot \frac{n}{|J|}$ of vertices removed from the container divided by vertices of J removed from the container, and let the container at that time step be D . As discussed above, if $|C_F|$ is sufficiently small then it already satisfies the desired conclusions of [Lemma 6.2](#), and this section only applies to the case that $|C_F|$ is large. To illustrate the key ideas, for now we will assume that $|D| \approx n$.

First, if $|C_F|$ is large then we can make an important observation about the structure of D because $C_F \subseteq D$ and $G[C_F]$ is sparse. In particular, we can show that either $|E(C_F, D \setminus C_F)|$ or $|E(D \setminus C_F)|$ is relatively large because G is ϵ -far from ρ -INDEPSET. In slightly more detail, if $|C(F)| > (1 - \frac{\sqrt{\ell}}{r})\rho n$ then we show that either $|E(C_F, D \setminus C_F)| \gtrsim \frac{\epsilon r}{\sqrt{\ell} \rho^2} |C_F| \cdot n$

or $|E(D \setminus C_F)| \gtrsim \frac{\epsilon r^2}{\ell \rho^2} n^2$ because otherwise we can add a random set of $\frac{\sqrt{\ell}}{r} \rho n$ vertices to C_F without adding too many edges and find that G is not ϵ -far from ρ -INDEPSET.

In order to use these lower bounds on the edges, we again use [Lemma 6.4](#) to argue about the degrees of vertices in dense subgraphs. In particular, by [Lemma 6.4](#), if there are a lot of edges in $E(D \setminus C_F)$ then there exists a degree threshold d such that at least roughly $\frac{1}{d} \frac{\epsilon r^2}{\ell \rho^2} n^2$ vertices in $D \setminus C_F$ have degree larger than d in $G[D]$. If there exists a vertex in J with degree at least d in $G[D]$ and degree smaller than roughly $\frac{1}{r} \frac{d}{n} |J|$ in $G[J]$ then the fingerprint procedure could have selected this vertex and the operation $\downarrow$ to remove $r \frac{n}{|J|}$ times more vertices from the container than from J , which is a contradiction since the ratio at this time step was $r \cdot \frac{n}{|J|}$. Otherwise, all the vertices in J with degree at least d in $G[D]$ have degree at least roughly $\frac{1}{r} \frac{d}{n} |J|$ in $G[J]$, and so there can only be at most $\frac{\epsilon r}{\ell \rho^2} \frac{n}{d} |J|$ of them based on the sparsity of $G[J]$. In this case, the fingerprint procedure could have selected the vertex in J with highest degree in D that is less than d , and the operation $\uparrow$ to remove $r \frac{n}{|J|}$ times more vertices from the container than from J , which again is a contradiction.

Since the first case leads to contradiction, it must be the case that $|E(C_F, D \setminus C_F)|$ is large. In this case, we again use [Lemma 6.4](#) to find that there is a degree threshold d such that at least roughly $\frac{1}{d} \frac{\epsilon r}{\sqrt{\ell} \rho^2} |C_F| n$ vertices in C have degree larger than d in $G[D]$. Using a similar argument, there can only be at most $\frac{\epsilon r}{\rho^2 \ell} \frac{n}{d} |J|$ vertices in J with degree larger than d in $G[D]$ otherwise the fingerprint procedure should have achieved a larger removal ratio, and so by removing vertices from C_F with degree larger than d in $G[D]$ we remove roughly $\sqrt{\ell} \cdot \frac{|C_F|}{|J|}$ times more vertices from C_F than vertices from J .

6.3 Fingerprint and Container Procedures

The greedy procedure used to construct the fingerprint sequence F for a sparse set J is in [Algorithm 4](#). We initialize the container to V , and the fingerprint to an empty sequence. Then, in each step, we consider each $v \in J$ and the two possible operations of “remove neighbours” or “remove higher degree vertices”. For the first type of operation, the number of vertices that can be removed from the current container C_{t-1} is $\deg_{G[C_{t-1} \cup \{v\}]}(v)$, where we have to use $C_{t-1} \cup \{v\}$ because v may not be in C_{t-1} , and the number of vertices that would be removed from J is at most $\deg_{G[J]}(v)$. Similarly, for the second type of operation, the vertices that can be removed from the current container C_{t-1} are $\{u \in C_{t-1} : |N(u) \cap C_{t-1}| > |N(v) \cap C_{t-1}|\}$, which from now on we will denote by $(C_{t-1})_{\uparrow v}$, and the number of vertices in J that would be removed in the process is at most $|(C_{t-1})_{\uparrow v} \cap J|$.

We desire the maximum ratio between the number of vertices removed in C_{t-1} and the number of vertices removed in J , however there is a small challenge. There may be a vertex in J that maximizes this ratio, but only removes a very small number of vertices in J (say just 1 vertex) and a very small number of vertices in C_{t-1} , and this would cause the procedure to take too long. To handle this, we take the maximum of the vertices removed in J with a threshold $\frac{\epsilon}{\sqrt{\ell}\rho^2}|J|$. We select this threshold to give the desired trade offs in the fingerprint size and the amount of J that is contained in the final container. Formally, we define the maximal removal ratio as follows.

Definition 6.3. Given a graph $G = (V, E)$, subsets of vertices $J, C \subseteq V$ and parameters ϵ, ρ, ℓ , define the *maximum removal ratio*, denoted $\text{MRR}_G(J, C)$, as

$$\text{MRR}_G(J, C) := \max_{v \in J} \max \left(\frac{\deg_{G[C \cup \{v\}]}(v)}{\max(\deg_{G[J]}(v), \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|)}, \frac{|C_{\uparrow v}|}{\max(|C_{\uparrow v} \cap J|, \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|)} \right).$$

While this quantity depends on ϵ, ρ and ℓ , we suppress this dependence in the notation $\text{MRR}_G(J, C)$ for clarity.

Overall, in each step of the fingerprint procedure we select the vertex $v \in J$ and operation that achieves the maximum removal ratio, add it to the fingerprint and run the operation on the container, and repeat.

The first part of the container function, [Algorithm 5](#), essentially simulates the container produced in the fingerprint procedure. In particular, given a fingerprint sequence F it repeatedly runs the operations on the container described by the fingerprint, and eventually finds container $C_{|F|}$. Afterwards, if a revision pair (i, u) is given, then returns $C_{|F|} \setminus (C_i)_{\uparrow u}$, and otherwise returns $C_{|F|}$.

6.4 An Initial Containment Lemma

In this section we prove that [Algorithm 4](#) makes reasonable progress in every step, which also implies an initial weak containment property on the final container constructed in [Algorithm 4](#). This will be used as a starting point in the proof of the graph container lemma ([Lemma 6.2](#)) in [Section 6.6](#).

Before we state and prove the lemma, we require the following [Lemma 6.4](#) which says that if there are subsets of vertices U, W in V such that $|E(U, W)|$ is large, then there are a relatively large number of vertices in U that have a relatively large number of neighbours

Algorithm 4: FINGERPRINT GENERATOR

Input: A graph $G = (V, E)$, and $J \subseteq V$ for which $G[J]$ has fewer than $\frac{\epsilon}{\ell\rho^2}|J|^2$ edges

```

1 Initialize  $t = 0, F \leftarrow ()$  and  $C_0 \leftarrow V$ ;
2 while  $G[C_t]$  has more than  $\epsilon n^2/4$  edges do
3    $t \leftarrow t + 1$ ;
4    $v_t \leftarrow \arg \max_{v \in J} \max \left( \frac{\deg_{G[C_{t-1} \cup \{v\}]}(v)}{\max(\deg_{G[J]}(v), \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|)}, \frac{|(C_{t-1})_{\uparrow v}|}{\max(|(C_{t-1})_{\uparrow v} \cap J|, \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|)} \right)$ ;
5   if  $v_t$  comes from the first type then
6      $C_t \leftarrow C_{t-1} \setminus N(v_t)$ ;
7     Append  $(v_t, \downarrow)$  to  $F$ ;
8   else
9      $C_t \leftarrow C_{t-1} \setminus (C_{t-1})_{\uparrow v_t}$ ;
10    Append  $(v_t, \uparrow)$  to  $F$ ;
11 Return  $F$ ;

```

Algorithm 5: CONTAINER GENERATOR

Input: A graph $G = (V, E)$, and fingerprint sequence F and revision R

```

1 Initialize  $C_0 \leftarrow V$ ;
2 for  $t = 1, 2, \dots, |F|$  do
3   if  $F[t] = (v, \downarrow)$ , for some  $v \in V$  then
4      $C_t \leftarrow C_{t-1} \setminus N(v)$ 
5   else if  $F[t] = (v, \uparrow)$ , for some  $v \in V$  then
6      $C_t \leftarrow C_{t-1} \setminus (C_{t-1})_{\uparrow v}$ 
7 if  $R = (i, u)$  then
8   Return  $C_{|F|} \setminus (C_i)_{\uparrow u}$ 
9 else
10  Return  $C_{|F|}$ ;

```

in W . In particular, there is a threshold $d \leq |W|$ such that at least roughly $|E(U, W)|/d$ vertices in U have more than d neighbours in W . [Lemma 6.4](#) will be used in the proof of [Lemma 6.5](#) as well as [Lemma 6.6](#) in the next section.

Lemma 6.4. *Let $G = (V, E)$ be a graph and let $U, W \subseteq V$. If $|E(U, W)| \geq m_{UW}$ then there exists $\frac{m_{UW}}{2|U|} \leq d \leq |W|$ such that the number of vertices in U with more than d neighbours in W is at least*

$$\frac{m_{UW}}{4d \log(2|U||W|/m_{UW})}.$$

Proof. For ease of presentation let $\delta = \frac{m_{UW}}{|U||W|}$, which is roughly the edge density between U and W . Let $d_k = \frac{1}{2^k}|W|$ for $k = 0, \dots, \log(2/\delta)$. Observe that for all k , $\frac{m_{UW}}{2|U|} \leq d_k \leq |W|$. We claim that there exists a k such that the number of vertices in U with more than d_k neighbours in W is at least $\frac{m_{UW}}{4 \log(2/\delta) d_k}$. To prove the claim, we suppose this is not the case, and find a contradiction. For $k = 1, \dots, \log(2/\delta)$, let U_k be the vertices in U with more than d_k neighbours and most d_{k-1} neighbours in W , and let $U_{\log(2/\delta)+1}$ be the vertices with at most $d_{\log(2/\delta)} = \frac{m_{UW}}{2|U|}$ neighbours in W . Then $U_1, \dots, U_{\log(2/\delta)+1}$ partitions U , and so the number of edges in $E(U, W)$ is at most

$$|U_{\log(2/\delta)+1}| \cdot \frac{m_{UW}}{2|U|} + \sum_{k=1}^{\log(2/\delta)} |U_k| \cdot d_{k-1} < \frac{m_{UW}}{2} + \sum_{k=1}^{\log(2/\delta)} \frac{m_{UW}}{4 \log(2/\delta) d_k} d_{k-1} \leq m_{UW},$$

where the first inequality uses the trivial bound of $|U_{\log(2/\delta)+1}| \leq |U|$, as well as the bound $|U_k| < \frac{m_{UW}}{4d_k \log(2/\delta)}$ for all k , which holds because of the assumption that there are fewer than $\frac{m_{UW}}{4 \log(2/\delta) d_k}$ vertices in U that have more than d_k neighbours in W . Hence, we have that $|E(U, W)| < m_{UW}$, which is a contradiction, and so there exists $\frac{m_{UW}}{2|U|} \leq d \leq |W|$ such that the number of vertices in U with more than d neighbours in W is at least $\frac{m_{UW}}{4d \log(2|U||W|/m_{UW})}$. $\square$

We can now state and prove an initial containment lemma. We show that if G is ϵ -far from ρ -INDEPSET, then every step of [Algorithm 4](#) makes progress, and we use this to give an initial upper bound on the size of the fingerprint sequence and an initial lower bound on the fraction of J that is in the final container of [Algorithm 4](#).

Lemma 6.5. *Let $\epsilon \leq \rho^2/2$ and $\ell \geq 256$. Let $G = (V, E)$ be ϵ -far from having a pn independent set, let $J \subseteq V$ be a set of vertices for which $G[J]$ has fewer than $\frac{\epsilon}{\ell \rho^2} |J|^2$ edges, and let C be the final container in [Algorithm 4](#). Then for each step t of the procedure,*

$$\text{MRR}_G(J, C_{t-1}) \geq \frac{\sqrt{\ell} \max(|C_{t-1}|, \rho n)}{16 \log(8\rho/\epsilon) |J|} \quad (6.1)$$

and $|C \cap J| \geq \left(1 - \frac{16 \log^2(8\rho/\epsilon)}{\sqrt{\ell}}\right) |J|$ and $|F| \leq \frac{16\rho^2 \log^2(8\rho/\epsilon)}{\epsilon}$.

Proof. Let t be a time step during the procedure of [Algorithm 4](#). We will first prove [Eq. \(6.1\)](#). First observe that $G[C_{t-1}]$ has at least $\frac{\epsilon}{4\rho^2} (\max(|C_{t-1}|, \rho n))^2$ edges. When $|C_{t-1}| < \rho n$ this follows because C_{t-1} has at least $\epsilon n^2/4$ edges, and when $|C_{t-1}| \geq \rho n$ this follows because G is ϵ -far from having a ρn independent set, and so any set of size larger than ρn has edge density at least $\frac{\epsilon}{\rho^2}$. Hence, by applying [Lemma 6.4](#) with $U = W = C_{t-1}$, there exists d such that $\frac{\epsilon}{8\rho^2|C_{t-1}|} (\max(|C_{t-1}|, \rho n))^2 \leq d \leq |C_{t-1}|$ and at least $\frac{\epsilon(\max(|C_{t-1}|, \rho n))^2}{16\rho^2 d \log(8\rho^2/\epsilon)}$ vertices in C_{t-1} have degree larger than d in $G[C_{t-1}]$.

Consider the vertices in J with more than d neighbours in C_{t-1} , we consider three cases. First, if there exists such a vertex $v \in J$ with degree smaller than $\frac{\epsilon}{\sqrt{\ell}\rho^2}|J|$ in $G[J]$, then

$$\frac{\deg_{G[C_{t-1} \cup \{v\}]}(v)}{\max(\deg_{G[J]}(v), \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|)} \geq \frac{d}{\frac{\epsilon}{\sqrt{\ell}\rho^2}|J|} \geq \frac{\frac{\epsilon}{8\rho^2|C_{t-1}|} (\max(|C_{t-1}|, \rho n))^2}{\frac{\epsilon}{\sqrt{\ell}\rho^2}|J|} \geq \frac{\sqrt{\ell} \max(|C_{t-1}|, \rho n)}{8|J|},$$

and so [Eq. \(6.1\)](#) holds. Next, if there exists such a vertex $v \in J$ with degree in $G[J]$ at least $\frac{\epsilon}{\sqrt{\ell}\rho^2}|J|$ but at most $\frac{2d|J|}{\sqrt{\ell} \max(|C_{t-1}|, \rho n)}$, then again observe that

$$\frac{\deg_{G[C_{t-1} \cup \{v\}]}(v)}{\max(\deg_{G[J]}(v), \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|)} \geq \frac{d}{\frac{2d|J|}{\sqrt{\ell} \max(|C_{t-1}|, \rho n)}} \geq \frac{\sqrt{\ell} \max(|C_{t-1}|, \rho n)}{2|J|},$$

and so we again we find that [Eq. \(6.1\)](#) holds. Finally, in the case that neither of the first two cases hold, we count the number of vertices in J with more than d neighbours in C_{t-1} . Since neither of the first two cases hold, then any such vertex has degree more than $\frac{2d|J|}{\sqrt{\ell} \max(|C_{t-1}|, \rho n)}$ in $G[J]$, and so there can be at most $\frac{\epsilon \max(|C_{t-1}|, \rho n)|J|}{\sqrt{\ell}\rho^2 d}$ such vertices because of the sparsity of $G[J]$. Let v be the vertex in J that has the most number of neighbours in C_{t-1} that is at most d . Such a vertex exists because $\frac{\epsilon \max(|C_{t-1}|, \rho n)|J|}{\sqrt{\ell}\rho^2 d} \leq |J|/2$ using the lower bound on d and that $\ell \geq 256$. Then

$$\frac{|(C_{t-1})_{\uparrow v}|}{\max(|(C_{t-1})_{\uparrow v} \cap J|, \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|)} \geq \frac{\frac{\epsilon(\max(|C_{t-1}|, \rho n))^2}{16\rho^2 d \log(8\rho^2/\epsilon)}}{\max(\frac{\epsilon \max(|C_{t-1}|, \rho n)|J|}{\sqrt{\ell}\rho^2 d}, \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|)} \geq \frac{\sqrt{\ell} \max(|C_{t-1}|, \rho n)}{16 \log(8\rho^2/\epsilon)|J|},$$

where the last inequality uses that $d \leq |C_{t-1}|$ in the case that $\frac{\epsilon \max(|C_{t-1}|, \rho n)|J|}{\sqrt{\ell}\rho^2 d} \leq \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|$. Using that $\log(8\rho^2/\epsilon) \leq \log(8\rho/\epsilon)$ completes the proof of [Eq. \(6.1\)](#).

Now, for any step t of the procedure, let $\beta_t|J|$ be the number of vertices removed from J when constructing C_t . Eq. (6.1) can be written as the following lower bound on the number of vertices removed from C_{t-1} to construct C_t :

$$|C_{t-1}| - |C_t| \geq \frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)} \frac{\max\left(\beta_t|J|, \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|\right)}{|J|} \max(|C_{t-1}|, \rho n). \quad (6.2)$$

Let t^* be the smallest index for which $|C_{t^*}| \leq \rho n$. Break up the procedure into three parts: when $t < t^*$, when $t = t^*$ and $t > t^*$. First, for each $t < t^*$, rearrange Eq. (6.2) to

$$|C_t| \leq \left(1 - \frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)} \max\left(\beta_t, \frac{\epsilon}{\sqrt{\ell}\rho^2}\right)\right) |C_{t-1}|,$$

and by applying for every $t < t^*$ we find that

$$|C_{t^*-1}| \leq \prod_{t=1}^{t^*-1} \left(1 - \frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)} \max\left(\beta_t, \frac{\epsilon}{\sqrt{\ell}\rho^2}\right)\right) \cdot |C_0|.$$

Since $|C_{t^*-1}| \geq \rho n$ and $|C_0| = n$, and using the upper bound $(1 - x) \leq e^{-x}$, we find

$$\rho \leq \exp\left(-\frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)} \sum_{t=1}^{t^*-1} \max\left(\beta_t, \frac{\epsilon}{\sqrt{\ell}\rho^2}\right)\right). \quad (6.3)$$

Using that $\max(\cdot)$ is at least as large as the first operand, Eq. (6.3) implies that $\sum_{t=1}^{t^*-1} \beta_t \leq \frac{16 \log(8\rho/\epsilon) \ln(1/\rho)}{\sqrt{\ell}}$. Similarly, using that $\max(\cdot)$ is at least as large as the second operand, Eq. (6.3) implies that $t^* \leq \frac{16\rho^2 \log(8\rho/\epsilon) \ln(1/\rho)}{\epsilon} + 1$.

In the case $t = t^*$, Eq. (6.2) implies that $\beta_t \leq \frac{16 \log(8\rho/\epsilon)}{\sqrt{\ell}}$ because $|C_{t^*}| \geq 0$. Finally, for each $t > t^*$, Eq. (6.2) states that

$$|C_{t-1}| - |C_t| \geq \frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)} \frac{\max\left(\beta_t|J|, \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|\right)}{|J|} \rho n.$$

Summing up over all $t > t^*$ we find

$$|C_{t^*}| - |C_{|F|}| \geq \sum_{t=t^*+1}^{|F|} \frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)} \frac{\max\left(\beta_t|J|, \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|\right)}{|J|} \rho n.$$

Since $|C_{t^*}| \leq \rho n$ and $|C_{|F|}| \geq 0$, lower bounding the max by the first operand gives $\sum_{t=t^*+1}^{|F|} \beta_t \leq \frac{16 \log(8\rho/\epsilon)}{\sqrt{\ell}}$, and lower bounding the max by the second operand gives $|F| - t^* \leq \frac{16 \log(8\rho/\epsilon) \rho^2}{\epsilon}$. Combining, we have that $\sum_{t=1}^{|F|} \beta_t \leq \frac{16 \log(8\rho/\epsilon) (\ln(1/\rho) + 2)}{\sqrt{\ell}}$ and $|F| \leq \frac{16 \rho^2 \log(8\rho/\epsilon) (\ln(1/\rho) + 2)}{\epsilon}$, which can be upper bounded as $\sum_{t=1}^{|F|} \beta_t \leq \frac{16 \log^2(8\rho/\epsilon)}{\sqrt{\ell}}$ and $|F| \leq \frac{16 \rho^2 \log^2(8\rho/\epsilon)}{\epsilon}$ by using $\epsilon \leq \rho^2/2$, which completes the proof of the lemma. $\square$

6.5 Revision Lemma

Our next lemma shows the existence of a “revision” in the case that the final container in [Algorithm 4](#) is too large relative to the fraction of J that is contained in it. In particular, suppose C is the final container in [Algorithm 4](#) and D is the container at some step of the procedure in [Algorithm 4](#). This lemma roughly says that if $G[C]$ is large and sparse and there wasn’t a good option for the fingerprint selection in [Algorithm 4](#), then there must be a vertex v in J such that there are a relatively high number of vertices in C with higher degree than v in D . This vertex can be used to reduce the size of the container C .

Lemma 6.6. *Let $\epsilon \leq \rho^2/2$ and let $\ell \geq 16$. Let $G = (V, E)$ be ϵ -far from having a ρn independent set, let $J \subseteq V$ be a set of vertices for which $G[J]$ has fewer than $\frac{\epsilon}{\ell \rho^2} |J|^2$ edges, and let $C, D \subseteq V$ be subsets of vertices such that $C \subseteq D$, $|E(C)| \leq \epsilon n^2/4$, and $|D| \geq \rho n$. If for some $r > \frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)} : |C| \geq \left(1 - \frac{\sqrt{\ell}}{16r \log(8\rho/\epsilon)}\right) \rho n$, and*

$$\text{MRR}_G(J, D) \leq \frac{r \left(|D| - \left(1 - \frac{\sqrt{\ell}}{16r \log(8\rho/\epsilon)}\right) \rho n \right)}{|J|} \quad (6.4)$$

then there exists $v \in J$ and $\gamma \geq \frac{\sqrt{\ell}}{16r \log(8\rho/\epsilon)}$ such that $|C \setminus D_{\uparrow v}| \leq (1 - \gamma) \rho n$ and $|D_{\uparrow v} \cap J| \leq \frac{16\gamma \log^2(8\rho/\epsilon)}{\sqrt{\ell}} |J|$.

Proof. Let Y be any subset of C of size $\left(1 - \frac{\sqrt{\ell}}{16r \log(8\rho/\epsilon)}\right) \rho n$. We first claim that for any $d \geq \frac{\epsilon r |D \setminus Y|}{\sqrt{\ell} \rho^2}$, the number of vertices in J with more than d neighbours in D is at most $\frac{2\epsilon r |J| |D \setminus Y|}{d \ell \rho^2}$. To show this, observe that any such vertex $v \in J$ has degree larger than $\frac{\epsilon}{\sqrt{\ell} \rho^2} |J|$ in $G[J]$ because otherwise $\frac{\deg_{G[D \cup \{v\}]}(v)}{\max(\deg_{G[J]}(v), \frac{\epsilon}{\sqrt{\ell} \rho^2} |J|)} > \frac{d}{\frac{\epsilon}{\sqrt{\ell} \rho^2} |J|} \geq \frac{r |D \setminus Y|}{|J|}$, which contradicts [Eq. \(6.4\)](#). Further, any such vertex $v \in J$ has degree larger than $\frac{d |J|}{r |D \setminus Y|}$ in $G[J]$ because otherwise we

find that $\frac{\deg_{G[D \cup \{v\}]}(v)}{\max(\deg_{G[J]}(v), \frac{\epsilon}{\sqrt{\ell\rho^2}}|J|)} > \frac{r|D \setminus Y|}{|J|}$, again contradicting Eq. (6.4). Hence, since any such vertex $v \in J$ has degree larger than $\frac{d|J|}{r|D \setminus Y|}$ in $G[J]$, then there can be at most $\frac{2\epsilon r|J||D \setminus Y|}{d\ell\rho^2}$ of them, based on the sparsity of J , which completes the proof of the claim. We also note that $\frac{2\epsilon r|J||D \setminus Y|}{d\ell\rho^2} \leq |J|/2$ using the lower bound on d and that $\ell \geq 16$, and so there always exists $v \in J$ with at most d neighbours in D .

We now claim that $|E(Y, D \setminus Y)| > \frac{4\log(8\rho/\epsilon)\epsilon r}{\sqrt{\ell\rho^2}}\rho n|D \setminus Y|$. Before proving this, we show how this implies the lemma. Apply Lemma 6.4 (with $U = Y$ and $W = D \setminus Y$) to find there exists $d_Y \in \left[\frac{2\log(8\rho/\epsilon)\epsilon r}{\sqrt{\ell\rho^2}|Y|}\rho n \cdot |D \setminus Y|, |D \setminus Y|\right]$ such that at least $\frac{\epsilon r}{\sqrt{\ell\rho^2}d_Y}\rho n \cdot |D \setminus Y|$ vertices in Y have degree larger than d_Y in $G[D]$. Let v_Y be the vertex in J that has largest degree in $G[D \cup \{v_Y\}]$ less than d_Y . Since $d_Y \geq \frac{\epsilon r|D \setminus Y|}{\sqrt{\ell\rho^2}}$, the first claim gives $|D_{\uparrow v_Y} \cap J| \leq \frac{2\epsilon r|J||D \setminus Y|}{d_Y\ell\rho^2}$, and we find that

$$\frac{|Y \cap D_{\uparrow v_Y}|}{\max(|D_{\uparrow v_Y} \cap J|, \frac{\epsilon}{\sqrt{\ell\rho^2}}|J|)} \geq \frac{\frac{\epsilon r}{\sqrt{\ell\rho^2}d_Y}\rho n \cdot |D \setminus Y|}{\max\left(\frac{2\epsilon r|J||D \setminus Y|}{d_Y\ell\rho^2}, \frac{\epsilon}{\sqrt{\ell\rho^2}}|J|\right)} \geq \frac{\sqrt{\ell}\rho n}{16\log(8\rho/\epsilon)|J|},$$

where we use that $r \geq \frac{\sqrt{\ell}}{16\log(8\rho/\epsilon)}$ and $|D \setminus Y| \geq d_Y$.

Since this applies for any $Y \subseteq C$ of size $\left(1 - \frac{\sqrt{\ell}}{16r\log(8\rho/\epsilon)}\right)\rho n$, consider all such choices of Y and let v be the v_Y with the minimum number of neighbours in D . We claim that v satisfies the conclusion of the lemma. First, if $|J \cap D_{\uparrow v}| \geq \frac{\log(8\rho/\epsilon)}{r}|J|$, then selecting γ so that $|J \cap D_{\uparrow v}| = \frac{16\gamma\log(8\rho/\epsilon)}{\sqrt{\ell}}|J|$ suffices because at least $\gamma\rho n \geq \frac{\sqrt{\ell}}{16r}\rho n$ vertices are removed from the associated Y , which is a subset of C , and $|C| < \rho n$. Otherwise, use $\gamma = \frac{\sqrt{\ell}}{16r\log(8\rho/\epsilon)}$ and observe that at least 1 vertex is removed from every set Y since v was selected to be the lowest degree such vertex. Hence, $|C \setminus D_{\uparrow v}| < \left(1 - \frac{\sqrt{\ell}}{16r\log(8\rho/\epsilon)}\right)\rho n = (1 - \gamma)\rho n$.

Now we prove the desired bound on $|E(Y, D \setminus Y)|$ for any $Y \subseteq C$ with $|Y| = \left(1 - \frac{\sqrt{\ell}}{16r\log(8\rho/\epsilon)}\right)\rho n$. To start, we claim that $|E(D \setminus Y)| \leq \frac{64\log^2(8\rho/\epsilon)\epsilon r^2}{\ell\rho^2}|D \setminus Y|^2$. To show this, suppose not and find a contradiction. Using Lemma 6.4 (with $U = W = D \setminus Y$), there exists a d satisfying $\frac{32\log^2(8\rho/\epsilon)\epsilon r^2}{\ell\rho^2}|D \setminus Y| \leq d \leq |D \setminus Y|$ such that at least $\frac{16\log(8\rho/\epsilon)\epsilon r^2}{\ell\rho^2 d}|D \setminus Y|^2$ vertices in $D \setminus Y$ have degree larger than d in $G[D \setminus Y]$ (which also means they have degree larger than d in $G[D]$). Again using the first claim (since d is sufficiently large), we find that at most $\frac{2\epsilon r|J||D \setminus Y|}{d\ell\rho^2}$ vertices in J have more than d neighbours in D .

Let v be the vertex in J having the highest number of neighbours in D that is less than

d. Then $|D_{\uparrow v} \cap J| \leq \frac{2\epsilon r |J| |D \setminus Y|}{d \ell \rho^2}$, and we find that

$$\frac{|(D \setminus Y) \cap D_{\uparrow v}|}{\max(|D_{\uparrow v} \cap J|, \frac{\epsilon}{\sqrt{\ell} \rho^2} |J|)} \geq \frac{\frac{16 \log(8\rho/\epsilon) \epsilon r^2}{\ell \rho^2 d} |D \setminus Y|^2}{\max(\frac{2\epsilon r |J| |D \setminus Y|}{d \ell \rho^2}, \frac{\epsilon}{\sqrt{\ell} \rho^2} |J|)} > \frac{r |D \setminus Y|}{|J|},$$

where we have to use that $r > \frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)}$ and $|D \setminus Y| \geq d$, which is a contradiction with Eq. (6.4). Hence $|E(D \setminus Y)| \leq \frac{64 \log^2(8\rho/\epsilon) \epsilon r^2}{\ell \rho^2} |D \setminus Y|^2$.

Now, to show the lower bound on $|E(Y, D \setminus Y)|$, let R be a random set of size $\frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon) r} \rho n$ vertices from $D \setminus Y$. We will count the number of edges in $G[R \cup Y]$ in expectation, and use this to get the desired bound on $|E(Y, D \setminus Y)|$. Using the upper bound on $|E(D \setminus Y)|$, the expected number of edges in $G[R]$ is at most $\frac{64 \log^2(8\rho/\epsilon) \epsilon r^2}{\ell \rho^2} |R|^2 \leq \epsilon n^2/4$. The expected number of edges between R and Y is $|E(Y, D \setminus Y)| \cdot \frac{|R|}{|D \setminus Y|} = |E(Y, D \setminus Y)| \cdot \frac{\sqrt{\ell} \rho n}{16 \log(8\rho/\epsilon) r |D \setminus Y|}$. Finally, using that the number of edges in $G[Y]$ is at most $\epsilon n^2/4$, then the expected number of edges in $G[R \cup Y]$ is at most $\epsilon n^2/4 + \epsilon n^2/4 + |E(Y, D \setminus Y)| \cdot \frac{\sqrt{\ell} \rho n}{16 \log(8\rho/\epsilon) r |D \setminus Y|}$. Since R was selected randomly there exists a set R for which there are at most that many edges in $G[R \cup Y]$, and since $|Y \cup R| = \rho n$ and G is ϵ -far from having a ρn independent set, then $|E(Y, D \setminus Y)| > \frac{4 \log(8\rho/\epsilon) r \epsilon}{\sqrt{\ell} \rho^2} \rho n |D \setminus Y|$. $\square$

6.6 Proof of Graph Container Lemma

In this section we complete the proof of Lemma 6.2, restated below.

Lemma 6.2. *Let $\epsilon < \rho^2/2$ and let $\ell \geq 1024 \log^4(1/\epsilon)$. If $G = (V, E)$ is ϵ -far from having a ρn independent set then there exists a set of fingerprints $\mathcal{F} \subseteq \mathcal{F}(V)$ and a function $C : \mathcal{F} \rightarrow 2^V$ that satisfies the following:*

1. *For any $(F, R) \in \mathcal{F}$, $|C(F, R)| \leq \left(1 - \frac{\epsilon}{2\rho^2}\right) \rho n$. Further, letting $\alpha_{(F, R)}$ denote the value in $[\frac{\epsilon}{2\rho^2}, 1]$ that satisfies $|C(F, R)| = (1 - \alpha_{(F, R)}) \rho n$, then*
2. *For any $(F, R) \in \mathcal{F}$, $|F| \leq \frac{32\alpha_{(F, R)} \rho^2 \log^2(8\rho/\epsilon)}{\epsilon}$, and*
3. *For any $J \subseteq V$ for which $G[J]$ has fewer than $\frac{\epsilon}{\ell \rho^2} |J|^2$ edges, there exists $(F, R) \in \mathcal{F}(J) \cap \mathcal{F}$ such that $|J \cap C(F, R)| \geq \left(1 - \frac{32\alpha_{(F, R)} \log^2(8\rho/\epsilon)}{\sqrt{\ell}}\right) |J|$.*

Proof. Let J be any subset of V for which $G[J]$ has fewer than $\frac{\epsilon}{\ell\rho^2}|J|^2$ edges. We will construct a fingerprint (F, R) that, along with container function [Algorithm 5](#), satisfies the conclusions of the lemma for set J .

Let F be the fingerprint sequence constructed by [Algorithm 4](#), and let $C_{|F|}$ be the final container of [Algorithm 4](#) run on J . First observe that since $G[C_{|F|}]$ has at most $\epsilon n^2/4$ edges then $|C_{|F|}| < \left(1 - \frac{\epsilon}{2\rho^2}\right)\rho n$ because otherwise by adding arbitrary vertices to $C_{|F|}$ until $|C_F| = \rho n$ (so at most $\frac{\epsilon}{2\rho^2}\rho n$ vertices added) we would find an induced subgraph on at least ρn vertices that has fewer than ϵn^2 edges, a contradiction. Hence, let $|C_{|F|}| = (1 - \alpha)\rho n$, where $\frac{\epsilon}{2\rho^2} \leq \alpha \leq 1$. By [Lemma 6.5](#), $|C_{|F|} \cap J| \geq \left(1 - \frac{16\log^2(8\rho/\epsilon)}{\sqrt{\ell}}\right)|J|$ and $|F| \leq \frac{16\rho^2\log^2(8\rho/\epsilon)}{\epsilon}$. Hence, if $\alpha \geq \frac{1}{2}$ then set $R = \perp$ and observe that the container function [Algorithm 5](#) outputs $C_{|F|}$ on input (F, R) and satisfies the conclusions of the lemma for set J . For the remainder of the proof we consider the case that $\alpha < \frac{1}{2}$.

Let t^* be the smallest index for which $|C_{t^*}| \leq \rho n$. Break up the procedure into two parts: when $t \leq t^*$ and when $t > t^*$. First, for $t \leq t^*$ we are going to argue about the step of the procedure that reaches the worst (smallest) ratio. Let r be the smallest value such that there exists a step $t \leq t^*$ with

$$\text{MRR}_G(J, C_{t-1}) = \frac{r(|C_{t-1}| - (1 - 2\alpha)\rho n)}{|J|}. \quad (6.5)$$

We claim that $t^* \leq \frac{\sqrt{\ell}\rho^2\log(2\rho/\epsilon)}{\epsilon} \frac{\log(2\rho/\epsilon)}{r}$ and $|C_{t^*} \cap J| \geq \left(1 - \frac{\log(2\rho/\epsilon)}{r}\right)|J|$. To show this, for each step $t \leq t^*$, let $\beta_t|J|$ be the number of vertices of J that are removed when constructing C_t from C_{t-1} . Then, based on the minimality of r , we have that

$$|C_{t-1}| - |C_t| \geq \max\left(\beta_t|J|, \frac{\epsilon}{\sqrt{\ell}\rho^2}|J|\right) \frac{r(|C_{t-1}| - (1 - 2\alpha)\rho n)}{|J|},$$

and by rearranging and subtracting $(1 - 2\alpha)\rho n$ from both sides, this can be rewritten as

$$|C_t| - (1 - 2\alpha)\rho n \leq \left(1 - r \max\left(\beta_t, \frac{\epsilon}{\sqrt{\ell}\rho^2}\right)\right) (|C_{t-1}| - (1 - 2\alpha)\rho n).$$

Since this applies for all $t = 1, \dots, t^*$, then

$$|C_{t^*}| - (1 - 2\alpha)\rho n \leq \prod_{t=1}^{t^*} \left(1 - r \max\left(\beta_t, \frac{\epsilon}{\sqrt{\ell}\rho^2}\right)\right) \cdot (|C_0| - (1 - 2\alpha)\rho n).$$

Since $|C_{t^*}| \geq (1 - \alpha)\rho n$ then $|C_{t^*}| - (1 - 2\alpha)\rho n \geq \alpha\rho n \geq \frac{\epsilon}{2\rho}n$. Also, since $\alpha < 1/2$ then $|C_0| - (1 - 2\alpha)\rho n < n$. Substituting in, and using that $(1 - x) \leq e^{-x}$, we find that

$$\frac{\epsilon}{2\rho} \leq \exp \left(-r \sum_{t=1}^{t^*} \max \left(\beta_t, \frac{\epsilon}{\sqrt{\ell}\rho^2} \right) \right). \quad (6.6)$$

Using that the $\max(\cdot)$ in [Eq. \(6.6\)](#) is at least as large as the first operand, we have that $\sum_{t=1}^{t^*} \beta_t \leq \frac{\ln(2\rho/\epsilon)}{r} \leq \frac{\log(2\rho/\epsilon)}{r}$. Similarly, using that the $\max(\cdot)$ in [Eq. \(6.6\)](#) is at least as large as the second operand, then $t^* \leq \frac{\sqrt{\ell}\rho^2 \ln(2\rho/\epsilon)}{r\epsilon} \leq \frac{\sqrt{\ell}\rho^2 \log(2\rho/\epsilon)}{r\epsilon}$.

For the remaining steps $t > t^*$, by [Lemma 6.5](#) we have that

$$|C_{t-1}| - |C_t| \geq \frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)} \frac{\max \left(\beta_t |J|, \frac{\epsilon}{\sqrt{\ell}\rho^2} |J| \right)}{|J|} \rho n.$$

Summing up over all $t > t^*$ we find

$$|C_{t^*}| - |C_{|F|}| \geq \sum_{t=t^*+1}^{|F|} \frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)} \frac{\max \left(\beta_t |J|, \frac{\epsilon}{\sqrt{\ell}\rho^2} |J| \right)}{|J|} \rho n.$$

Since $|C_{t^*}| \leq \rho n$ and $|C_{|F|}| \geq (1 - \alpha)\rho n$, lower bounding the max by the first operand gives $\sum_{t=t^*+1}^{|F|} \beta_t \leq \frac{16\alpha \log(8\rho/\epsilon)}{\sqrt{\ell}}$, and lower bounding the max by the second operand gives $|F| - t^* \leq \frac{16\alpha \log(8\rho/\epsilon)\rho^2}{\epsilon}$. Combining, we have that $\sum_{t=1}^{|F|} \beta_t \leq \frac{\log(2\rho/\epsilon)}{r} + \frac{16\alpha \log(8\rho/\epsilon)}{\sqrt{\ell}}$ and $|F| \leq \frac{\sqrt{\ell}\rho^2 \log(2\rho/\epsilon)}{r\epsilon} + \frac{16\alpha\rho^2 \log(8\rho/\epsilon)}{\epsilon}$.

Now, if $\alpha > \frac{\sqrt{\ell}}{32r \log(8\rho/\epsilon)}$ then set $R = \perp$ and again observe that the container function [Algorithm 5](#) outputs $C_{|F|}$ on input (F, R) and satisfies the conclusions of the lemma for set J . Otherwise, let t be the step of the procedure that achieves [Eq. \(6.5\)](#) described above. Using that $r > \frac{\sqrt{\ell}}{16 \log(8\rho/\epsilon)}$ (by [Lemma 6.5](#)), apply [Lemma 6.6](#) with $C = C_{|F|}$ and $D = C_{t-1}$ to find that there exists a vertex $v \in J$ and $\gamma \geq \frac{\sqrt{\ell}}{16r \log(8\rho/\epsilon)}$ such that $|C_{|F|} \setminus (C_{t-1})_{\uparrow v}| \leq (1 - \gamma)\rho n$ and $|J \cap (C_{t-1})_{\uparrow v}| \leq \frac{16\gamma \log^2(8\rho/\epsilon)}{\sqrt{\ell}} |J|$. Hence, the container $C_{|F|} \setminus (C_{t-1})_{\uparrow v}$ satisfies the lemma conclusions for J because

$$\begin{aligned} |(C_{|F|} \setminus (C_{t-1})_{\uparrow v}) \cap J| &\geq \left(1 - \frac{\log(2\rho/\epsilon)}{r} - \frac{16\alpha \log(8\rho/\epsilon)}{\sqrt{\ell}} - \frac{16\gamma \log^2(8\rho/\epsilon)}{\sqrt{\ell}} \right) |J| \\ &\geq \left(1 - \frac{32\gamma \log^2(8\rho/\epsilon)}{\sqrt{\ell}} \right) |J| \end{aligned}$$

and $|F| \leq \frac{16\gamma\rho^2 \log^2(8\rho/\epsilon)}{\epsilon}$. Hence, set $R = (t-1, v)$, and observe that the container function [Algorithm 5](#) outputs $C_{|F|} \setminus (C_{t-1})_{\uparrow v}$ on input (F, R) as desired.

Construct the collection $\mathcal{F}$ by applying the above process to construct a fingerprint (F, R) for every $J \subseteq V$ for which $G[J]$ has fewer than $\frac{\epsilon}{\ell\rho^2}|J|^2$ edges. This completes the proof of the lemma. $\square$

Remark 6.7. [Lemma 6.2](#) provides a collection of containers for sparse subgraphs $G[J]$ with fewer than $\frac{\epsilon}{\ell\rho^2}|J|^2$ edges for $\ell = \Omega(\log^4(1/\epsilon))$, and, as seen in [Theorem 1.10](#), this implies a (ϵ', ϵ) -tolerant tester with $\epsilon' \sim \frac{\epsilon}{\log^4(1/\epsilon)}$. It is desirable to have an efficient fully tolerant tester, which is a tester that applies for any $\epsilon' < \epsilon$, because it implies an efficient distance approximation algorithm. Using our approach, this would require a graph container lemma that applies for sparse subgraphs $G[J]$ with fewer than $\frac{\epsilon}{\ell\rho^2}|J|^2$ edges for any $\ell > 1$.

While we do not try to optimize the logarithmic dependence in our results, we note here that there is a simple example that demonstrates the challenge with removing the log factors completely in our approach. In particular, consider a graph $G = (V, E)$ formed by planting a $\rho n/2$ independent set I in the random graph $G(n, \frac{8\epsilon}{\rho^2})$. Further, select a set $J \subseteq I$ of size $\frac{\rho n}{100}$ (although the size does not matter) and plant a random subgraph $G(|J|, \frac{\epsilon}{\rho^2})$. Finally, for each vertex v in J , select $\frac{8\epsilon}{\rho^2}n$ random vertices in $V \setminus N(v)$ and add edges between v and those vertices so that the vertices in J have the highest degree in the graph. First observe that this graph is ϵ -far from ρ -INDEPSET. Now, consider running [Algorithm 4](#) on J and G . With high probability, at each step the vertices in J have the highest degree in the graph, and so the only option is to remove neighbours. However, doing this removes a $\frac{16\epsilon}{\rho^2}$ fraction of the container in each step, and so it requires at least roughly $\frac{\rho^2}{16\epsilon} \log(1/\rho)$ steps before the container is of size less than ρn . However, in doing so we remove at least roughly $\frac{\epsilon}{\rho^2}$ fraction of J in each step, and so after roughly $\frac{\rho^2}{16\epsilon} \log(1/\rho)$ steps only roughly $\rho^{1/16}$ of J is still contained in the container, which in general may be much smaller than $1/2$. In order to guarantee that the final container contains more of J we would need $G[J]$ to have edge density less than roughly $\frac{\epsilon}{\rho^2 \log(1/\rho)}$.

6.7 A Tolerant Tester for the Independent Set Property

In this section we prove [Theorem 1.10](#) using [Lemma 6.2](#). The argument is roughly as follows. The tester takes a random sample S of $s \sim \frac{\rho^3 \text{polylog}(1/\epsilon)}{\epsilon^2}$ vertices and accepts if and

only if S contains a set J of size ρs such that $G[J]$ has at most roughly $\frac{\epsilon}{\text{polylog}(1/\epsilon)} s^2$ edges. The challenging part is showing that the algorithm rejects instances that are ϵ -far from ρ -INDEPSET, and for this we use Lemma 6.2. By Lemma 6.2, if G is ϵ -far then there is a relatively small collection of fingerprints, and associated containers, such such that for any sparse set J of size ρs , there exists a fingerprint whose vertices all come from J , such that J is mostly contained in the container associated with the fingerprint. In particular, as long as we select the parameter ℓ in Lemma 6.2 to be sufficiently large, the amount of J that is contained in the container is strictly more than the expected number of vertices that would be drawn from the container in a random sample S . Using a Chernoff bound we can upper bound the probability that S contains a sufficiently high number of vertices from any one container, then taking a union bound over all possible fingerprints, and associated containers, gives an upper bound on the probability that S contains a sparse subgraph on ρs vertices.

The proof of Theorem 1.10 follows a similar union bound argument used in the proof of Theorem 1.3, but using Lemma 6.2 instead of a graph container lemma for independent sets. The main difference is that instead of upper bounding the probability of finding ρs vertices from a container of size less than ρn , we upper bound the probability of finding at least $(1 - \alpha/2)\rho s$ vertices from a container of size $(1 - \alpha)\rho n$.

Theorem 1.10. *There exists a constant c such that for any $\epsilon > 0$ the sample complexity of $\left(\frac{\epsilon}{c \log^4(1/\epsilon)}, \epsilon\right)$ -tolerant testing the ρ -INDEPSET property is $\tilde{O}(\rho^3/\epsilon^2)$.*

Proof. Let c_1 be a sufficiently large constant. On input G , we want to distinguish between the case that G has a set $U \subseteq V$ of size ρn such that $G[U]$ has fewer than $\frac{\epsilon}{c_1 \log^4(1/\epsilon)} n^2$ edges, from the case that for every ρn set $U \subseteq V$, $G[U]$ has at least ϵn^2 edges. To do this, the tester takes a uniformly random set S of $s = c_2 \frac{\rho^3 \log^5(1/\epsilon)}{\epsilon^2}$ vertices without replacement, inspects all possible edges in $G[S]$, and accepts if and only if there is a set $J \subset S$ of size ρs such that $G[J]$ has at most $2 \frac{\epsilon}{c_1 \log^4(1/\epsilon)} s^2$ edges, where c_2 is a sufficiently large constant.²

First, we wish to show that the algorithm accepts with probability at least $1/4$ when G contains a sparse subgraph of size ρn . In this case, let $U \subseteq V$ be a set of size ρn such that $G[U]$ has fewer than $\frac{\epsilon}{c_1 \log^4(1/\epsilon)} n^2$ edges. The probability that S contains ρs vertices J that form a sparse subgraph $G[J]$ is at least the probability that S contains ρs vertices J from U that form a sparse subgraph. First observe that with probability at least $\frac{1}{2}$, S contains at least ρs vertices from U since the number of vertices drawn from U follows a hypergeometric

²Here and throughout we ignore integer rounding issues as they do not affect the asymptotics of the results.

distribution, and the median of the distribution is at least ρs [Neu70, KB80]. Next, observe that if J is a random subset of U of size ρs , then the expected number of edges in $G[J]$ is at most $\frac{\epsilon}{c_1 \log^4(1/\epsilon)} n^2 \frac{|J|^2}{\rho^2 n^2} = \frac{\epsilon}{c_1 \log^4(1/\epsilon)} s^2$. Hence, using Markov's Inequality, for such a random subset $J \subseteq U$ the probability that $G[J]$ has more than $2 \cdot \frac{\epsilon}{c_1 \log^4(1/\epsilon)} s^2$ edges is at most $1/2$. In other words, the probability that S contains ρs vertices J from U for which $G[J]$ has at most $2 \cdot \frac{\epsilon}{c_1 \log^4(1/\epsilon)} s^2$ edges, given that it contains at least ρs vertices from U , is at least $\frac{1}{2}$. Combining, we find that the probability that S contains at least ρs vertices from U and S contains ρs vertices J from U for which $G[J]$ has at most $2 \cdot \frac{\epsilon}{c_1 \log^4(1/\epsilon)} s^2$ edges, is at least $\frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$.

Now, consider the case that G is ϵ -far from having a ρn independent set. We wish to show that the algorithm accepts with only very small probability. First, let $\ell = \frac{c_1 \log^4(1/\epsilon)}{2}$. Since c_1 is a sufficiently large constant, then, by Lemma 6.2, there exists an absolute constant c_3 , a collection $\mathcal{F}$ of fingerprints, and a container function $C : \mathcal{F} \rightarrow 2^V$ such that for any $J \subseteq V$ of size ρs for which $G[J]$ has fewer than $\frac{\epsilon}{\ell^2} |J|^2 = \frac{\epsilon}{\ell} s^2$ edges there exists $(F, R) \in \mathcal{F}(J) \cap \mathcal{F}$ and α such that $|C(F, R)| = (1 - \alpha)\rho n$, $|F| \leq \frac{c_3 \alpha \rho^2 \log^2(1/\epsilon)}{\epsilon}$ and $|J \cap C(F, R)| \geq (1 - \alpha/2) |J|$. This implies that for any such sparse subgraph $G[J]$, there exists $(F, R) \in \mathcal{F}(J) \cap \mathcal{F}$ such that

$$\frac{|J \cap C(F, R)|}{|J|} - \frac{|C(F, R)|}{\rho n} \geq \frac{\epsilon}{2c_3 \rho^2 \log^2(1/\epsilon)} \cdot |F|.$$

Hence, the probability that S contains such a sparse subgraph $G[J]$ is at most

$$\sum_{(F, R) \in \mathcal{F}} \Pr \left[(F, R) \in \mathcal{F}(S) \text{ and } |S \cap C(F, R)| \geq \left(\frac{\epsilon}{2c_3 \rho^2 \log^2(1/\epsilon)} \cdot |F| + \frac{|C(F, R)|}{\rho n} \right) \rho s \right]. \quad (6.7)$$

Now, for any $(F, R) \in \mathcal{F}$, let F_{vert} be the set of vertices appearing in the sequence F and the revision R , and observe that $(F, R) \in \mathcal{F}(S)$ implies that $F_{\text{vert}} \subseteq S$. Then, by breaking up the sum in Eq. (6.7) by fingerprints (F, R) for which the size of F_{vert} is $f = 1, \dots, f_{\max} = \frac{c_3 \rho^2 \log^2(1/\epsilon)}{\epsilon} + 1$, we can upper bound the probability by

$$\sum_{f=1}^{f_{\max}} \sum_{\substack{(F, R) \in \mathcal{F} \\ |F_{\text{vert}}|=f}} \Pr[F_{\text{vert}} \subseteq S] \cdot \Pr \left[|S \cap C(F, R)| \geq \left(\frac{\epsilon}{2c_3 \rho^2 \log^2(1/\epsilon)} \cdot |F| + \frac{|C(F, R)|}{\rho n} \right) \rho s \mid F_{\text{vert}} \subseteq S \right].$$

Now, for any (F, R) , if $|F_{\text{vert}}| = f$ then

$$\Pr[F_{\text{vert}} \subseteq S] = \frac{\binom{n-f}{s-f}}{\binom{n}{s}} = \frac{\binom{s}{f}}{\binom{n}{f}},$$

because there are at most $\binom{n}{s}$ ways to choose S , and $\binom{n-f}{s-f}$ ways to choose an S that includes F_{vert} . Next, we can upper bound the probability that S contains at least $\left(\frac{\epsilon}{2c_3\rho^2 \log^2(1/\epsilon)} \cdot |F| + \frac{|C(F,R)|}{\rho n}\right) \rho s$ vertices from $C(F, R)$, conditioned on the fact that S contains F_{vert} , using a Chernoff bound. In particular, let X be a random variable to denote the number of vertices among the other $s - f$ sampled vertices that belong to $C(F, R)$, and so $E[X] = \frac{|C(F,R)|}{n}(s - f) < \frac{|C(F,R)|}{\rho n} \rho s$. By the Chernoff bound in [Lemma 2.2](#), the probability that S contains at least $\left(\frac{\epsilon}{2c_3\rho^2 \log^2(1/\epsilon)} \cdot |F| + \frac{|C(F,R)|}{\rho n}\right) \rho s - f$ vertices from $C(F, R)$ in the other $s - f$ vertices drawn to form S is

$$\begin{aligned} & \Pr \left[X \geq \left(\frac{\epsilon}{2c_3\rho^2 \log^2(1/\epsilon)} \cdot |F| + \frac{|C(F,R)|}{\rho n} \right) \rho s - f \right] \\ & \leq \exp \left(- \frac{\left(\frac{\epsilon}{2c_3\rho^2 \log^2(1/\epsilon)} \cdot |F| \cdot \rho s - f \right)^2}{2\rho s} \right) \\ & \leq \exp \left(- \frac{\epsilon^2 |F|^2 s}{32c_3^2 \rho^3 \log^4(1/\epsilon)} \right), \end{aligned}$$

where the last inequality uses that $f \leq |F| + 1$ and $\frac{\epsilon}{4c_3\rho^2 \log^2(1/\epsilon)} \rho s \geq 1$ because s is sufficiently large.

Substituting back into the probability bound above, the probability that S contains a ρs set J for which $G[J]$ has fewer than $\frac{\epsilon}{\ell} s^2$ edges is at most

$$\begin{aligned} & \sum_{f=1}^{f_{\max}} \sum_{\substack{(F,R) \in \mathcal{F} \\ |F_{\text{vert}}|=f}} \frac{\binom{s}{f}}{\binom{n}{f}} \exp \left(- \frac{\epsilon^2 |F|^2 s}{32c_3^2 \rho^3 \log^4(1/\epsilon)} \right) \\ & \leq \sum_{f=1}^{f_{\max}} \frac{1}{\binom{n}{f}} \sum_{\substack{(F,R) \in \mathcal{F} \\ |F_{\text{vert}}|=f}} \exp \left(f \log s - \frac{\epsilon^2 |F|^2 s}{32c_3^2 \rho^3 \log^4(1/\epsilon)} \right), \end{aligned}$$

where the inequality uses the upper bound $\binom{s}{f} \leq s^f \leq \exp(f \log(s))$. Hence, since $s = c_2 \frac{\rho^3 \log^5(1/\epsilon)}{\epsilon^2}$ for a sufficiently large constant c_2 , and $f \leq |F| + 1$, this can be upper bounded

by

$$\sum_{f=1}^{f_{\max}} \frac{1}{\binom{n}{f}} \sum_{\substack{(F,R) \in \mathcal{F} \\ |F_{\text{vert}}|=f}} \exp(-4|F| \log(s)).$$

We again split the sum up into fingerprints (F, R) such that $|F| = k$, for $k = f, \dots, f_{\max}$, and use that there are at most $\binom{n}{f} 2^k f^k \cdot f \cdot k \leq \binom{n}{f} \exp(2k \log(k))$ possible (F, R) pairs such that $|F_{\text{vert}}| = f$ and $|F| = k$ to upper bound the probability by

$$\begin{aligned} \sum_{f=1}^{f_{\max}} \frac{1}{\binom{n}{f}} \sum_{k=f}^{f_{\max}} \sum_{\substack{(F,R) \in \mathcal{F} \\ |F_{\text{vert}}|=f \\ |F|=k}} \exp(-4k \log(s)) &\leq \sum_{f=1}^{f_{\max}} \sum_{k=f}^{f_{\max}} \exp(2k \log(k) - 4k \log(s)) \\ &\leq \sum_{f=1}^{f_{\max}} \sum_{k=f}^{f_{\max}} \exp(-2k \log(s)). \end{aligned}$$

Finally, using that the sum is upper bounded by the case when $k = 1$, we find that the probability that S contains a ρs set J for which $G[J]$ has fewer than $\frac{\epsilon}{\ell} s^2$ edges is at most

$$f_{\max}^2 \exp(-2 \log(s)) \leq \exp(-\log(s)) \leq \frac{1}{s} \ll \frac{1}{8},$$

where the first inequality uses that $f_{\max}^2 \leq s$. □

Remark 6.8. The proof of [Theorem 1.10](#) shows that the tester accepts graphs that are $\tilde{\Omega}(\epsilon)$ -close to ρ -INDEPSET with probability at least $1/4$ and accepts graphs that are ϵ -far from ρ -INDEPSET with probability at most $1/8$. With standard error reduction techniques it is possible to boost the probabilities so that the algorithm answers correctly with probability at least $2/3$ without changing the asymptotic sample complexity.

6.8 Application to Non-Canonical Testers

In this section we prove [Corollary 1.11](#), restated below. The proof idea is the following. Similar to [Theorem 1.3](#) and [Theorem 1.10](#) the tester takes a random sample S of roughly ρ^3/ϵ^2 vertices. However, instead of querying every pair of vertices in the sample and checking if $G[S]$ has a ρs subgraph with a small number of edges, the tester queries roughly

$r \sim \frac{\rho}{\epsilon}|S|$ edge locations at random within $G[S]$ and accepts if and only if there exists a $\rho|S|$ set of vertices with a relatively small number of queried edges. Using [Theorem 1.10](#), if G is ϵ -far from ρ -INDEPSET then $G[S]$ is also roughly ϵ -far from ρ -INDEPSET. In other words, every $\rho|S|$ subgraph in $G[S]$ has a relatively high number of edges. Hence, by a simple union bound argument we can upper bound the probability that there exists a $\rho|S|$ set of vertices for which the number of queried edges of $G[S]$ is small.

Corollary 1.11. *There exists a constant c such that for any $\epsilon > 0$ the query complexity of $\left(\frac{\epsilon}{c \log^4(1/\epsilon)}, \epsilon\right)$ -tolerant testing the ρ -INDEPSET property is $\tilde{O}(\rho^4/\epsilon^3)$.*

Proof. Let c_1, c_2, c_3 be sufficiently large constants. On input G , we want to distinguish between the case that G has a set $U \subseteq V$ of size ρn such that $G[U]$ has fewer than $\frac{1}{16} \frac{\epsilon}{c_1 \log^4(1/\epsilon)} n^2$ edges, from the case that for every ρn set $U \subseteq V$, $G[U]$ has at least ϵn^2 edges. To do this, the tester takes a uniformly random set S of $s = c_2 \frac{\rho^3 \log^5(1/\epsilon)}{\epsilon^2}$ vertices without replacement, selects $r = c_3 \frac{\rho s \text{polylog}(1/\epsilon)}{\epsilon}$ random pairs of vertices in S and queries the adjacency matrix on the pairs, and accepts if and only if there is a set $J \subset S$ of size ρs such that the number of edges queried within $G[J]$ is at most $\frac{\epsilon}{c_1 \log^4(1/\epsilon)} \cdot r$. For ease of presentation let $\beta = \frac{\epsilon}{c_1 \log^4(1/\epsilon)}$.

First, we wish to show that the algorithm accepts with probability at least $1/4$ when G contains a sparse subgraph of size ρn . In this case, by [Theorem 1.10](#), with probability at least $1/4$ the sampled vertices S contain ρs vertices J for which $G[J]$ has at most $\frac{1}{8} \beta s^2$ edges. Conditioned on the case that such a set $J \subseteq S$ exists, the number of edges queried by the tester within $G[J]$ is at most $r \cdot \frac{\beta}{2}$ since each queried pair of vertices finds an edge within $G[J]$ with probability at most $4 \cdot \frac{\beta}{8}$. By a Markov bound, the probability that the number of edges queried by the tester within $G[J]$ is greater than βr is at most $\frac{1}{2}$, again conditioned on the case that S contain ρs vertices J for which $G[J]$ has at most $\frac{1}{8} \beta s^2$ edges. Hence, the probability that the number of edges queried by the tester within $G[J]$ is greater than βr is at most $1/8$.

In the case that G is ϵ -far from ρ -INDEPSET, by [Theorem 1.10](#), the probability that the sampled vertices S contain ρs vertices J for which $G[J]$ has at most $2\beta s^2$ edges is at most $1/s$. In such a case, for any set $J \subseteq S$ of ρs vertices, the expected number of edges in $G[J]$ queried by the tester is at most $4\beta r$. Then, by a Chernoff bound, the probability that the tester queries at most βr edges from $G[J]$ is at most $\exp(-\beta r/2)$. Finally, applying a union bound over all possible J of size ρs , the probability that there exists a set $J \subseteq S$ of

ρs vertices for which the tester queries at most $4\beta r$ edges in $G[J]$ is at most

$$\binom{s}{\rho s} \cdot \exp(-\beta r/2) \leq \exp(\rho s \ln(s) - \beta r/2),$$

and for the selection $r = c_3 \frac{\rho s \text{polylog}(1/\epsilon)}{\epsilon}$ this probability is sufficiently small. $\square$

Chapter 7

Future Directions

This thesis demonstrates that the container method provides a natural framework for studying graph property testing algorithms. By proving new graph and hypergraph container lemmas, we gave new upper bounds on the sample complexity of testing a number of properties including ρ -INDEPSET, k -COLOR, and (q, k) -SAT, and gave new upper bounds on the sample complexity of tolerant testing ρ -INDEPSET. The ideas in this thesis lead to a number of future directions. We begin with concrete questions about additional testing problems and progressively broaden the scope, concluding with the question of whether the applicability of the container method can be characterized in general.

7.1 Testing and Tolerant Testing more Properties

A natural first question is whether the container method framework can be applied to other graph and hypergraph properties, both in the standard and tolerant testing settings. Below we discuss several such properties for which the framework appears promising. In some cases, these directions have already led to subsequent work.

Testing the Independent Set Property in Hypergraphs. The ρ -INDEPSET problem in q -uniform hypergraphs is the problem of distinguishing between hypergraphs with a ρn independent set and hypergraphs for which every ρn induced subgraph has at least $\epsilon \binom{n}{q}$ edges. What is the sample complexity of testing the ρ -INDEPSET property in q -uniform hypergraphs? Following similar techniques to [FLS04], Langberg showed an upper bound of roughly $\frac{q^{O(q)} \rho^{2q}}{\epsilon^3}$ and a lower bound of roughly $\frac{\rho^{2q-1}}{\epsilon^2}$ [Lan04].

In subsequent work [GNS26], we apply the container method framework to give a new upper bound of $\tilde{O}(\frac{q^5 \rho^{2q-3}}{\epsilon^2})$ on the sample complexity of testing the ρ -INDEPSET property in q -uniform hypergraphs, achieving the optimal ϵ dependence and making an exponential improvement on the q dependence compared to [Lan04]. However there still remains a gap in the ρ and q dependence of our upper bound and Langberg’s lower bound. Can the container method approach seen in [GNS26] be strengthened to get the optimal bounds in ρ, q and ϵ ?

Testing More Graph Partition Properties. Our result on satisfiability testing immediately gives upper bounds on testing a class of graph partition properties known as semihomogeneous graph partition properties (Corollary 1.8). These are graph properties which correspond to partitioning a graph into parts where the density between parts is either 1, 0 or there are no restrictions (see Chapter 2 for definitions). There is a more general class of graph partition properties where the density between parts can be some arbitrary constant. These graph partition properties were originally studied by Goldreich, Goldwasser and Ron [GGR98], and more recently by Shapira and Stagni [SS24], and it is known that they are testable with sample complexity that is polynomial in $1/\epsilon$ and k , where k is the number of partitions of the property. However, the polynomial dependence is quite large, and it remains open what the optimal ϵ -dependence is on testing these partition properties. An interesting open problem is to determine whether the container method framework can be extended to this more general class of graph partition properties.

There are immediate challenges when applying the container method to partition properties with arbitrary density requirements between parts. As an example, consider the property of having a ρn induced subgraph with edge density less than $1/2$. This is a natural graph partition property, however the container ideas from this thesis do not directly apply because they are designed for independent sets (or sparse subgraphs) in graphs that are ϵ -far from having a large independent set. For the property of having a ρn subgraph with edge density less than $1/2$ we require containers for subgraphs with edge density less than $1/2$ in a graph that is ϵ -far from this property.

Tolerant Testing Bipartiteness and the Bogdanov Li Conjecture. Theorem 1.10 shows that $(\tilde{\Omega}(\epsilon), \epsilon)$ -tolerant testing ρ -INDEPSET has the same sample complexity, up to logarithmic factors, as the standard testing problem. Are there other graph properties for which tolerant testing is no harder than standard testing?

In general, any graph property that is testable with sample complexity $s(\epsilon)$ is (ϵ_1, ϵ_2) -tolerantly testable with sample complexity that is at most doubly exponential in $s(\epsilon_2 - \epsilon_1)$

[SKG24]. However it is not known if/when this doubly exponential loss is necessary.

One particular problem of interest is tolerant testing the bipartiteness problem. There are tight bounds of $\tilde{\Theta}(1/\epsilon)$ for the sample complexity of testing bipartiteness [AK02] in the non-tolerant setting. It has been conjectured by Bogdanov and Li that the sample complexity of $(\tilde{\Omega}(\epsilon), \epsilon)$ -tolerant testing the bipartiteness property has the same sample complexity of $\tilde{O}(1/\epsilon)$ [BL10]. It has been shown that for some restricted graphs the conjecture holds [GR10, BL10], but in general it still remains open.

This conjecture is especially interesting because of its implications on adaptive (non-canonical) bipartiteness testing. By a lower bound by Bogdanov and Trevisan, it is known that the canonical tester for bipartiteness achieves (asymptotically) the optimal query complexity of all non-adaptive testers [BT04]. However, it is not known whether adaptive algorithms can achieve a lower query complexity. Bogdanov and Trevisan show that if their conjecture on tolerant testing is true, then it implies an adaptive testing algorithm for bipartiteness with query complexity $O(1/\epsilon^c)$, with $c < 2$, which beats the query complexity of the canonical tester of $\tilde{O}(1/\epsilon^2)$ [BL10].

Given the applicability of the container method framework to testing k -COLOR and to tolerant testing ρ -INDEPSET, applying the container method to tolerant testing bipartiteness and k -colorability is a natural future direction.

Asymptotically Optimal Bounds. In addition to testing other properties, an equally important direction is to improve the quantitative bounds for properties studied in this thesis. Of particular interest are the k -COLOR and (q, k) -SAT properties. For k -COLOR, the best upper bound is $O(k \log(1/\epsilon)/\epsilon)$ [KS26] (our upper bound in Corollary 1.6 gives $O(k \log^2(1/\epsilon)/\epsilon)$ and [KS26] recently removed one of the logarithmic factors), and the best lower bound is $\Omega(\log k/\epsilon)$ [KS26]. For non-constant k , there is roughly a factor k gap between the upper and lower bounds. In the regime when k is a constant (such as bipartiteness), there still remains a $\log(1/\epsilon)$ gap between upper and lower bounds. A similar gap exists for (q, k) -SAT in the k dependence between our upper bound of $O(kq^3 \log^2(1/\epsilon)/\epsilon)$ and the best lower bound of $\Omega(\log k/\epsilon)$, which again comes from [KS26]. Further, for satisfiability testing, it is not known whether the polynomial dependence in q is necessary.

In the case of testing ρ -INDEPSET there is also a logarithmic gap between the upper bound in Theorem 1.3 of $O(\rho^3 \log^3(1/\epsilon)/\epsilon^2)$ and the best known lower bound of $\Omega(\rho^3/\epsilon^2)$ [FLS04]. We do not know whether the container method approach can be used to close this gap.

7.2 Non-Canonical Testing

Much of the work in this thesis focused on *canonical* testers. It is known that there is at most a quadratic gap between the query complexity of canonical and non-canonical testers (even with adaptivity) [GT03], but for many properties it is not known whether this quadratic gap exists. Does the container method framework extend to analyzing non-canonical testers?

Using our result on tolerant testing ρ -INDEPSET, Corollary 1.11 shows that there is a gap in the query complexity of canonical testers and non-canonical testers for the ρ -INDEPSET property. We prove Corollary 1.11 by analyzing a natural non-canonical tester that takes a random sample S of vertices, queries the adjacency matrix on a number of random pairs of vertices from the sample, and accepts if and only if the queried edges are consistent with $G[S]$ satisfying the ρ -INDEPSET property. Does this approach apply to other graph properties?

In a slightly different direction, it may be possible to use the container method framework to study other non-canonical testers directly. For example, in prior work we demonstrated a similar gap in the query complexity of non-canonical and canonical testers for ρ -INDEPSET (although with a weaker upper bound on the query complexity) by analyzing a different non-canonical tester directly [BS24]. Very briefly, the tester for ρ -INDEPSET takes two vertex samples R and S , queries all pairs of vertices in R and all pairs with one endpoint in R and one point in S , and checks if there is a large set of vertices consistent with $G[R \cup S]$ satisfying ρ -INDEPSET. In that work we introduced an object called an *independent star*, which is a combination of two sets of vertices I, J , where I is an independent set, and there are no edges between I and J in the graph (see Fig. 7.1). Analyzing this tester corresponds to analyzing the probability that the tester finds an independent set star with $I \in R$ and $J \in S$. In order to analyze this tester we introduced a new type of container lemma for independent set stars.

Is it possible to study other non-canonical testers directly with this type of container approach? Doing so may require container lemmas for new types of objects instead of independent sets (as was seen with independent set stars).

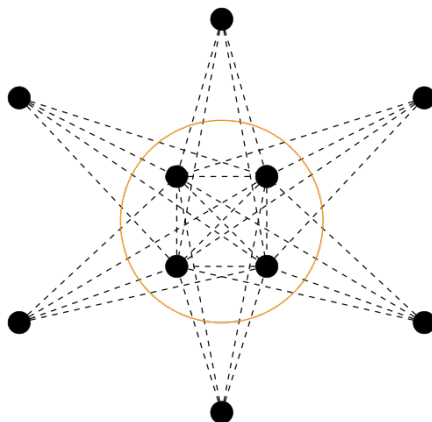


Figure 7.1: An example of an independent set star. In the center is a core of vertices forming an independent. The vertices outside of the core have no edges to the core. There are no requirements on the edges between vertices on the outside.

7.3 Combinatorial Applications of our new Container Lemmas

The graph and hypergraph container methods have many applications in extremal combinatorics. While the intended application of our new container lemmas was for property testing, we expect they will have other combinatorial applications. To demonstrate this, in [Set25] we apply Lemma 6.2 to generalize one of the classic applications of the container method, that of counting independent sets in regular graphs, to counting sparse subgraphs in regular graphs. The main ideas are outlined below. Are there other combinatorial applications of our new container lemmas?

Counting Sparse Subgraphs in Regular Graphs. The problem of counting independent sets in d -regular graphs is very well understood. Proving a conjecture of Granville, Alon first showed that the number of independent sets in a d -regular graph is at most $2^{\frac{n}{2}(1+O(d^{-0.1}))}$, and conjectured that the graph with $\frac{n}{2d}$ disjoint copies of $K_{d,d}$, which has $(2^{d+1} - 1)^{\frac{n}{2d}}$ independent sets, has the maximum number of independent sets [Alo91]. Sapozhenko improved the upper bound by Alon to $2^{\frac{n}{2}(1+O(\sqrt{\log(d)/d}))}$ using the graph container method [Sap01] (see Section 2.2 for an outline of this application). Since then, Alon's conjecture has been proven, first by Kahn for bipartite graphs [Kah01], and then by Zhao for general graphs [Zha10].

But what can be said about the number of sparse subgraphs (of at most some edge density) in a d -regular graph? As far as we know, there have been no results studying this natural generalization of the problem. Using a similar argument to Sapozhenko [Sap01], but using Lemma 6.2 instead of a standard graph container lemma for independent sets, we can prove the following theorem, which applies to sparse subgraphs and not just independent sets.

Theorem 7.1 ([Set25]). *Let $G = (V, E)$ be a d -regular graph. There exists a constant c such that for any $k \geq c \log^9(n)$ the number of subsets $J \subseteq V$ such that $G[J]$ has edge density at most $\frac{1}{k} \frac{d}{n}$ is at most*

$$2^{\frac{n}{2} \left(1 + O\left(\frac{\log^3 n}{d}\right) + O\left(\frac{\log^3 n}{k^{1/3}}\right) \right)}.$$

A d -regular graph has edge density roughly $\frac{2d}{n}$, and using a Markov bound it is not hard to show that any d -regular graph has at least $\frac{1}{2} 2^n$ induced subgraphs with edge density less than $\frac{4d}{n}$. In other words, on one extreme Zhao's result [Zha10] says that any d -regular graph has fewer than roughly $2^{n/2}$ induced subgraphs with density 0 (i.e. independent sets), and on the other extreme we have that any d -regular graph has at least $\frac{1}{2} 2^n$ induced subgraphs with edge density less than $\frac{4d}{n}$. Theorem 7.1 says that for k and d at least $\text{polylog}(n)$, the number of sparse subgraphs with edge density $\frac{1}{k} \frac{d}{n}$ drops to nearly $2^{n/2}$.

Also, it can be shown via a simple counting argument that the graph with $\frac{n}{2d}$ disjoint copies of $K_{d,d}$ has at least $2^{\frac{n}{2} \left(1 + \frac{1}{2d} + \Omega\left(\frac{\log(k)}{k}\right) \right)}$ subgraphs with density at most $\frac{1}{k} \frac{d}{n}$ (see [Set25] for details). This lower bound demonstrates that there is some $\frac{n}{\text{poly}(k)}$ dependence in the exponent, but it remains open what the exact bound should be, and whether the graph with $\frac{n}{2d}$ disjoint copies of $K_{d,d}$ is the extremal graph.

7.4 Characterizations of Properties Testable with the Container Method

This thesis demonstrates that the container method provides a natural framework for studying a number of natural graph and hypergraph properties. In this chapter we outlined a number of other properties that may be well suited to the container method framework. More generally, is it possible to classify the class of properties for which the container method framework applies?

In recent work [CSY26], we demonstrate that there is a deeper connection between graph property testing and the container method. In particular, we show that for any hereditary graph property Π , Π is testable with one sided error and sample complexity s if and only if there exists a type of container lemma, parameterized by s , for covering Π -subgraphs in a graph that is ϵ -far from Π . (Note that a hereditary graph property is a property that is closed under taking induced subgraphs.)

To illustrate this, consider the k -colorability property, which is a hereditary property. Our container lemma for satisfiability, Lemma 5.1, applied to k -colorability roughly says the following: if $G = (V, E)$ is ϵ -far from k -COLOR then there exists a collection of containers such that for every $S \subseteq V$ for which $G[S]$ is in k -COLOR, there exists a container such that S is a subset of the container. This type of container lemma immediately implies a tester for k -COLOR with sample complexity that depends on the fingerprint and container sizes claimed by the container lemma. Our new result says that for any hereditary property Π , it is testable with sample complexity s if and only if this type of container lemma exists for Π , where s is a parameter in the container and fingerprint sizes.

This equivalence between testability and container lemmas is relevant to the open question asked by Alon, Fischer, Newman and Shapira regarding finding a quantitative characterization of the graph properties testable with certain sample complexity [AFNS09]. That work, along with the work of Alon and Shapira [AS08], characterizes the class of properties testable with two sided error and one sided error, respectively. However both results go through the regularity lemma, and so the resulting sample complexity may be as large as tower type. In [AFNS09] they asked whether there was a more *quantitative* version of these characterizations. Our result demonstrates that the hereditary graph properties that are testable with sample complexity s are the class of properties exhibiting container lemmas parameterized by s . Continuing in this direction, is it possible to show an equivalence between testability and the existence of container lemmas for all graph properties (not just hereditary properties)?

The results of this thesis demonstrate that the container method provides a powerful framework for analyzing property testing algorithms. Beyond giving new sample complexity bounds for several classic testing problems, the techniques developed here suggest new directions in tolerant testing, non-canonical testing, extremal combinatorics, and the quantitative characterization of the testable graph properties. It is our hope that these ideas will continue to broaden the role of the container method, both within property testing and beyond.

References

- [AFNS09] Noga Alon, Eldar Fischer, Ilan Newman, and Asaf Shapira. A combinatorial characterization of the testable graph properties: It’s all about regularity. volume 39, pages 143–167. SIAM, 2009.
- [AK02] Noga Alon and Michael Krivelevich. Testing k-colorability. *SIAM Journal on Discrete Mathematics*, 15(2):211–227, 2002.
- [AKS98] Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. *Random Structures & Algorithms*, 13(3-4):457–466, 1998.
- [Alo91] Noga Alon. Independent sets in regular graphs and sum-free subsets of finite groups. *Israel journal of mathematics*, 73(2):247–256, 1991.
- [AS03] Noga Alon and Asaf Shapira. Testing satisfiability. *Journal of Algorithms*, 47(2):87–103, 2003.
- [AS08] Noga Alon and Asaf Shapira. A characterization of the (natural) graph properties testable with one-sided error. *SIAM Journal on Computing*, 37(6):1703–1727, 2008.
- [BESS78] Béla Bollobás, Paul Erdős, Miklós Simonovits, and Endre Szemerédi. Extremal graphs without large forbidden subgraphs. *Annals of Discrete Mathematics*, 3:29–41, 1978.
- [BHK⁺19] Boaz Barak, Samuel Hopkins, Jonathan Kelner, Pravesh K Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. *SIAM Journal on Computing*, 48(2):687–735, 2019.
- [BL10] Andrej Bogdanov and Fan Li. A better tester for bipartiteness? arXiv preprint 1011.0531, November 2010.

- [BMS15] József Balogh, Robert Morris, and Wojciech Samotij. Independent sets in hypergraphs. *Journal of the American Mathematical Society*, 28(3):669–709, 2015.
- [BMS18] József Balogh, Robert Morris, and Wojciech Samotij. The method of hypergraph containers. In *Proceedings of the International Congress of Mathematicians: Rio de Janeiro 2018*, pages 3059–3092. World Scientific, 2018.
- [BS24] Eric Blais and Cameron Seth. New graph and hypergraph container lemmas with applications in property testing. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1793–1804, 2024.
- [BS25] Eric Blais and Cameron Seth. Testing graph properties with the container method. volume 54, pages 1626–1642. SIAM, 2025.
- [BT04] Andrej Bogdanov and Luca Trevisan. Lower bounds for testing bipartiteness in dense graphs. In *Proceedings of the 19th IEEE Annual Conference on Computational Complexity*, pages 75–81, 2004.
- [BY22] Arnab Bhattacharyya and Yuichi Yoshida. *Property Testing: Problems and Techniques*. Springer Nature, 2022.
- [CS01] Artur Czumaj and Christian Sohler. Testing hypergraph coloring. In *Automata, Languages and Programming: 28th International Colloquium, ICALP 2001 Crete, Greece, July 8–12, 2001 Proceedings 28*, pages 493–505. Springer, 2001.
- [CS26] Marcelo Campos and Wojciech Samotij. Towards an optimal hypergraph container lemma. *Combinatorica*, 46(3):24, 2026.
- [CSY26] Gaia Carenini, Cameron Seth, and Yuichi Yoshida. A quantitative container characterization of one-sided testability. *arXiv preprint arXiv:2608.01523*, 2026.
- [Die12] Reinhard Diestel. Graph theory, volume 173 of. *Graduate texts in mathematics*, 7, 2012.
- [FF05] Eldar Fischer and Lance Fortnow. Tolerant versus intolerant testing for boolean properties. In *20th Annual IEEE Conference on Computational Complexity (CCC’05)*, pages 135–140. IEEE, 2005.
- [FLS04] Uriel Feige, Michael Langberg, and Gideon Schechtman. Graphs with tiny vector chromatic numbers and huge chromatic numbers. *SIAM Journal on Computing*, 33(6):1338–1368, 2004.

- [FMS10] Eldar Fischer, Arie Matsliah, and Asaf Shapira. Approximate hypergraph partitioning and applications. *SIAM Journal on Computing*, 39(7):3155–3185, 2010.
- [FN07] Eldar Fischer and Ilan Newman. Testing versus estimation of graph properties. *SIAM Journal on Computing*, 37(2):482–501, 2007.
- [FR86] Peter Frankl and Vojtech Rödl. Large triangle-free subgraphs in graphs without k 4. *Graphs and Combinatorics*, 2(1):135–144, 1986.
- [FR21] Nimrod Fiat and Dana Ron. On efficient distance approximation for graph properties. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1618–1637. SIAM, 2021.
- [GGR98] Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998.
- [GNS26] Elena Grigorescu, Shreya Nasa, and Cameron Seth. Testing the independent set property in hypergraphs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2026)*, pages 73–1, 2026.
- [Gol17] Oded Goldreich. *Introduction to property testing*. Cambridge University Press, 2017.
- [GR10] Mira Gonen and Dana Ron. On the benefits of adaptivity in property testing of dense graphs. *Algorithmica*, 58(4):811–830, 2010.
- [GT03] Oded Goldreich and Luca Trevisan. Three theorems regarding testing graph properties. *Random Structures & Algorithms*, 23(1):23–57, 2003.
- [GW22] Oded Goldreich and Avi Wigderson. Robustly self-ordered graphs: Constructions and applications to property testing. *TheoretCS*, 1, 2022.
- [Has96] J Hastad. Clique is hard to approximate within $n^{1-\epsilon}$. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 627–627. IEEE Computer Society, 1996.
- [HLMZ26] Zicheng Han, Yupeng Lin, Jie Ma, and Xiande Zhang. A hypergraph container method on spread sat: Approximation and speedup. *arXiv preprint arXiv:2604.15031*, 2026.

- [JPP23] Matthew Jenssen, Will Perkins, and Aditya Potukuchi. Approximately counting independent sets in bipartite graphs via graph containers. *Random Structures & Algorithms*, 2023.
- [Kah01] Jeff Kahn. An entropy approach to the hard-core model on bipartite graphs. *Combinatorics, Probability and Computing*, 10(3):219–237, 2001.
- [Kar72] Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.
- [KB80] Rob Kaas and Jan M Buhrman. Mean, median and mode in binomial distributions. *Statistica Neerlandica*, 34(1):13–18, 1980.
- [KS26] Nick Kushnir and Asaf Shapira. Towards tight bounds for testing k -colorability. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2026)*, pages 35–1, 2026.
- [KW82] Daniel J. Kleitman and Kenneth J. Winston. On the number of graphs without 4-cycles. *Discrete Mathematics*, 41(2):167–172, 1982.
- [Lan04] Michael Langberg. Testing the independence number of hypergraphs. In *International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 405–416. Springer, 2004.
- [Man07] W. Mantel. Problem 28. *Wiskundige Opgaven*, 10:60–61, 1907.
- [Mul18] Wolfgang Mulzer. Five proofs of Chernoff’s bound with applications. *Bulletin of EATCS*, 1(124), 2018.
- [Nen24] Rajko Nenadov. Probabilistic hypergraph containers. *Israel Journal of Mathematics*, 261(2):879–897, 2024.
- [Nen25] Rajko Nenadov. Counting sparse induced subgraphs in locally dense graphs. *European Journal of Combinatorics*, 126:104125, 2025.
- [Neu70] Peter Neumann. Über den median einiger diskreter verteilungen und eine damit zusammenhängende monotone konvergenz. *Wissenschaftliche Zeitschrift der Technischen Universität Dresden*, 19:29–33, 1970.
- [NP25] Rajko Nenadov and Huy Tuan Pham. Short proof of the hypergraph container theorem. *Combinatorics, Probability and Computing*, 34(5):621–624, 2025.

- [NR18] Yonatan Nakar and Dana Ron. On the testability of graph partition properties. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018)*, page 13, 2018.
- [PRR06] Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012–1042, 2006.
- [RD85] Vojtěch Rödl and Richard A. Duke. On graphs with small subgraphs of large chromatic number. *Graphs and Combinatorics*, 1(1):91–96, 1985.
- [Ron10] Dana Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends® in Theoretical Computer Science*, 5(2):73–205, 2010.
- [Sam15] Wojciech Samotij. Counting independent sets in graphs. *European journal of combinatorics*, 48:5–18, 2015.
- [Sap01] Alexander Sapozhenko. On the number of independent sets in expanders. *Discrete Math. Appl*, 11(2):155–161, 2001.
- [Sap05] Alexander Sapozhenko. Systems of containers and enumeration problems. In *Stochastic Algorithms: Foundations and Applications (SAGA 2005)*, pages 1–13. Springer, 2005.
- [Sap08] Alexander A Sapozhenko. The cameron–erdős conjecture. *Discrete mathematics*, 308(19):4361–4369, 2008.
- [Set25] Cameron Seth. A tolerant independent set tester. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 1031–1042, 2025.
- [SKG24] Asaf Shapira, Nick Kushnir, and Lior Gishboliner. Testing versus estimation of graph properties, revisited. *Random Structures & Algorithms*, 2024.
- [Soh12] Christian Sohler. Almost optimal canonical property testers for satisfiability. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science*, pages 541–550, 2012.
- [SS24] Asaf Shapira and Henrique Stagni. A tight bound for testing partition properties. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 4305–4320. SIAM, 2024.

- [ST15] David Saxton and Andrew Thomason. Hypergraph containers. *Inventiones mathematicae*, 201(3):925–992, 2015.
- [Zam23] Or Zamir. Algorithmic applications of hypergraph and partition containers. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, 2023.
- [Zha10] Yufei Zhao. The number of independent sets in a regular graph. *Combinatorics, Probability and Computing*, 19(2):315–320, 2010.
- [Zuc07] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of computing*, 3(1):103–128, 2007.