博　士　論　文

Testable and Untestable Classes of First-Order Logic

（一階述語論理の検査可能・不可能なサブクラスについて）

北海道大学大学院情報科学研究科

Charles Jordan

Testable and Untestable Classes of First-Order Logic

Charles Jordan

Submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in the field of

Computer Science

at the

Graduate School of Information Science and Technology,

Hokkaido University

March 2012

Abstract

Property testing is essentially a kind of constant-time randomized approximation. Alon *et al.* [3] were the first to consider the idea of testing properties expressible in syntactic subclasses of first-order logic. They proved the testability of all properties of undirected, loop-free graphs expressible with quantifier prefix $\exists^*\forall^*$, and also that there exist untestable properties of undirected, loop-free graphs expressible with quantifier prefix $\forall^*\exists^*$.

In this dissertation, we continue the study of testing subclasses of first-order logic. In particular, we focus on the classification of prefix-vocabulary classes, or classes defined by quantifier prefix and vocabulary, according to their testability.

The main results are as follows. First, we develop a framework for relational property testing including variations corresponding to the different models considered in the literature for non-uniform hypergraph testing. We then use this framework to prove the following.

1. All (relational) properties expressible by formulae in Ackermann's class with equality ($[\exists^*\forall\exists^*, \mathit{all}\,]_=$) are *testable* in all of our models.

2. All (relational) properties expressible in Ramsey's class ($[\exists^*\forall^*, \mathit{all}\,]_=$) are *testable* in all of our models. This extends the result by Alon *et al.* [3] to the full class.

3. There exist graph properties expressible in the class $[\forall^3\exists, (0,1)]_=$ that are *untestable* in all of our models. This considerably sharpens the untestable class of Alon *et al.* [3].

4. There exist graph properties expressible in the class $[\forall\exists\forall, (0,1)]_=$ (the Kahr-Moore-Wang prefix) that are untestable in all of our models.

5. There exist graph properties expressible in the class $[\forall\exists\forall, (0,1)]$ (*without* equality) that are untestable in (at least) one of our models.

概要

近年は大規模なデータセットが増え、情報爆発の時代であると言われている。ある種のデータセットでは、その正確なサイズが不明であり、全てのデータを一瞥することすら困難なほど巨大である。例えば、インターネットはそのサイズが刻々と変化しており、また、様々なサーバに分散されている全ての情報にアクセスすることは現実的ではない。このような情報爆発に対応するため、新しいアルゴリズムや手法が必要になってきている。

入力データ長に対してその線形時間がなければほとんど何も計算できないと考えられがちだが、実際には定数時間だけでも計算できるものが多数ある。その一つがプロパティー検査である。プロパティー検査とは、大規模なグラフやデータベースからランダムサンプリングを行い、そのごく僅かなサンプルを元に、確率的な近似アルゴリズムを用いて高速に結論を出力するタイプの検査手法である。プロパティー検査によって、NP 完全問題の一部も検査可能であると知られている。Lovász [50] がいうように、プロパティー検査は帰納法の応用であると考えられる。

プロパティー検査は、形式的検証の分野において、計算量の高い検証を行う前の高速なフィルターとして提案された。形式的検証では、ユーザーが何かしら形式論理等の形式言語で目的のプロパティーを定義し、コンピュータはシステムがそれを満たしているかどうかを計算する。このように形式言語で定義されるプロパ

ティーの検査は自然な課題である。また、大規模な関係データベースへの応用も考えられる。その場合は、ユーザーがSQL等でクエリーを書いてコンピュータが自動的に効率的の良い確率的な近似を行えることが望ましい。SQLは一階述語論理の拡張に近いこと (Libkin [49] 等参考) や、形式的検証も形式論理でプロパティーを定義することからも、論理式の検査が重要になる。

本研究では、一階述語論理のプロパティー検査について研究する。一階述語論理で定義できるプロパティーには検査可能と検査不可能なものが存在する。したがって、検査できるサブクラスとできないサブクラスにプロパティーを分類することが目的である。この課題は、Alon ら [3] が初めて研究を行い、一階述語論理を二つのクラスに分け、その一方は検査可能であり他方が検査不可能であると証明している。しかしながら、その結果はループなしの無向グラフに限っている。Alon らの結果は深くて影響力もあるが、一階述語論理をより細かく分類するとより良い結果を導くことが可能である。

本研究の目的は、一階述語論理について、検査可能および検査不可能な文法的サブクラス（Börger ら [15] のように定義するもの）の完全な分類を行うことである。本研究の成果はゲーデルクラスを除けば完全な分類になっている。本研究の主な成果は、以下の通りである。

1. グラフプロパティー検査やハイパーグラフプロパティー検査の拡張である関係プロパティー検査を提案している。また、従来のモデルとの関係を明らかにし、いくつかのバリエーションも提案している。さらに、それぞれのバリエーションの差を明らかにし、関係プロパティー検査を元にしたプロパティー検査の分類問題の定義を行っている。

2. 関係プロパティー検査において、いくつかの基本的な成果を証明している。

本研究では、この成果を何回も応用しているため、まとめて証明している。

3. Alon ら [3] が検査可能だと証明しているクラスを、ループなしの無向グラフから関係ストラクチャーへ拡張し、ラムゼイクラスという一階述語論理の古典的に有名なサブクラスが全て検査可能であることを証明している。検査可能とはバリエーションによらないで、どれを使っても検査可能である。この成果は Austin と Tao [11] が証明しているハイパーグラフ理論の複雑な成果を応用している。

4. 等号ありのアッカーマンクラスという一階述語論理の古典的に有名なサブクラスが全て検査可能であることを証明している。こちらの証明はモデル理論の議論からできている。

5. Alon ら [3] が検査不可能だと証明しているクラスのより強い成果を証明している。Alon らの証明では、量化子が 17 個あれば（パターンによって）検査不可能なプロパティーを定義できることが示されている。本研究の成果では、Alon らと同じパターンで 4 個でも足りることを証明している。この証明は、Alon らが証明しているグラフ理論の定理を応用している。

6. Kahr-Moore-Wang クラスという一階述語論理の古典的に有名なサブクラスが検査不可能だと証明している。これは検査不可能性として最小クラスの一つである。この成果から、量化子が 3 個あるとこのパターンで検査不可能なものを表すことができると分かる。量化子が 2 個以下だと上の成果から検査可能だと分かるため、検査不可能なプロパティーを一階述語論理で定義するには量化子の必要充分な数は 3 であると分かる。この証明は、Alon ら [2] が示した、ブール関数の同型問題が検査不可能であることを証明するためのアイディアを応用している。

7. エコールなしの Kahr-Moore-Wang クラスは、検査不可能なプロパティーを含むことを証明している。ただし、この成果においては、関係プロパティー検査のバリエーションに限る。すなわち、普段のバリエーションでは検査不可能だが、違うバリエーションでは検査可能かどうかは未だ証明していない。

形式的検証等への応用を考えると、ユーザーが定義するプロパティーを自動的に検査するために、コンピュータが与えられた論理式から検査アルゴリズムを自動的に作成ことが必要である。本研究で検査可能であると証明しているクラスについては、検査アルゴリズムを論理式から作る方法も同時に提案している。したがって、ユーザが与えた論理式から、自動的に定数時間の確率的近似を行うシステムを構築することが可能である。

本研究の成果を得るために、形式論理とモデル理論、グラフとハイパーグラフ理論、ブール関数の理論等の道具を応用している。

Contents

List of Figures

Acknowledgments

This dissertation would not have been written without the help and support of many people. I am especially grateful to my advisor, Prof. Thomas Zeugmann, for his endless support and constant discussions over the past six years, and also for first introducing me to the paper that eventually led to this thesis. His love of research and science has been inspiring.

I am thankful to Prof. Shin-ichi Minato for his generous help, continuous since the day I first visited the Laboratory for Algorithmics, and also for providing access to countless opportunities via the ERATO Minato project.

I am indebted to Prof. Hiroki Arimura for his guidance over the past six years, and also for the generous support of the GCOE-NGIT project. His generosity has allowed me to attend international conferences, visit the University of Latvia and spend a semester at the University of Massachusetts, Amherst.

I appreciate the help of Prof. Takuya Kida, most recently in proofreading the Japanese abstract for this dissertation.

Prof. Neil Immerman first introduced me to the beauty of descriptive complexity and finite model theory when I was an undergraduate at the University of Massachusetts, Amherst. I am thankful to him for allowing me to take 741 as an undergraduate, and look forward to the future of descriptive programming.

I am thankful to Prof. Rūsiņš Freivalds, Prof. Beate Bollig and Neil Immerman for hosting me as a visitor over the past years.

I would not have been able to complete this dissertation without the expert help of Sachiko Soma, who ably saved me from being paralyzed by paperwork on countless occasions. I am also grateful to the GCOE staff, in particular Ayami Yonemoto and Miyuki Yoshimura, for their understanding and assistance over the past three years.

I appreciate the wonderful environment generously provided by the Graduate School of Information Science and Technology at Hokkaido University. The amazing mountain view from my office has been inspirational. I am especially indebted to the Japan Society for the Promotion of Science, which generously supported me for three years with a research fellowship. I am very thankful to MEXT for Grant No. 2100195209, which has supported my research over the past years, and also for earlier scholarships.

I am thankful to Miki Sakurada for her patience and for keeping me sane over the past few years. Finally, I will always be indebted to my family for their unfailing love and support, for nurturing in me a love of understanding the world, and for putting up with me being so far away for so long.

Chapter 1

Introduction

Property testing is an application of inductive reasoning. Given a large object, for example a massive graph or database, we wish to state some conclusion about the entire structure after examining only a small, randomly selected sample. Lovász [50] has described it as the "third reincarnation" of this kind of general approach, after statistics and machine learning.

The astonishing growth in massive data-sets requires us to find new techniques and approaches for much of the computation that we want to do. In fact, for many large data-sets of interest (e.g., the Internet), we cannot even determine the precise *size* of the data. Such objects are generally stored remotely, and there is a non-trivial cost for examining particular bits. It may be impractical to move beyond constant time if we cannot even determine the size of the data we are interested in, and the sheer amount of data renders many common algorithms impractical. Given the (time) cost of accessing remote data, we may also wish to minimize the number of bits that we examine.

At first, it may seem that linear time is a minimum requirement for meaningful computation; after all, how can we compute a property of some data that we don't have time to look at? Surprisingly, there is an entire world of interesting computation that can be done in sub-linear time, and even in constant time.

Informally[1], we call a property testable if we can approximate it as closely as we like in constant time. In this thesis, we focus on testing properties that are expressible in first-order logic. The focus on queries expressed in a formal languages is quite natural; in databases, for example, users generally make queries against massive objects in a formal query language (e.g., SQL). Given the "unusual effectiveness" of logic in computer science [35], it is natural to focus (in particular) on first-order properties.

This thesis indicates the possibility of a system that takes queries in a restricted formal language (e.g., the restricted syntactic fragments of first-order logic that are testable), and automatically generates probabilistically approximate answers to these queries in constant time. Such a system is also natural from the perspective of formal verification (where this general approach originated, see Section 1.1). There, users generally specify properties of systems they wish to verify by writing (Boolean) queries in a logic. Given the computational demands of verification, a very fast (constant-time) randomized approximate verification could be useful by allowing us to quickly reject very bad systems.

The idea of testing syntactic subclasses of first-order logic was first-considered in an influential paper by Alon *et al.* [3] (see Section 1.1 for a discussion of related work), and this thesis builds upon the classification that began there. Here, we

[1]See Section 2.3 for formal definitions.

prove a nearly-complete classification of prefix-vocabulary classes[2] according to their testability.

The thesis is structured as follows. We begin by introducing related work, together with a brief history of property testing in Section 1.1. Then, we state the main results of this thesis in Section 1.2.

Chapter 2 focuses on definitions and other preliminaries. In Chapter 3, we prove basic, fundamental results about property testing that we will need for proofs in later chapters. These basic results first appeared in [38, 43], and will also appear in [42]. We consider testable classes in Chapter 4. The results in Section 4.1 first appeared in [38] in a preliminary form with two-sided error. The results in Section 4.2 first appeared in [39].

We proceed to untestable classes in Chapter 5. The results in Section 5.1 first appeared in [40]. These last three results will also appear as [42]. The results in Section 5.2 first appeared in [41], while the results in Section 5.3 have not been previously published.

1.1 Related Work

We begin with a brief history and overview of property testing. There is a recent introduction to graph property testing by Goldreich [28], and two recent surveys by Ron, one focusing on connections with learning theory [61] and one focusing on the algorithmic techniques [62] used in testability. There are also earlier surveys, including those by Ron [60] and the wonderfully-titled Fischer [19].

[2]A prefix-vocabulary class is defined by the pattern of quantifiers and vocabulary, see Definition 3 in Section 2.2.

Property testing is a form of approximation where we trade accuracy for efficiency. It seems that de Leeuw *et al.* [47] was the first to formalize probabilistic machines. They showed that such machines cannot compute uncomputable properties under reasonable assumptions. However, they mention the possibility that probabilistic machines could be more *efficient* than deterministic machines, a topic which was then investigated by Gill [26]. An early example of such a result is Freivalds' [25] matrix multiplication checker.

The study of property testing itself began in program verification (see, e.g., Blum *et al.* [14] as well as Rubinfeld and Sudan [63]). Goldreich *et al.* [29] first considered the testability of graph properties in a seminal paper and showed the existence of testable NP-complete properties. An approach using incidence lists to represent bounded-degree graphs was introduced by Goldreich and Ron [27]. Parnas and Ron [55] generalized this approach and attempted to move away from the functional representation of structures. There has been a great deal of recent work on graph property testing, see the survey by Alon and Shapira [7].

For other types of structures, Alon *et al.* [5] showed that the regular languages are testable and that there exist untestable context-free languages. Chockler and Kupferman [17] extended the positive result to the ω-regular languages.

There is also recent work on testing properties of (usually uniform) hypergraphs. Fischer *et al.* [22] defined a general model that is roughly equivalent to one of our models, namely $\mathcal{T}_r$ based on Definition 5 below, and showed that hypergraph partition problems are testable in this framework. Very recently, Austin and Tao [11] have shown that all hereditary properties[3] of colored, directed, non-uniform hyper-

[3]A hereditary property is one that is closed under taking induced substructures.

graphs are testable in a model that is roughly equivalent to another of our models, $\mathcal{T}_{mr}$ based on Definition 8 below.

Szemerédi's regularity lemma (see, e.g., the survey by Rödl and Schacht [59]) has been extremely influential in (dense) graph property testing and there has been a great deal of work on recent extensions (see, e.g., [30, 58, 68]) of this lemma to hypergraphs. We are not aware of any extensions to non-uniform hypergraphs or finite (relational) structures, but are very interested in such topics. As Alon *et al.* [3] noted, proofs of testability that avoid the regularity lemma often result in better query complexity. We therefore prove testability directly when we know how to.

Alon *et al.* [3] began a *logical* characterization of the testable (graph) properties, see Subsection 1.1.1. Alon and Shapira [8] gave a characterization of a natural subclass of the graph properties testable with one-sided error, which Rödl and Schacht [57] generalized to hypergraphs. Alon *et al.* [4] showed a combinatorial characterization of the graph properties testable with a constant number of queries. It would be particularly interesting to consider extensions of this last result to hypergraphs or relational structures.

1.1.1 Previous Work on the Classification

We briefly outline prior work on the classification for testability before stating our main results. We begin with monadic first-order logic. Löwenheim [51] proved that satisfiability is decidable[4] for monadic first-order logic, and McNaughton and

[4]A class is said to be *decidable* (for satisfiability) if, given an arbitrary formula from the class, one can decide if there exists a (possibly infinite) model satisfying the formula.

Papert [52] showed that it (with ordering and some arithmetic) characterizes the star-free regular languages. The testability of this class is then implied by a result of Alon *et al.* [5]. Using instead Büchi's [16] result that monadic second-order logic characterizes the regular languages, we get a parallel with Skolem's [66] extension of Löwenheim's result to second-order logic. Of course, we are focused on the testability of classes of first-order formulae.

Below, we use the classification notation that will be introduced formally in Definition 3. Informally, we represent classes with a triple $[\Pi, p]_e$, where Π denotes the pattern of quantifiers allowed, the infinite sequence p denotes the maximum number of permitted predicate symbols for each arity (we omit trailing zeros and *all* means that any number of predicate symbols with any arities are permitted), and e denotes whether $=$ is allowed.

Skolem [67] also showed that $[\forall^*\exists^*, \mathit{all}\,]$ is a reduction class[5]. Alon *et al.* [3] found an untestable graph property (essentially an encoding of graph isomorphism). In particular, this property is expressible in $[\forall^*\exists^*, (0,1)]_=$, and an examination of the proof reveals that a prefix of $\forall^{12}\exists^5$ suffices.

The class $[\exists^*\forall^*, \mathit{all}\,]_=$ was first studied in a seminal paper by Ramsey [56], who showed that it is decidable as part of a stronger result characterizing its spectrum. Alon *et al.* [3] showed that the restriction of Ramsey's class to undirected loop-free graphs (a restriction of $[\exists^*\forall^*, (0,1)]_=$) is testable.

[5]A class is a *reduction class* if the satisfiability problem for first-order logic can be reduced to the satisfiability problem for the class. These classes are therefore undecidable (for satisfiability).

1.2 Results

As mentioned above, Alon *et al.* [3] found an untestable property expressible with seventeen quantifiers ($\forall^{12}\exists^{5}$). Although this is an impressive result, we might wish to know whether it is optimal. More concretely, we would like to know the minimum number of universal quantifiers, as well as of existential quantifiers, required to express an untestable property. In addition, we would like to know the minimum total number of quantifiers needed to express an untestable property, as it is not *prima facie* necessary that one can achieve these two minima simultaneously. Previous work by Alon *et al.* [3] implies upper bounds of twelve universal, five existential and seventeen total quantifiers, and it is natural to ask if these bounds can be improved. The following is an informal summary of our results addressing this question.

Remark 1. *The minimum number of quantifiers sufficient to express an untestable property in a first-order relational language is*

1. *Two universal quantifiers;*
2. *One existential quantifier;*
3. *Three quantifiers in total.*

These minima can be achieved in the vocabulary of directed graphs (i.e., one binary relation).

We now introduce the main results of this thesis. First, we develop a framework for relational property testing including variations corresponding to the different models considered in the literature for non-uniform hypergraph[6] testing. We use this framework to prove the following.

1. All (relational) properties expressible by formulae in Ackermann's class with equality ($[\exists^*\forall\exists^*, \mathit{all}\,]_=$) are *testable* in all of our models.

2. All (relational) properties expressible in Ramsey's class ($[\exists^*\forall^*, \mathit{all}\,]_=$) are *testable* in all of our models. This extends the result by Alon *et al.* [3] to the full class.

3. There exist graph properties expressible in the class $[\forall^3\exists, (0,1)]_=$ that are *untestable* in all of our models. This considerably sharpens the untestable class of Alon *et al.* [3].

4. There exist graph properties expressible in the class $[\forall\exists\forall, (0,1)]_=$ (the Kahr-Moore-Wang prefix) that are untestable in all of our models.

5. There exist graph properties expressible in the class $[\forall\exists\forall, (0,1)]$ (*without* equality) that are untestable in (at least) one of our models.

The last four results improve upon results of Alon *et al.* [3] in various ways, and the second result relies on an application of a strong result by Austin and Tao [11].

In the notation introduced as Definition 3 below, the current classification for testability is as follows.

- Testable classes

    1. Monadic first-order logic: $[\mathit{all}\,, (\omega)]_=$.

[6]A hypergraph is uniform if all edges have the same arity, and non-uniform if edges may have different arities.

2. Ackermann's class with equality: $[\exists^*\forall\exists^*, \mathit{all}\,]_=$.
3. Ramsey's class: $[\exists^*\forall^*, \mathit{all}\,]_=$.

- Untestable classes
    1. $[\forall^3\exists, (0,1)]_=$.
    2. $[\forall\exists\forall, (0,1)]_=$.
    3. $[\forall\exists\forall, (0,1)]$[7].

[7]Our proof for this class without equality is currently restricted to one of our models of testability ($\mathcal{T}_{mr}$ based on Definition 8 below). We suspect that this class is also untestable in the other models, however $\mathcal{T}_{mr}$-style testing seems the most natural to us.

Chapter 2

Preliminaries

In this chapter, we introduce our notation and definitions. We separate these preliminaries into sections related to basic, fundamental notions (e.g., sets, etc.), logic (e.g., prefix vocabulary classes) and property testing. Much of this material is standard and readers familiar with a particular section can safely skip it. However, we introduce several variations of property testing for relational structures in Section 2.3 and encourage readers unfamiliar with our variations to review, at least, their relationship with other models from the literature.

2.1 Fundamentals

Before proceeding further, we recall fundamental definitions and introduce notation for familiar objects such as natural numbers, sets and strings. Our definitions are standard and readers familiar with this material can safely skip to Section 2.2.

The natural numbers are denoted by $\mathbb{N}$ and are the set of non-negative integers. We denote the set of real numbers by $\mathbb{R}$, although these are generally used for probabilities and so we usually use only real numbers $p \in [0, 1]$. We use bold characters to denote vectors, for example $\mathbf{x} \in \mathbb{R}^3$. Vectors are row vectors unless otherwise noted, we denote the *transpose* of a vector by $\mathbf{x}^T$. If $\mathbf{x} = (x_1, \ldots, x_a)$ is a vector, we call x_i the i-th *component* of $\mathbf{x}$.

The empty set is denoted by $\emptyset$. If A and B are sets, then the union of A and B is $A \cup B := \{x \mid x \in A \text{ or } x \in B\}$ and the intersection of A and B is $A \cap B := \{x \mid x \in A \text{ and } x \in B\}$. Furthermore, the set difference of A and B is $A \backslash B := \{x \mid x \in A \text{ and } x \notin B\}$. We generalize the union and intersection in the usual way, $\bigcup_{i \geq 0} A_i := A_0 \cup A_1 \cup \ldots$ and $\bigcap_{i \geq 0} A_i := A_0 \cap A_1 \cap \ldots$ respectively.

Set A is a subset of set B, written $A \subseteq B$ if $A \backslash B = \emptyset$. Set A is a *proper* subset of set B, written $A \subset B$ if $A \subseteq B$ and $B \backslash A \neq \emptyset$. The cardinality of a set A is the number of elements in the set, written $|A|$.

The product of sets A and B is the set of ordered pairs, $A \times B := \{(a, b) \mid a \in A \text{ and } b \in B\}$. The set of n-tuples of set A, written A^n is defined inductively as follows. First, $A^1 = A$. Then, $A^{n+1} = A^n \times A$. We will always omit the extra parentheses, and so $(1, 2, 3)$ denotes $((1, 2), 3)$. The number of elements in the tuple is the *arity* n. A *predicate* P with arity n of set A is any subset of A^n. If $\mathbf{x} \in A^n$, we will generally abbreviate the proposition $\mathbf{x} \in P$ with $P(\mathbf{x})$.

An alphabet Σ is a set of symbols, and a string w over Σ is some sequence of the symbols in Σ. The empty string is denoted by λ. For example, $\{0, 1\}$ is the alphabet of binary strings and 0100 is an example of such a string. We number the positions in a string w from left to right with $0, 1, \ldots, n - 1$ where n is the length of the string. Of course, the empty string λ has length 0. As usual, Σ^* is the free monoid of Σ and any subset $L \subseteq \Sigma^*$ of it is a *language*.

Let w be a string over the alphabet Σ. The *concatenation* of strings u and v is uv, while the *product* of two sets of strings L_1 and L_2 is $L_1L_2 := \{uv \mid u \in L_1 \text{ and } v \in L_2\}$. The *reversal* of w is written $\overleftarrow{w}$. Position i of $\overleftarrow{w}$ corresponds to position $n-1-i$ of w. Formally, $\overleftarrow{\lambda} = \lambda$ and $\overleftarrow{aw} = \overleftarrow{w}a$ for $a \in \Sigma$.

We mention a number of well-known classes of languages, for example the classes of regular and context-free languages. Hopcroft and Ullman [36] is a well-known introduction to these classes.

It is natural to represent a binary string $w \in \{0,1\}^*$ as a pair $\{U, \mathcal{S}\}$ where U is the finite set of bit positions $0, \ldots, n-1$[1] and $\mathcal{S} \subseteq A$ is a monadic *predicate*. We will define $\mathcal{S}(i)$ to mean that "bit position i of w is 1."

Graphs provide another natural example and allow for representation as a pair, $(V, \mathcal{E})$. Here V is the set of vertices and the edge set $\mathcal{E} \subseteq V^2$, a set of ordered pairs of V. The "names" of the vertices are not interesting to us, and we will identify them as $0, \ldots, n-1$ where n is the number of vertices. It is therefore natural to represent a graph as a pair $\{V, \mathcal{E}\}$ where $\mathcal{E}$ is a binary predicate over V.

We will formalize these notions more exactly in the following section. In particular, one of our goals is a generalized notion of property testing instead of restricting ourselves to fixed kinds of structures such as graphs and binary strings. The definitions in the following section are therefore necessarily abstractions of the ideas above.

[1]The universe is the empty set if w is the empty string.

2.2 Logic

We are particularly interested in the testability of classes of first-order logic, and so we need various definitions related to logic. We begin with vocabularies and structures.

Definition 1. *A (relational)* vocabulary τ *is a tuple of distinct predicate symbols* R_i *together with their arities* a_i*,*

$$\tau := (R_1^{a_1}, \ldots, R_s^{a_s}) .$$

Two examples (unique up to renaming) of vocabularies are $\tau_G := (E^2)$, the vocabulary of directed *graphs* and $\tau_S := (S^1)$, the vocabulary of *binary strings.*

Definition 2. *A* τ-structure A *is an* $(s+1)$*-tuple*

$$A := (U, \mathcal{R}_1^A, \ldots, \mathcal{R}_s^A) ,$$

where U *is a finite universe and each* $\mathcal{R}_i^A \subseteq U^{a_i}$ *is a predicate corresponding to the predicate symbol* R_i *of* τ*.*

We generally identify U with the non-negative integers $\{0, \ldots, n-1\}$ and use $n = \#(A)$ for the size of the universe of a structure A. The universe U of a binary string is the set of bit positions, which we will identify as $\{0, \ldots, n-1\}$ from left to right. For $i \in U$, we interpret $i \in \mathcal{S}$ as "bit i of the string is 1." We generally omit the superscript A from the relations and include it only when we wish to explicitly distinguish the same relation in different structures.

The set of all τ-structures with universe size n is $STRUC^n(\tau)$ and the set of all (finite) τ-structures is $STRUC(\tau) := \bigcup_{n\geq 0} STRUC^n(\tau)$. A *property P of τ-structures* is any subset of $STRUC(\tau)$. We also call such properties τ-properties. We say that a τ-structure A *has* P if $A \in P$.

We use *language* to refer to string properties and P to denote properties. We refer to members of $STRUC(\tau_G)$ as *graphs*, and note that our graphs are directed and may contain loops.

A simple example of a graph property is the property of being a complete graph. This property is the set of all (finite) graphs which have full edge relations, i.e., $P_K := \bigcup_{n\geq 0}\{(U_n, \mathcal{E}^G) \mid U_n = \{0, \ldots, n-1\}, \mathcal{E}^G = U_n \times U_n\}$.

We use a predicate logic with equality that does not contain function symbols. There are no ordering symbols such as $\leq$ or arithmetic relations such as *PLUS*. The first-order logic of vocabulary τ is built from the atomic formulae $x_i = x_j$ and $R_i(x_1, \ldots, x_{a_i})$ for variable symbols x_j and predicate symbols $R_i \in \tau$ by using the Boolean connectives and quantifiers $\exists$ and $\forall$ in the usual way.

Formula φ of vocabulary τ is evaluated in the usual way and defines property $P := \{A \mid A \in STRUC(\tau) \text{ and } A \models \varphi\}$. Lower-case Greek letters φ, ψ and γ refer to first-order formulae and x, y, and z to first-order variables.

Our classification definitions are from Börger *et al.* [15] except that we omit function symbols. Essentially, we classify first-order sentences according to their pattern of quantifiers and vocabulary.

Definition 3. *A* prefix vocabulary class *is specified as* $[\Pi, p]_e$*, where* Π *is a string over the four-character alphabet* $\{\exists, \forall, \exists^*, \forall^*\}$*,* p *is a sequence over* $\mathbb{N}$ *and the first infinite ordinal* ω*, and* e *is '='* *or the empty string.*

We often use *all* as an abbreviation for the sequence $(\omega, \omega, \omega, \ldots)$. Now that we have defined the syntactic specification of a prefix vocabulary class, we define the class specified by a triple $[\Pi, p]_e$. Recall that a first-order sentence φ is in prenex normal form if it is in the form $\varphi := \pi_1 x_1 \pi_2 x_2 \ldots \pi_r x_r : \psi$, with quantifiers π_i, $1 \leq i \leq r$, and quantifier-free ψ. Such a φ is a member of the prefix vocabulary class given by $[\Pi, (p_1, p_2, \ldots)]_e$, where $p_i \in \mathbb{N} \cup \{\omega\}$ if

1. The string $\pi_1 \pi_2 \ldots \pi_r$ is contained in the language specified by Π when Π is interpreted as a regular expression[2].

2. If p is not *all*, at most p_i distinct predicate symbols of arity i appear in ψ.

3. Equality $(=)$ appears in ψ only if e is '$=$'.

Here, Π is the pattern of quantifiers, p is the maximum number of predicate symbols of each arity and e determines whether or not the equality symbol is permitted.

2.3 Property Testing

In property testing, the basic goal is to distinguish between structures that have a desired property and those that are far from having the property. Formalizing this requires a definition of "far", and different definitions results in different models of testing. We will give three different distance measures, each based on progressively refining a generalization of the dense graph testing model introduced by Goldreich *et al.* [29].

[2]Technically, we also let the empty string match expressions $\forall$ and $\exists$. See the discussion in Börger *et al.* [15].

2.3.1 Distance Measures

Our first definition is $\text{dist}(A, B)$, the fraction of tuples with differing assignments in A and B. We use $\oplus$ to denote exclusive-or.

Definition 4. *Let $A, B \in STRUC(\tau)$ be any τ-structures such that $\#(A) = \#(B) = n$. The* distance *between structures A and B is*

$$\text{dist}(A, B) := \frac{\sum_{i=1}^{s} |\{\mathbf{x} \mid \mathbf{x} \in U^{a_i} \text{ and } \mathcal{R}_i^A(\mathbf{x}) \oplus \mathcal{R}_i^B(\mathbf{x})\}|}{\sum_{i=1}^{s} n^{a_i}} .$$

This is a natural definition; it is equivalent to mapping the structures to binary strings in the usual way and using the normal string testing definitions (based on normalized Hamming distance). We note that Definition 4 is common in the literature on graph property testing, but that it is generally not used in non-uniform hypergraph testing. However, it results in the *weakest* (cf. Theorem 1 below) notion of testability that we consider, and so we prefer to use it when proving *untestability* results.

The simplicity of Definition 4 is attractive, however it has some shortcomings. In particular, any difference in low-arity relations is asymptotically dominated by the number of high-arity tuples. This has a number of undesirable effects, as testing relational structures degenerates roughly to testing uniform hypergraphs. For example, consider (not necessarily admissible[3], vertex) 3-colored graphs with the vocabulary $\tau_C := (E^2, R^1, G^1, B^1)$, where we use the binary predicate E to represent edges and the monadic predicates to represent colors. We might wish to test if the given coloring is admissible. However, if we use Definition 4, then (in

[3]An admissible vertex-coloring is one that assigns distinct colors to adjacent vertices.

large graphs), the given coloring is insignificant and we actually test whether the graph is 3-colorable. We need a different model for our task.

Our first attempt to resolve this is rdist.

Definition 5. *Let $A, B \in STRUC^n(\tau)$ be τ-structures. Then, the* r-distance *is*

$$\text{rdist}(A, B) := \max_{1 \le i \le s} \frac{|\{\mathbf{x} \mid \mathbf{x} \in U^{a_i} \text{ and } \mathcal{R}_i^A(\mathbf{x}) \oplus \mathcal{R}_i^B(\mathbf{x})\}|}{n^{a_i}} \, .$$

While Definition 4 gave equal weight to each *tuple* regardless of its arity, the above gives equal weight to each *relation.* The model of testability resulting from Definition 5 is essentially equivalent to the model used by Fischer *et al.* [22].

However, loops (i.e., self-edges (x, x)) in graphs and other *subrelations* of relations are similar to low-arity relations. In Definition 5, these are still dominated by the "non-degenerate" tuples. Definition 8 will resolve this issue and result in a model of testability essentially equivalent to that implicit in Austin and Tao [11]. We begin by defining the syntactic notion of *subtype* before proceeding to subrelations.

Definition 6. *A* subtype *S of a predicate symbol with arity a is any partition of the set $\{1, \ldots, a\}$.*

For example, graphs have a single, binary predicate symbol E^2 which has two subtypes: $\{\{1, 2\}\}$ and $\{\{1\}, \{2\}\}$, corresponding to loops and non-loops respectively. Let $SUB(R_i^{a_i})$ denote the set of subtypes of predicate symbol $R_i^{a_i}$.

Definition 7. *Let $A \in STRUC(\tau)$ be a τ-structure with universe U, and let S be a subtype of predicate symbol $R_i^{a_i} \in \tau$. We define the following.*

- $s^U(S)$, *the* tuples that belong to S, *is the set of* $(x_1, \ldots, x_{a_i}) \in U^{a_i}$ *satisfying the following condition. For every* $1 \leq j, k \leq a_i$, $x_j = x_k$ *iff* j *and* k *are contained in the same element of* S.

- *The* subrelation $s^A(S)$ *of* A *corresponding to* S *is* $s^A(S) := s^U(S) \cap \mathcal{R}_i^A$.

Returning to our example of graphs, the sets of loops and non-loops are the subrelations of the edge relation $\mathcal{E}$ corresponding to the subtypes $\{\{1, 2\}\}$ and $\{\{1\}, \{2\}\}$ of E^2, respectively.

We denote the symmetric difference of sets U and V by $U \triangle V$, i.e.,

$$U \triangle V := (U \backslash V) \cup (V \backslash U).$$

Definition 8. *Let* $A, B \in STRUC^n(\tau)$ *be* τ*-structures with universe size* n*. The* mr-distance *between* A *and* B *is*

$$\operatorname{mrdist}(A, B) := \max_{R \in \tau} \max_{S \in SUB(R)} \frac{|s^A(S) \triangle s^B(S)|}{n!/(n - |S|)!}.$$

The mr-distance between structures is the fraction of assignments that differ in the most different subtype. Although this definition is the most involved, it has a number of advantages. First, it is essentially equivalent to the model used by Austin and Tao [11] based on the fraction of induced structures (of a particular size) differing between structures.

More importantly, it does not allow us to make untestable properties testable by increasing the arity of relations (i.e., untestable graph properties encoded in binary subrelations of higher-arity relations remain untestable). This means that the untestable properties (and prefix-vocabulary classes) are closed downwards in the way required by Gurevich's Classifiability Theorem[4], and so we are guaranteed a finite classification of the testable and untestable prefix vocabulary classes.

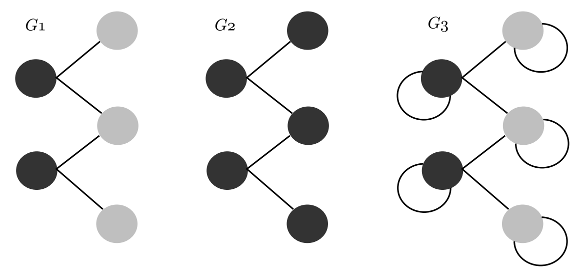


Figure 2.1. Comparison of distance measures dist, rdist and mrdist.

Figure 2.1 demonstrates the differences between the distance measures. The colored graphs in the figure have a binary edge relation and a monadic color relation. If we assume that the graphs are large enough for asymptotic behavior to dominate, then we can make the following observations.

1. The dist between all graphs is small. This is because the non-loop edges do not differ, and these tuples dominate dist.

2. The rdist is large between G_1 and G_2, large between G_2 and G_3, and small between G_1 and G_2. This is because the rdist reflects the difference in monadic color assignments, but still allows non-loop edges to dominate loops in the edge relation.

3. The mrdist is large between all graphs. This is because rdist reflects differences in each subrelation.

Given that testers make queries to a small portion of the structure, mrdist is

[4]Gurevich [34] gives a nice introduction to this theorem, which first appeared as [32] (in English as [33]). See Section 2.3 of Börger *et al.* [15] for a nice proof and related material.

particularly natural as a distance measure. This is because testers can notice differences in subrelations and low-arity relations, which are best reflected in mrdist.

2.3.2 Testing Definitions

All three distance measures generalize to distances from properties in the usual way. The distance from a structure to a property is the distance to the closest structure that has the property. For example, $\mathrm{dist}(A, P)$ is defined as follows.

Definition 9. *Let P be a τ-property and let A be a τ-structure with universe size n. Then,*

$$\mathrm{dist}(A, P) := \min_{A' \in P \cap STRUC^n(\tau)} \mathrm{dist}(A, A') \ .$$

The remaining two distance measures extend in the same way.

Definition 10. *An ε-*tester *for property P is a randomized algorithm given an oracle which answers queries for the universe size and truth values of relations on desired tuples in a structure A. The tester must accept with probability at least $2/3$ if A has P and must reject with probability at least $2/3$ if $\mathrm{dist}(A, P) \geq \varepsilon$.*

Testers are called *oblivious* (see Alon and Shapira [8]) if they are not allowed to make decisions based on the size of the universe. More concretely, a tester in their setting is only allowed to give the oracle a natural Q, and the oracle then uniformly randomly selects Q elements of the universe of A and returns the resulting induced substructure. However, if A is of size smaller than Q, then the entire structure is returned. This is more restricted than our model, but our positive results hold even in the oblivious setting.

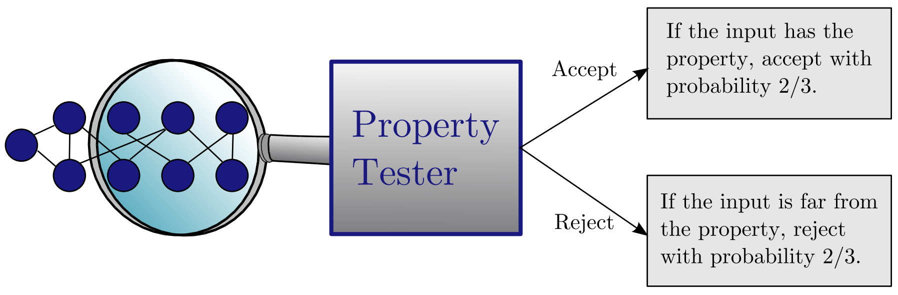

Property Tester
Accept
Reject
If the input has the property, accept with probability 2/3.
If the input is far from the property, reject with probability 2/3.

Figure 2.2. A property tester.

Figure 2.2 is an example of a property tester. Note that the sample is of constant size – one must choose a new tester to get a new sample size. Also, note that although the graph in Figure 2.2 is not bipartite, it is not far from being bipartite.

Some of our results hold even when the testers are restricted to one-sided error, where the following definition applies.

Definition 11. *An ε-tester for P has* one-sided error *if it accepts with probability* 1 *if A has P and rejects with probability at least* $2/3$ *if* $\operatorname{dist}(A, P) \geq \varepsilon$.

Definition 12. *Property P is* testable *if for every $\varepsilon > 0$ there is an ε-tester making a number of queries which is upper-bounded by a function depending only on ε.*

We say that a property P is testable *with one-sided error* if the ε-testers satisfy the additional restriction of having one-sided error. Note that we allow the ε-testers to be different for each $\varepsilon > 0$, which results in uniform and non-uniform versions of testability. Although most positive results in the literature hold for uniform testability, see Alon and Shapira [9] for a property that is testable only with uncomputable $c(\varepsilon)$, or [43] for an undecidable property that is testable nonuniformly but not uniformly. Our results hold in both cases[5] and so we will not distinguish between them.

We let $\mathcal{T}$ be the set of testable properties using the dist definition, $\mathcal{T}_r$ be the set of testable properties using the rdist definition and $\mathcal{T}_{mr}$ be the set of testable properties using the mrdist definition. For convenience, we often refer to, e.g., $\mathcal{T}_{mr}$-style testers or $\mathcal{T}_r$-style testing.

Definition 13. *We use the following conventions to avoid unwieldy language.*

1. *A sentence is* (un)testable *if the property it defines is (un)testable.*
2. *A prefix class is* testable *if every sentence in it expresses a testable property for every vocabulary in which it is evaluable.*
3. *A prefix class is* untestable *if it contains an untestable sentence.*

[5]That is, our negative results hold for non-uniform testing and positive results for uniform testing. In the uniform case, we must restrict Lemma 7 to decidable properties. All properties considered in the present paper are clearly decidable.

Chapter 3

Basic Results

In this chapter, we prove various basic results that we will need for later results. We include the proofs here for completeness, even though these results are not particularly difficult.

3.1 Testing is Hardest in Minimal Vocabularies

We begin with the following simple lemma, which justifies the intuition that we can focus on the minimal vocabulary needed in a formula and ignore vocabularies that include extraneous predicate symbols. Here, an extension of a vocabulary τ is any vocabulary formed by adding a new, distinct predicate symbol to τ.

Lemma 1. *Let φ be a formula in the first-order logic of vocabulary τ and let τ' be any extension of τ. If φ defines a testable τ-property, then the τ'-property it defines is also testable.*

Proof (Lemma 1). Let φ define τ-property P and τ'-property P'. Assume the "new" predicate symbol in τ' is N of arity a. Let T_ε^τ be an ε-tester for P. We will show that it is also an ε-tester for P'. Assume $A \in STRUC(\tau')$ has property P'. Removing the N predicate, the corresponding $A' \in STRUC(\tau)$ has property P and so T_ε^τ accepts with probability at least 2/3, as desired.

Assume that $\text{dist}(A, P') \geq \varepsilon$ and again let A' be the structure of type τ formed by removing the N predicate from A. By the definition of distance,

$$\text{dist}(A', P) = \min_{B \in P} \frac{\sum_{i=1}^{s} |\{\mathbf{x} \mid \mathbf{x} \in U^{a_i} \text{ and } \mathcal{R}_i^{A'}(\mathbf{x}) \oplus \mathcal{R}_i^{B}(\mathbf{x})\}|}{\sum_{i=1}^{s} n^{a_i}} \geq$$
$$\min_{B \in P} \frac{\sum_{i=1}^{s} |\{\mathbf{x} \mid \mathbf{x} \in U^{a_i} \text{ and } \mathcal{R}_i^{A}(\mathbf{x}) \oplus \mathcal{R}_i^{B}(\mathbf{x})\}|}{n^a + \sum_{i=1}^{s} n^{a_i}} = \text{dist}(A, P') \geq \varepsilon \; .$$

The tester rejects such an A with probability at least 2/3, as desired. □ Lemma 1

Testable properties remain testable when the vocabulary is extended. So it suffices to consider the minimal relevant vocabulary. Simple modifications of the proof of Lemma 1 give the corresponding results for $\mathcal{T}_r$ and $\mathcal{T}_{mr}$-style testing.

3.2 Comparing Models of Testability

In Subsection 2.3.1 above, we gave several distance definitions, each of which results in a model of relational property testing. In this section, we consider these different models and prove the relationship between them. The main result is Theorem 1, which shows that these models form a strict hierarchy.

Theorem 1. $\mathcal{T}_{mr} \subset \mathcal{T}_r \subset \mathcal{T}$.

That is, $\mathcal{T}_{mr}$-style testing is the most difficult, while $\mathcal{T}$-style testing is the easiest. Theorem 1 provides guidance for many of our results: when possible, we prefer to prove positive results in the strictest ($\mathcal{T}_{mr}$) model and negative results in the weakest ($\mathcal{T}$).

In addition, $\mathcal{T}_r$-testing is equivalent to the model of Fischer *et al.* [22], while $\mathcal{T}_{mr}$-testing is equivalent to the model of Austin and Tao [11]. Theorem 1 therefore provides some way of relating these results. However, we will see in Lemma 5 below that although in general these models are all distinct, $\mathcal{T}_r$ and $\mathcal{T}_{mr}$ are equivalent for many natural classes of properties.

3.2.1 Proof of Theorem 1

We begin the proof of Theorem 1 with the following simple lemma.

Lemma 2. *Let τ be a vocabulary and $A, B \in STRUC^n(\tau)$. Then,*

$$\mathrm{dist}(A,B) \leq \mathrm{rdist}(A,B) \leq \mathrm{mrdist}(A,B).$$

Proof (Lemma 2). We first show $\mathrm{dist}(A,B) \leq \mathrm{rdist}(A,B)$. If an ε-fraction of all assignments differs and we partition the assignments, there must be a partition such that at least an ε-fraction of the assignments differs in the partition. Let $\mathrm{dist}(A,B) = \varepsilon$ and let α_i be the fraction of R_i-assignments that differ between the structures,

$$\alpha_i := \frac{|\{\mathbf{x} \mid \mathbf{x} \in U^{a_i} \text{ and } R_i^A(\mathbf{x}) \oplus R_i^B(\mathbf{x})\}|}{n^{a_i}} .$$

Then, $\mathrm{rdist}(A,B) = \max_i \alpha_i$ and we can write $\mathrm{dist}(A,B)$ in terms of the α_i,

$$\mathrm{dist}(A,B) = \frac{\sum_i \alpha_i n^{a_i}}{\sum_i n^{a_i}} = \varepsilon .$$

This implies that $\sum_i \alpha_i n^{a_i} = \varepsilon \sum_i n^{a_i}$, and so there must be an $\alpha_i \geq \varepsilon$.

Next, we show that $\mathrm{rdist}(A, B) \leq \mathrm{mrdist}(A, B)$. The proof is nearly identical to the above. If $\mathrm{rdist}(A, B) = \varepsilon$ then there is an R_i such that an ε-fraction of the R_i-assignments differs between the structures. If we partition the R_i-assignments into the subtypes of R_i (which are disjoint), then there must be some partition such that at least an ε-fraction of the assignments in that partition differ. □ Lemma 2

Assume a tester distinguishes between structures A having some property P and those for which $\mathrm{mrdist}(A, P) \geq \varepsilon$. Lemma 2 trivially implies that it also distinguishes between structures A that have P and those for which $\mathrm{rdist}(A, P) \geq \varepsilon$. The case with rdist and dist is analogous, which proves the following.

Corollary 1. $\mathcal{T}_{mr} \subseteq \mathcal{T}_r \subseteq \mathcal{T}$.

Of course it is always desirable to show that such containments are strict. We show the separations by encoding the following language of binary strings; recall that $\overleftarrow{u}$ denotes the usual reversal of string u. It is also possible to use, e.g., one of the untestable properties that will be seen in Chapter 5 to prove the separations with a first-order expressible property that is closed under isomorphisms.

Theorem 2 (Alon *et al.* [5])**.** *Language* $L = \{u\overleftarrow{u}v\overleftarrow{v} \mid u, v$ *are strings over* $\{0, 1\}\}$ *is not testable.*

We are now ready to prove Theorem 1.

Proof (Theorem 1). The inclusions are by Corollary 1 and so only the separations remain. We first show that $\mathcal{T}\backslash\mathcal{T}_r$ is not empty. It suffices to give a vocabulary τ

and a τ-property that is $\mathcal{T}$-testable but not $\mathcal{T}_r$-testable. We use the vocabulary $\tau_C := (E^2, S^1)$.

We will show $P_1 \in \mathcal{T} \backslash \mathcal{T}_r$, where $P_1 \subseteq STRUC(\tau_C)$ is the set of structures where the S assignments encode the language L of Theorem 2. Recall that n denotes the size of the universe and our convention is that $S(i)$ is interpreted as "bit i of the string is 1". Therefore, A has P_1 if there is some $0 \leq k \leq n/2$ such that for all $0 \leq i < k$, $S(i)$ is true iff $S(2k - 1 - i)$ is true and for all $0 \leq j < (n - 2k)/2$, $S(2k + j)$ is true iff $S(n - 1 - j)$ is true. The property uses only the low-arity relation S; the E relation is for "padding" to make P_1 testable under the dist definition for distance.

We first show that P_1 is in $\mathcal{T}$. A structure with a universe of odd size cannot have P_1. A tester can begin by checking the parity of n and rejecting if it is odd and so we assume in the following that the size of the universe is even.

Lemma 3. *Property P_1 is testable under the* dist *definition for distance.*

Proof (Lemma 3). For any (even) n, 1^n is of the form $u\overleftarrow{u}v\overleftarrow{v}$. Changing all $S(i)$ assignments to true in any given A results in the string 1^n. This involves at most n modifications and so $\text{dist}(A, P_1) \leq \text{dist}(A, A') = O(n)/\Theta(n^2) < \varepsilon$, where the final inequality holds for sufficiently large n. Let $N(\varepsilon)$ be the smallest value of n for which it holds. The following is an ε-tester for P_1, where the input has universe size n.

1. If $n < N(\varepsilon)$, query all assignments and output whether the input has P_1.

2. Otherwise, accept.

If A has P_1, we accept with zero error. If $\text{dist}(A, P_1) \geq \varepsilon$, then $n < N(\varepsilon)$. In this case we query all assignments and reject with zero error. □ Lemma 3

It remains to show that P_1 is not testable when using the rdist definition for distance. We do this by showing that it would contradict Theorem 2.

Lemma 4. *Property P_1 is not testable under the* rdist *definition for distance.*

Proof (Lemma 4). Suppose there exist $\mathcal{T}_r$-type ε-testers T^ε for all $\varepsilon > 0$. The following is an ε-tester using *Definition* 4 for the language L of Theorem 2. Let the input be w, a binary string of length n.

1. Run T^ε and intercept all queries.
2. When a query is made for $S(i)$, return the value of $S(i)$ in w.
3. When a query is made for $E(i, j)$, return 0.
4. Output the decision of T^ε.

We run T^ε on the $A \in STRUC^n(\tau_C)$ that agrees with w on S and where all E assignments are false. If $w \in L$, then any such A has property P_1 and so our tester accepts with probability at least 2/3.

Assume $\text{dist}(w, L) \geq \varepsilon$. Then, $\text{rdist}(A, P_1) = \text{dist}(w, L) \geq \varepsilon$ and so our tester rejects with probability at least 2/3. These are testers for the untestable language of Theorem 2, and so P_1 is untestable under the rdist definition. □ Lemma 4

Lemmata 3 and 4, together with Corollary 1 show $\mathcal{T}_r \subset \mathcal{T}$. The separation $\mathcal{T}_{mr} \subset \mathcal{T}_r$ is shown in a similar way, using a property with sufficient "padding" to make $\mathcal{T}_r$ testing simple but $\mathcal{T}_{mr}$ testing would contradict Theorem 2.

For example, one can use the property P_2 of graphs in which the "loops" $E(i, i)$ encode the language from Theorem 2. That is, a graph has P_2 if there is some $0 \leq k \leq n/2$ such that for all $0 \leq i < k$, $E(i, i)$ is true iff $E(2k-1-i, 2k-1-i)$ is true and for all $0 \leq j < (n-2k)/2$, $E(2k+j, 2k+j)$ is true iff $E(n-1-j, n-1-j)$ is true. The non-loops are used as padding to ensure $\mathcal{T}_r$ testability while $\mathcal{T}_{mr}$ testability would allow us to violate Theorem 2. □ Theorem 1

There exist properties that are testable in the rdist sense but not in the mrdist sense. However, the definition of subtypes and $\mathcal{T}_{mr}$ testability allows for a simple mapping between vocabularies such that rdist-testability of certain *classes* of properties implies mrdist-testability of the same classes. For these classes, proving testability in the rdist sense is equivalent to proving it in the mrdist sense, and so it suffices to use whichever definition is more convenient.

Lemma 5 is given in the context of the classification problem for first-order logic but it is not difficult to prove similar results in other contexts. We will use Lemma 5 in Section 4.1, to prove the testability of a class that has this particular form.

Lemma 5. *Let $\mathcal{C} := [\Pi, all]_=$ be a prefix vocabulary class. Then, $\mathcal{C}$ is testable in the* rdist *sense iff it is testable in the* mrdist *sense.*

Proof. Recalling Theorem 1, $\mathcal{T}_{mr}$ testability implies $\mathcal{T}_r$ testability. We prove $\mathcal{T}_r$ testability of such prefix classes implies $\mathcal{T}_{mr}$ testability using Lemma 6. In the following, $\mathfrak{S}(n, k)$ is the Stirling number of the second kind.

Lemma 6. *Let $\mathcal{C} = [\Pi, (p_1, p_2, \ldots)]_=$ be a prefix vocabulary class and, furthermore, let $q_j = \sum_{i \geq j} p_i \mathfrak{S}(i, j)$. If $\mathcal{C}' = [\Pi, (q_1, q_2, \ldots)]_=$ is $\mathcal{T}_r$ testable, then $\mathcal{C}$ is $\mathcal{T}_{mr}$*

testable.

Proof (Lemma 6). Let $\varphi \in \mathcal{C}$ be arbitrarily fixed and assume that the predicate symbols of φ are $\{R^1_1, R^1_2, \ldots, R^1_{p_1}, R^2_1, \ldots\}$, where the arity of R^i_j is i. We construct a $\varphi' \in \mathcal{C}'$ and show that $\mathcal{T}_r$ testability of φ' implies $\mathcal{T}_{mr}$ testability of φ. In φ' we will use a distinct predicate symbol for each subtype of each R^i_j in φ. A subtype $\mathcal{S}$ of R^i_j such that $|\mathcal{S}| = k$ is a partition of the integers $\{1, \ldots, i\}$ into k non-empty sets and so there are $\mathfrak{S}(i,k)$ such subtypes. We therefore require a total of q_k distinct predicate symbols of arity k.

For example, we will map the "loops" in a binary predicate E to a new monadic predicate and the non-loops to a separate binary predicate. Formally, recall that s^U maps the subtypes of a predicate to the sets of tuples comprising the subtypes. For our example of a binary predicate, $(0,1) \in s^U(\{\{1\},\{2\}\})$ and $(0,0) \in s^U(\{\{1,2\}\})$. Next, we let r be a bijection from the subtypes of predicates to their new names, the predicate symbols that we will use in φ'.

We create φ' by modifying φ. Replace all occurrences of $R^i_j(x_1, \ldots, x_i)$ with

$$\left(\bigvee_{\mathcal{S} \in SUB(R^i_j)} \left[(x_1, \ldots, x_i) \in s^U(\mathcal{S}) \wedge r(\mathcal{S}, R^i_j)(\mathbf{y}) \right] \right).$$

Note that $(x_1, \ldots, x_i) \in s^U(\mathcal{S})$ is an abbreviation for a simple conjunction, e.g., $x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge \cdots$. Likewise, $\mathbf{y}$ is an $|\mathcal{S}|$-ary tuple, formed by removing the duplicate components of $(x_1, \ldots, x_i)$. The implicit mapping from $(x_1, \ldots, x_i)$ is invertible given $\mathcal{S}$. To continue our example of a binary predicate E, we would replace all occurrences of $E(x,y)$ in φ with

$$([x = y \land E_1(x)] \lor [x \neq y \land E_2(x,y)])\,.$$

We assume that φ' is $\mathcal{T}_r$ testable, and so there exists an ε-tester T^ε for it. We run this tester and intercept all queries. For a query to $r(\mathcal{S}, R^i_j)(\mathbf{y})$, we return the value of $R^i_j(x_1,\ldots,x_i)$. This is possible because r is a bijection, and so we can retrieve $\mathcal{S}$ and R^i_j using its inverse. Then, we can reconstruct the full i-ary tuple $(x_1,\ldots,x_i)$ from $\mathbf{y}$ and $\mathcal{S}$.

The tester implicitly defines a map[1] from structures A which we wish to test for φ to structures A' (with the same universe as A) which we can test for φ'.

Given an $A \models \varphi$, the corresponding $A' \models \varphi'$ and so T^ε will accept with probability at least $2/3$.

We map each subtype $\mathcal{S}$ to a distinct predicate symbol with arity $|\mathcal{S}|$. Therefore, for any structures A, B, the implicit mapping to A', B' is such that

$$\mathrm{mrdist}(A,B) = \mathrm{rdist}(A',B').$$

For convenience, let $P := \{B \mid B \models \varphi\}$ and $P' := \{B' \mid B' \models \varphi'\}$. For an A such that $\mathrm{mrdist}(A,P) \geq \varepsilon$, we simulate T^ε on an A' such that $\mathrm{rdist}(A',P') \geq \varepsilon$. The tester T^ε rejects with probability at least $2/3$, as desired. $\Box$ Lemma 6

Proving $\mathcal{T}_r$ testability for $[\Pi, all]_=$ implies proving it for all $(q_1,\ldots)$ that are

[1]Explicitly, map A to an A' with the same universe size, where $\mathbf{y} \in r(\mathcal{S}, R^i_j)$ in A' if $(x_1,\ldots,x_i) \in R^i_j$ in A. Note that we have not yet defined the assignments of tuples $\mathbf{y}$ with duplicate components. By construction, the assignments of these tuples do not affect φ' and so any reasonable convention will do. For example, for any predicate symbol symbol Q of φ' and any tuple $\mathbf{z}$ that has at least one duplicate component, we define $\mathbf{z} \notin Q$. The resulting map is injective but not necessarily surjective.

"images" of some $(p_1, \ldots)$ and so Lemma 6 is stronger than required. $\square$ Lemma 5

3.3 Indistinguishability

Indistinguishability is a notion introduced by Alon *et al.* [3] that we use in several of our proofs. Essentially, two τ-properties are indistinguishable if sufficiently large structures that have one of the properties become arbitrarily close to having the other (i.e., the limit of the distance goes to zero as the size of the structures increases).

Definition 14 (Alon *et al.* [3]). *Let $P_1, P_2 \subseteq STRUC(\tau)$ be τ-properties that are closed under isomorphisms. We say that P_1 and P_2 are* indistinguishable *if for every $\varepsilon > 0$ there exists an $N := N(\varepsilon) \in \mathbb{N}$ such that the following holds for all $n > N$. For every $A \in STRUC^n(\tau)$, if A has property P_1, then* $\text{mrdist}(A, P_2) < \varepsilon$ *and if A has P_2, then* $\text{mrdist}(A, P_1) < \varepsilon$.

The most important fact regarding indistinguishability is that it preserves testability.

Lemma 7 (Alon *et al.* [3]). *Let $P_1, P_2 \subseteq STRUC(\tau)$ be indistinguishable*[2] *τ-properties. Property P_1 is testable iff P_2 is testable.*

The proof by Alon *et al.* [3] extends to relational structures (from undirected loop-free graphs) without difficulty once we use mrdist in Definition 14. In fact, because the proof shows that one can construct an ε-tester for one of the properties by taking the majority vote of three runs of an $\varepsilon/2$-tester for the other property, the query complexities of the two properties are closely related.

[2]If we are interested only in *uniform* testability, then the properties must also be decidable.

3.4 Yao's Principle

There are a number of tools used to prove lower bounds for testing, including reductions to problems from communication complexity (cf. Blais *et al.* [13]). Yao's Principle (cf. Yao [69]), an interpretation of von Neumann's minimax theorem [53] in the context of probabilistic computation, is probably the most commonly used such tool, and several of our results in Chapter 5 rely on applications of this principle.

There are many variations of Yao's Principle; we use the following.

Principle 1 (Yao's Principle). *If there is an $\varepsilon \in (0,1)$ and a distribution over $\mathcal{G}^n$ such that all deterministic testers with complexity c have an error-rate greater than $1/3$ for property P, then property P is not testable with complexity c.*

The definition of "testable" is of course our usual one involving random testers. In applications, one usually seeks to show that for sufficiently large n and some increasing function $c := c(n)$, there is a distribution of inputs such that all deterministic testers with complexity c have error-rates greater than $1/3$.

For completeness, we prove our version of Principle 1. We prove only the direction of the minimax theorem that is required for our purposes; for a survey of minimax theorems and their proofs see Simons [65].

3.4.1 Proof of Principle 1

We begin by providing the definitions required to state Principle 1.

Definition 15. *A* deterministic tester *is a binary tree where each internal node is labeled with a non-negative integer, each leaf is labeled "accept" or "reject," and the two edges from a node are labeled* 0 *and* 1*.*

Given an input, we execute the tester as follows. Beginning at the root, we interpret the labels on internal nodes as the bit position of the input that will be queried. If the result of the query is 0, we follow the 0 edge and otherwise the 1 edge. When we reach a leaf we output the decision on the label. Note that it is equivalent to label the internal nodes with atomic formulae such as $E(0,1)$ as these are equivalent to specific bits of the input, the positions of which can be easily computed.

Definition 16. *The* complexity *of a deterministic tester is the number of internal nodes, including the root unless it is a leaf, on the longest path from the root to a leaf.*

The complexity of a deterministic tester is then the maximum number of queries that it makes. Our interest is limited to testers of finite complexity, i.e., those that output a decision in finite time. Without loss of generality, we can restrict our attention to balanced binary trees, by making extra, useless queries on the shorter paths in order to "pad" their lengths.

For testability, we are concerned only with the error-rate on inputs that either have our desired property or are ε-far from having the property. Therefore, we define the error-rate of a tester to be non-zero only on such examples. Because of this, it suffices to restrict our attention to distributions that give zero probability to the remaining "possible" inputs (those that do not have the property in question, but are also not ε-far from it).

We begin by proving the following direction of the minimax theorem. We say that a vector $\mathbf{x} \in \mathbb{R}^{|\mathbf{x}|}$ is a *probability vector* if each of its components is a non-negative real number and the sum of its components is 1. For $n > 0$, we let $\mathbb{P}^n$ be the set of probability vectors with n components,

$$\mathbb{P}^n := \left\{ \mathbf{x} \mid \mathbf{x} = (x_1, \dots, x_n) \in \mathbb{R}^n, x_i \geq 0 \text{ for all } 1 \leq i \leq n, \text{ and } \sum_{i=1}^{n} x_i = 1 \right\}.$$

It is well-known that equality holds in the following, however we restrict ourselves to stating and proving only the direction required for Principle 1, as mentioned above.

Theorem 3 (Minimax Theorem)**.** *Let M be an $a \times b$ matrix of non-negative reals and $X = \mathbb{P}^a$ and $Y = \mathbb{P}^b$ be the sets of all probability vectors with a and b components, respectively. Then,*

$$\max_{\mathbf{y} \in Y} \min_{\mathbf{x} \in X} \mathbf{x} M \mathbf{y}^T \leq \min_{\mathbf{x} \in X} \max_{\mathbf{y} \in Y} \mathbf{x} M \mathbf{y}^T. \tag{3.1}$$

Proof (Theorem 3). Let $\mathbf{y}^*$ be any of the arg max on the left in (3.1),

$$\mathbf{y}^* := \arg\max_{\mathbf{y} \in Y} \min_{\mathbf{x} \in X} \mathbf{x} M \mathbf{y}^T,$$

and $\mathbf{x}^*$ be any of the arg min on the left in (3.1),

$$\mathbf{x}^* := \arg\min_{\mathbf{x} \in X} \mathbf{x} M \mathbf{y}^{*T}.$$

Then, for any probability vector $\mathbf{x} \in X$, by the definition of minimum,

$$\mathbf{x}M\mathbf{y}^{*T} \geq \mathbf{x}^*M\mathbf{y}^{*T}. \tag{3.2}$$

Let $\mathbf{x}^+$ and $\mathbf{y}^+$ be any of the arg min and arg max on the right of (3.1), that is

$$\mathbf{x}^+ := \arg\min_{\mathbf{x}\in X}\max_{\mathbf{y}\in Y}\mathbf{x}M\mathbf{y}^T \text{ and } \mathbf{y}^+ := \arg\max_{\mathbf{y}\in Y}\mathbf{x}^+M\mathbf{y}^T.$$

Then, by the definition of maximum, $\mathbf{x}^+M\mathbf{y}^{+T} \geq \mathbf{x}^+M\mathbf{y}^{*T}$. Therefore, by (3.2),

$$\min_{\mathbf{x}\in X}\max_{\mathbf{y}\in Y}\mathbf{x}M\mathbf{y}^T = \mathbf{x}^+M\mathbf{y}^{+T} \geq \mathbf{x}^+M\mathbf{y}^{*T} \geq \mathbf{x}^*M\mathbf{y}^{*T} = \max_{\mathbf{y}\in Y}\min_{\mathbf{x}\in X}\mathbf{x}M\mathbf{y}^T.$$

□ Theorem 3

Given Theorem 3, it is easy to show Principle 1. In (3.1), we call $\mathbf{x}$ on the left and $\mathbf{y}$ on the right "inner vectors." This is because we can think of them as being chosen after the "outer vectors" ($\mathbf{y}$ on the left and $\mathbf{x}$ on the right) are fixed. For the "inner" vectors, it suffices to consider unit vectors $\mathbf{e}_i$, where component i is 1 and all other elements are 0. This is because once the outer vector is fixed, denoting the i-th component of a vector $\mathbf{z}$ by $[\mathbf{z}]_i$,

$$\min_{\mathbf{x}\in X}\mathbf{x}M\mathbf{y}^T = \mathbf{e}_{\arg\min_i[M\mathbf{y}^T]_i}M\mathbf{y}^T$$

and likewise for the maximum on the right of (3.1). This proves the following simple corollary of Theorem 3.

Corollary 2. *Let M be an $a \times b$ matrix of non-negative reals and $X = \mathbb{P}^a$ and*

$Y = \mathbb{P}^b$ be the sets of all probability vectors with a and b components, respectively. Then,

$$\max_{\mathbf{y}\in Y} \min_{\mathbf{e}_i \in X} \mathbf{e}_i M \mathbf{y}^T \leq \min_{\mathbf{x}\in X} \max_{\mathbf{e}_j \in Y} \mathbf{x} M \mathbf{e}_j^T. \tag{3.3}$$

Proof (Principle 1). Principle 1 is an interpretation of (3.3) in the context of testing. We let a be the number of deterministic testers with complexity c whose queries are evaluable in structures of type τ that have n elements. We assume that there is an enumeration of these testers. Then, a randomized tester with complexity c for structures of n elements is given by a probability vector $\mathbf{x} \in X$, where $[\mathbf{x}]_i$ is interpreted as the probability that the randomized tester behaves like the i-th deterministic tester. A unit vector $\mathbf{e}_i \in X$ specifies the i-th deterministic tester.

Likewise, we assume there is an enumeration of structures of type τ that have n elements and let b be the number of such structures. Then, a probability vector $\mathbf{y} \in Y$ is a distribution over these structures and a unit vector $\mathbf{e}_j$ specifies the j-th structure.

For matrix M, we let M_{ij} be 1 if the i-th deterministic tester is incorrect on the j-th input and this input either has the desired property or is ε-far from it. Otherwise, we let M_{ij} be 0.

We now have have an $a \times b$ matrix M and a meaning for a and b component probability vectors and so we can interpret the meaning of Corollary 2. On the left, $\mathbf{e}_i M \mathbf{y}^T$ is the average-error of the i-th deterministic tester on a structure chosen according to distribution $\mathbf{y}$. Likewise, on the right, $\mathbf{x} M \mathbf{e}_j^T$ is the error-rate of the randomized tester specified by $\mathbf{x}$ on the j-th structure.

Therefore, the left side of (3.3) is the average-error of the "best" deterministic tester on the "worst" distribution of inputs, when we define "best" as the lowest average-error. If we find some distribution $\mathbf{y}^+$ of inputs such that all deterministic testers have an error-rate greater than 1/3, then 1/3 is a lower bound on the left side, and therefore the right side, of (3.3).

The right side of (3.3) is the error-rate of the "best" randomized tester on the "worst" input structure, when "best" is defined as the best worst-case. If the "best" randomized tester with complexity c has an error-rate greater than 1/3 on an input, we can conclude that the property in question is not testable with complexity c.

$\square$ Principle 1

3.5 Summary

In this chapter, we proved various basic results that we will use in later chapters. In particular, Theorem 1 relates the three models of testability that we introduced in Section 2.3. This encourages us to prove positive results in the strongest model (i.e., we will focus on $\mathcal{T}_{mr}$ in Chapter 4) and negative results in the weakest model (i.e., we will focus on $\mathcal{T}$ in Chapter 5).

Chapter 4

Testable Classes

We are now ready to prove the testability of two large, syntactic subclasses of first-order logic. We prove the testability of Ackermann's class with equality ($[\exists^*\forall\exists^*, all]_=$) in Section 4.1, and of Ramsey's class ($[\exists^*\forall^*, all]_=$) in Section 4.2.

4.1 Ackermann's Class with Equality

In this section we show that Ackermann's class with equality ($[\exists^*\forall\exists^*, all\,]_=$) is testable. We begin by reviewing the history of this class, which has a number of nice properties.

Ackermann's class was first considered (without equality) by Ackermann [1], who showed that the satisfiability problem for the class is decidable and that it has the finite model property[1]. Kolaitis and Vardi [45] showed the satisfiability problem for Ackermann's class with equality is complete for NEXPTIME and

[1]A class is said to have the *finite model property* if every satisfiable formula in the class has a finite model. Classes without this property have *infinity axioms*, i.e., sentences with only infinite models.

that a 0-1 law holds for existential second-order logic[2] where the first-order part belongs to $[\exists^*\forall\exists^*, \mathit{all}]_=$. Lewis [48] proved that satisfiability for Ackermann's class *without* equality is complete for (deterministic) EXPTIME. Grädel [31] showed that satisfiability for Ackermann's class without equality is complete for EXPTIME even with the addition of arbitrarily-many function symbols.

If we allow equality and a unary function symbol, the result is Shelah's class, which Shelah [64] proved decidable. Shelah's class is a decidable class that does not have the finite model property, and it would be interesting to determine if it is testable. This would require extending relational testing to allow function symbols.

Ackermann's class with equality has been studied in other settings as well. For example, Fermüller and Salzer [18] used an extension of resolution to decide an extension of Ackermann's class with equality using automated theorem provers.

The main goal of this subsection is Theorem 4 below. Recalling Theorem 1, this also implies that such properties are testable in the dist and rdist senses. If the vocabulary consists of a single relation, the rdist and dist definitions are equivalent to the dense hypergraph model. We therefore obtain the corresponding results in the dense hypergraph and dense graph models as special cases.

We denote the set of monadic predicate symbols in a vocabulary τ by $M := \{R_i \mid R_i \in \tau \text{ and } a_i = 1\}$. The set of assignments of the symbols in M for an element in a universe is called the *color* of the element and there are $2^{|M|}$ possible

[2]A class C of first-order logic has an associated 0-1 law if all existential *second*-order sentences $\varphi := \exists C_1 \dots C_a \psi$, where ψ is a first-order sentence in C, have the property that the limit as $n \to \infty$ of the probability that a random structure of size n satisfies φ exists and is either 0 or 1. Recall that the focus is on existential *second*-order because all of first-order admits a 0-1 law, see the references in Kolaitis and Vardi [46].

colors. We define $\mathrm{Col}(A, c)$ to be the set of colors that occur at least c times in A.

Theorem 4. *All formulae in* $[\exists^*\forall\exists^*, all]_=$ *define properties that are in* $\mathcal{T}_{mr}$ *with one-sided error.*

Proof (Theorem 4). Recall that Ackermann's class with equality is $[\exists^*\forall\exists^*, all]_=$ and, therefore, it suffices to show the testability of property P of type $\tau = (R_1^{a_1}, \ldots, R_s^{a_s})$ defined by formula $\varphi := \exists x_1 \ldots \exists x_a \forall y \exists z_1 \ldots \exists z_b : \psi$, where ψ is quantifier-free. Note that a is the number of leading existential quantifiers and b is the number of trailing existential quantifiers. We can trivially test any φ that has only finitely-many models with a constant number of queries and zero error, and so it suffices to assume that φ has infinitely-many models.

The class $[\exists^*\forall\exists^*, all]_=$ is of the form required by Lemma 5 above, and so it is mrdist-testable iff it is rdist-testable. It therefore suffices to show that P is testable in the rdist sense. We will show that the following is an ε-tester in the rdist sense for P on input $A \in STRUC^n(\tau)$. Here, $k := k(\tau, \varepsilon)$ is the number of elements queried and $N := N(\varphi, \tau, \varepsilon)$ is a constant, both of which are determined below. Note the actual number of *queries* in Step 2 is not exactly k, but rather a constant multiple of it depending on τ. Finally, we explicitly give $\kappa := \kappa(\varphi, \tau)$ below.

1. If $n < N$, query all of A and decide exactly whether A has P.

2. Uniformly and independently choose k members of the universe of A and query all monadic predicates on the members in this sample. Let B be the observed substructure.

3. Search over all $A' \in STRUC^{\kappa}(\tau)$. Accept if an A' is found such that $A' \models \varphi$ and $\mathrm{Col}(B, a+1) \subseteq \mathrm{Col}(A', a+1)$.

4. Otherwise, reject.

We will show that the tester accepts (with probability 1) if $A \models \varphi$ and rejects with probability 2/3 if $\mathrm{rdist}(A, P) > \varepsilon$. We first show that if $A \models \varphi$, then the tester is guaranteed to accept. Then, we will show in Lemma 9 that with probability at least 2/3, we get a "good" sample in Step 2. A sample is "good" if it contains at least $(a+1)$-many distinct representatives of each color that occurs on at least an $\varepsilon/(2 \cdot 2^{|M|})$ fraction of the elements of A. We then show that the tester is correct if it obtains a good sample, and therefore rejects with probability at least 2/3 if $\mathrm{rdist}(A, P) > \varepsilon$.

We will now show that if $A \models \varphi$, the tester will accept with probability 1. We begin with Lemma 8.

Lemma 8. *Let A be a model of φ such that $\#(A) > N$ and let*

$$\kappa := a + 3b\left(a + 2^{\sum_{i=1}^{s}\sum_{j=1}^{a_i}\binom{a_i}{j}a^{a_i-j}}\right) + 2^{|M|}(a+1).$$

Then, there is an $A' \models \varphi$ such that $\#(A') = \kappa$ and $\mathrm{Col}(A, a+1) \subseteq \mathrm{Col}(A', a+1)$.

Proof. Assume that $N > \kappa$. The structure A is a model of φ, and so there exists at least one tuple of a elements $(u_1, \ldots, u_a)$ such that φ is satisfied when the existential quantifiers bind u_i to x_i. We consider the x_i and the substructure induced by them to be fixed, and refer to this substructure as A_x.

There are at most $\kappa_2 := a + 2^{\sum_{i=1}^{s}\sum_{j=1}^{a_i}\binom{a_i}{j}a^{a_i-j}}$ many *distinct* structures constructed by adding an element labeled y to A_x when we include the structures where the label y is simply placed on one of the x_i. We let $v \leq \kappa_2$ be the number of such structures that occur in A and assume there is an enumeration of them.

For each of these v substructures there exist b elements, $w_1, \ldots, w_b$, such that when we label w_i with z_i, the substructure induced by $(x_1, \ldots, x_a, y, z_1, \ldots, z_b)$ models ψ. We construct $A_{i,j}$ for $1 \leq i \leq 3$ and $1 \leq j \leq v$ such that $A_{i,j}$ is a copy of the $w_1, \ldots, w_b$ used for the j-th structure (see Figure 4.1). We connect each $A_{i,j}$ to A_x in the same way as in A, modifying assignments on tuples $(A_x \cup A_{i,j})^{a_k}$.

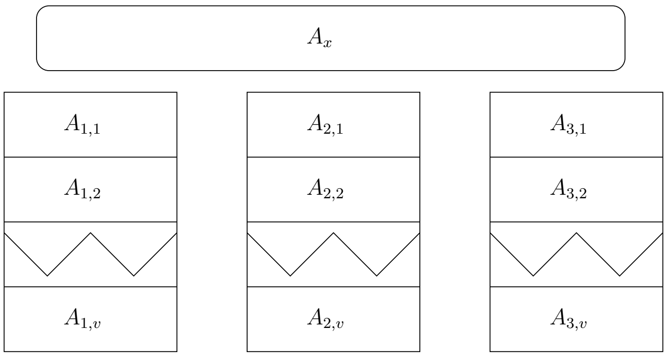


Figure 4.1. A sketch of the new structure

For each w_h in $A_{i,j}$, we consider the case where y is bound to w_h. By construction the substructure induced by $(x_1, \ldots, x_a, y)$ occurs in A. We assume it is the g-th structure and use the elements of $A_{i+1 \bmod 3, g}$ to construct a structure satisfying ψ. We modify the assignments of tuples as needed to create a structure identical to that in A satisfying ψ. Note that by construction all of these assignments are of tuples that contain w_h and at least one element from $A_{i+1 \bmod 3, g}$. The resulting structure, which we call A_1, is a model of φ. Before this step we have not modified any assignments "spanning" the "columns" $A_{i,j}$ of A_1 and so there are no assignments that we modify more than once.

However, there may be some color from $\mathrm{Col}(A, a+1)$ that does not appear $a+1$ times in A_1. We therefore add a new block, denoted A_e, of at most $2^{|M|}(a+1)$ elements which consists of $a+1$ copies of each color from $\mathrm{Col}(A, a+1)$. Each of these colors occurred at least $a+1$ times in A, and so for each such color C, there is an element q in A with color C such that q is *not* part of A_x. If the substructure induced by (A_x, q) in A is the j-th structure in our enumeration, then we do the following for each member p of A_e that has the same color as q. First, we make the substructure induced by (A_x, p) identical to that induced by (A_x, q) in A. Next, we make the substructure induced by $(p, A_{1,j})$ identical to that induced by q and the corresponding z_i in A. All of these modifications are on tuples containing a $p \in A_e$ and so we do not modify any tuples more than once. We call this structure A_2.

Finally, so far we only have an upper-bound on the size of A_2 while the lemma states it to be *exactly* of size κ. We therefore pad in the following simple way[3]. We know that $N > \kappa > 2^{|M|}a$ and so there is a color that occurs at least $a+1$ times in A. If $\#(A_2) < \kappa$, we simply make an additional $\kappa - \#(A_2)$ many copies of this color in A_e and modify the assignments of tuples containing these new elements in the same manner as above. The resulting A' has size κ and satisfies the requirements of the lemma. $\square$ Lemma 8

For any sample B of A, it is true that $\mathrm{Col}(B, a+1) \subseteq \mathrm{Col}(A, a+1)$. If $A \models \varphi$, then Lemma 8 implies that our tester will find an A' satisfying the conditions of Step 3 and will therefore accept. This holds for any sample B and so the tester will accept such A with probability 1.

[3]One could instead change the tester to search structures with size at most κ.

Next, assume that $\mathrm{rdist}(A, P) \geq \varepsilon$. In this case we must show that the tester rejects with probability at least $2/3$. First, we show that the tester obtains a "good" sample with probability at least $2/3$.

Lemma 9. *There are constants k and N such that, with probability at least $2/3$, the tester obtains a sample that contains at least $(a+1)$-many distinct representatives of each color in $\mathrm{Col}(A, \varepsilon n/(2 \cdot 2^{|M|}))$.*

Proof (Lemma 9). The probability that any particular query misses a fixed color that occurs on at least an $\varepsilon/(2 \cdot 2^{|M|})$ fraction of A is at most $(1 - \varepsilon/(2 \cdot 2^{|M|}))$. Moreover, the probability that we miss such a fixed color after k_1 independent queries is at most $(1-\varepsilon/(2 \cdot 2^{|M|}))^{k_1}$. There are at most $2^{|M|}$ such colors, and so the probability that a sample of k_1 elements fails to contain at least one representative of all such colors is at most

$$p_1 := 2^{|M|} \left(1 - \frac{\varepsilon/2}{2^{|M|}}\right)^{k_1}.$$

The $|M|$ is a constant, and we choose k_1 such that $(a+1)p_1$ is at most $1/6$.

Let $k := (a+1)k_1$. We will make k independent queries, and consider the total sample as $(a+1)$ separate samples of size k_1. All of these smaller samples will contain at least one representative of every color in $\mathrm{Col}(A, \varepsilon n/(2 \cdot 2^{|M|}))$ with probability at least $1 - (a+1)p_1 \geq 5/6$. However, there is the possibility that some of these smaller samples contain elements in common. We will choose N such that for $n > N$, the probability that any particular element in the universe of A is chosen more than once is at most $1/6$. In particular, if $\#(A) = n$ and we define $x^{\underline{y}} := x(x-1)\cdots(x-(y-1)) = x!/(x-y)!$, then the probability that some element is queried more than once is

$$p_2 := 1 - \frac{n!}{n^{(a+1)k_1}(n-(a+1)k_1)!} = 1 - \frac{n^{\underline{(a+1)k_1}}}{n^{(a+1)k_1}}.$$

The sample size $(a+1)k_1$ is a constant, and so we can choose N such that $p_2 \leq 1/6$ for $n > N$.

The probability that the tester obtains a sample that contains at least $(a+1)$-many distinct representatives of each color in $\mathrm{Col}(A, \varepsilon n/(2 \cdot 2^{|M|}))$ is at least

$$1 - (a+1)p_1 - p_2 \geq 2/3.$$

□ Lemma 9

Our goal is to show that if $\mathrm{rdist}(A, P) \geq \varepsilon$, then we reject with probability at least 2/3. It is easier to show the contrapositive: if the tester accepts with probability strictly greater than 1/3, then $\mathrm{rdist}(A, P) < \varepsilon$.

If we accept a structure A with probability strictly greater than 1/3, then we must accept it when we obtain a good sample. We construct a B such that $B \models \varphi$ and $\mathrm{rdist}(A, B) < \varepsilon$ from the A' that the tester must find to accept. We begin with Lemma 10, which we will use to "grow" smaller models.

Lemma 10. *Let $\varphi := \exists x_1 \ldots \exists x_a \forall y \exists z_1 \ldots \exists z_b : \psi$ be a formula with vocabulary τ, where ψ is quantifier-free and $A \in STRUC(\tau)$ be such that $A \models \varphi$. Additionally, let $B \in STRUC(\tau)$ be any structure containing A as an induced substructure such that $\#(B) = \#(A) + 1$. If the additional element of B has a color that occurs at least $a+1$ times in A, then we can construct a $B' \models \varphi$ by modifying at most a* *constant number of non-monadic assignments in B.*

Proof (Lemma 10). Structure B contains an induced copy of A and one additional element, which we will denote by q. By assumption, A is a model of φ and therefore contains an a-tuple $(u_1, \dots, u_a)$ such that the formula is satisfied when x_i is bound to u_i. In addition, there are at least $a+1$ elements in A that have the same color as q. Therefore, there is at least one such element p that is not one of the u_i. We will make q equivalent to p without modifying any monadic assignments.

We begin by modifying the assignments as needed to make the structure induced by $(x_1, \dots, x_a, q)$ identical to that induced by $(x_1, \dots, x_a, p)$. This requires at most $\sum_{i=1}^{s} \sum_{j=1}^{a_i} \binom{a_i}{j} a^{a_i - j} = O(1)$ modifications, all of which are non-monadic. There must be $(v_1, \dots, v_b)$ in A such that ψ is satisfied when z_i is bound to v_i and y to p. We modify the assignments needed to make the structure induced by $(q, v_1, \dots, v_b)$ identical to that induced by $(p, v_1, \dots, v_b)$[4]. This requires at most $\sum_{i=1}^{s} \sum_{j=1}^{a_i} \binom{a_i}{j} b^{a_i - j} = O(1)$ modifications, all of which are non-monadic. The result has $\#(A)+1$ elements, models φ and was constructed from B by making a constant number of modifications to non-monadic assignments. $\square$ Lemma 10

Let A be the structure that the tester is running on and A' be the structure found in Step 3 of the tester. As mentioned above, we will construct a $B \models \varphi$ from A' such that $B \models \varphi$ and $\mathrm{rdist}(A, B) < \varepsilon$.

Note that there must exist at least one color in $\mathrm{Col}(A, \varepsilon n/(2 \cdot 2^{|M|}))$ and assume that N is large enough that $\varepsilon n/(2 \cdot 2^{|M|}) \geq a+1$. We first make a constant-sized portion of A identical to A'. This requires at most $O(1)$-many modifications to

[4]The case where $v_i = p$ can be handled by replacing v_i with q in $(q, v_1, \dots, v_b)$.

each relation. All colors in $\mathrm{Col}(A, \varepsilon n/(2 \cdot 2^{|M|}))$ occur at least $a+1$ times in A', allowing us to recursively apply Lemma 10 and add the elements of A that have colors in $\mathrm{Col}(A, \varepsilon n/(2 \cdot 2^{|M|}))$. This entails making $O(1)$-many modifications to non-monadic relations (and none to monadic relations) at each step, for a total of $O(n)$ modifications to the non-monadic relations.

Finally, we consider the elements of A that have colors which occur at most $\varepsilon n/(2 \cdot 2^{|M|})$ times. There are at most $2^{|M|}$ such colors and at most $\varepsilon n/2$ elements with these colors. We change the monadic assignments on such elements as required to give them colors contained in $\mathrm{Col}(A, \varepsilon n/(2 \cdot 2^{|M|}))$. This requires at most $\varepsilon n/2$ modifications to each of the monadic assignments. We again recursively apply Lemma 10 to A, making $O(1)$ modifications to non-monadic assignments at each step. The resulting structure is B and is such that $B \models \varphi$.

Finally, we show that $\mathrm{rdist}(A, B) < \varepsilon$. If R_i is a monadic relation, then the i-th term of the maximum in the definition of rdist (cf. Definition 5 above) is at most $\varepsilon/2 + o(1)$. If R_i has arity at least two, then the i-th term of the maximum is $O(n)/\Omega(n^2) = o(1)$. All $o(1)$ terms can be made arbitrarily small by choosing $N(\varphi, \tau, \varepsilon)$ appropriately and so we can assume that all terms are strictly less than ε. The maximum is then strictly less than ε and so $\mathrm{rdist}(A, B) < \varepsilon$ as desired.

$\square$ Theorem 4

4.2 Ramsey's Class

In this section we revisit a result of Alon *et al.* [3] in the light of recent work by Austin and Tao [11]. The main result is the testability of the full Ramsey's class (i.e., removing the restriction to undirected loop-free graphs). As we did for Ackermann's class with equality in Section 4.1, we begin by reviewing the history and properties of the class, denoted $[\exists^*\forall^*, \mathit{all}\,]_=$.

Ramsey's class is also known as the Bernays-Schönfinkel-Ramsey class. Bernays and Schönfinkel [12] proved the finite model property and that satisfiability is decidable for the class *without* equality. Ramsey [56] extended these results to the class with equality as part of a stronger result. Lewis [48] showed that satisfiability is NEXPTIME-complete for Ramsey's class and Kolaitis and Vardi [44] proved that a 0-1 law holds for existential second-order logic where the first-order part belongs to $[\exists^*\forall^*, \mathit{all}\,]_=$. Omodeo and Policriti [54] have recently shown that the class is semidecidable for set theory.

The main goal of this subsection is Theorem 5 below. Recalling Theorem 1, this also implies testability in the $\mathcal{T}$ and $\mathcal{T}_r$ senses. The proof of Theorem 5 follows the proof by Alon *et al.* [3], and relies on a reduction to a strong result by Austin and Tao [11].

An outline of the proof is as follows. First, we show that all sentences in $[\exists^*\forall^*, \mathit{all}\,]_=$ define properties which are indistinguishable from instances of a generalized colorability problem. Next, we note that all such problems are hereditary and therefore testable when mapped to the setting defined by Austin and Tao [11]. Finally, we show that this implies testability under our definitions, which gives the following.

Theorem 5. *All sentences in $[\exists^*\forall^*, \mathit{all}]_=$ define properties in $\mathcal{T}_{mr}$.*

We begin the proof of Theorem 5 by defining a generalized colorability problem, as did Alon *et al.* [3].

For any fixed set F of structures with vocabulary τ, some positive number of colors c, and functions that assign a color between 1 and c to each element of each structure in F, we define the F-colorability problem as follows. A structure $A \in STRUC(\tau)$ is F-colorable if there exists some (not necessarily proper) c-coloring of A such that A does not contain any induced substructures isomorphic to a member of F. We let P_F be the set of structures that are F-colorable.

For example, we can consider the case of graphs and let F contain c copies of K_2. We enumerate these copies in some fashion from 1 to c, and for copy i, color both vertices with i. The resulting problem is of course the usual (k- or equivalently) c-colorability. The following is a straightforward generalization of the proof by Alon *et al.* [3].

Lemma 11. *Let φ be any first-order sentence in the class $[\exists^*\forall^*, all]_=$. There exists an instance of the F-colorability problem that is indistinguishable from P, the property defined by φ.*

Proof (Lemma 11). Let $\varepsilon > 0$ be arbitrary and $\varphi := \exists x_1 \dots \exists x_t \forall y_1 \dots \forall y_u : \psi$ be any first-order formula with quantifier-free ψ and vocabulary τ. We note, as did Alon *et al.* [3], that we can restrict our attention to formulae ψ where it is sufficient to consider only cases where the variables are bound to distinct elements. This is because, given any ψ', we can construct a ψ satisfying this restriction that is equivalent on structures with at least $t + u$ elements, and the smaller structures do not matter in the context of indistinguishability.

Let $P = \{A \mid A \in STRUC(\tau), A \models \varphi\}$ be the property defined by φ. We now define an instance of F-colorability that we will show to be indistinguishable from P. We denote our c colors by the elements of

$$\{(0,0)\} \cup \{(a,b) \mid 1 \leq a \leq \pi_1, 1 \leq b \leq \pi_2, a,b \in \mathbb{N}\}\,.$$

Here, π_1 is the number of distinct structures of vocabulary τ with exactly t elements, $\pi_1 := 2^{\sum_{i=1}^{s} t^{a_i}}$. Similarly, we denote by π_2 the number of ways it is possible to "connect" or "add" a single element to some existing, fixed t-element structure of vocabulary τ, i.e., $\pi_2 := 2^{\sum_{i=1}^{s}\sum_{j=1}^{a_i-1}\binom{a_i}{j}t^{a_i-j}}$. We will use fixed enumerations of these π_1 structures with t elements and π_2 ways of connecting an additional element to a fixed t element structure.

We impose on the coloring of the structure the following restrictions. Each can be expressed by prohibiting finite sets of colored induced substructures.

(1) The color $(0,0)$ may be used at most t times. Therefore, we prohibit all $(t+1)$-element structures that are colored completely with $(0,0)$.[5]

(2) The graph must be colored using only $\{(0,0)\} \cup \{(a,b) \mid 1 \leq b \leq \pi_2\}$ for some fixed $a \in \{1,\ldots,\pi_1\}$. Therefore, we prohibit all two-element structures colored $((a,b),(a',b'))$ with $a \neq a'$.

(3) We now consider some fixed coloring of a u-element structure V, whose universe we identify with $\{v_1,\ldots,v_u\}$. We assume that this coloring satisfies the previous restriction and that color $(0,0)$ does not appear. We must decide whether to prohibit this structure. In order to do so, we first take the

[5]Note that introducing a constraint guaranteeing the existence of t such elements cannot be done by forbidding finite sets of structures, and would result in a non-hereditary property.

fixed a guaranteed by the previous restriction, and consider the t-element structure E, whose universe we identify with $\{e_1, \ldots, e_t\}$, that is the a^{th} structure in our enumeration of t element structures. We connect each v_i to E in the following way. If v_i is colored (a, b), we use the b^{th} way of connecting an additional element to a t-element structure in our enumeration. We denote the resulting $(t + u)$-element structure as M and allow (do not prohibit) V iff M is a model of ψ when we replace x_i with u_i and y_j with v_j.

We now show that the resulting F-colorability problem is indistinguishable from P. Recall the definition of indistinguishability (Definition 14) and assume that we are given an $A \models \varphi$. Color the t vertices existentially bound to the x_i with $(0, 0)$. Then, we can color all remaining vertices v_i with (a, b), where a corresponds to the substructure induced by $\{x_1, \ldots, x_t\}$ in our enumeration of t-element structures, and b corresponds to the connection between v_i and $\{x_1, \ldots, x_t\}$. It is easy to see that this coloring satisfies the restrictions of our F-colorability problem. We have not made any modifications to the structure and so clearly $\text{mrdist}(A, P_F) = 0$ (i.e., $A \in P_F$).

Next, we assume that we are given a structure with a coloring that satisfies our restrictions. We will show that we can obtain a model of φ by making only a small number of modifications. First, if there are less than t elements colored $(0, 0)$, we arbitrarily choose additional elements to color $(0, 0)$ so that there are exactly t such elements. We will denote these t elements with $\{e_1, \ldots, e_t\}$. Restriction (2) guarantees that all colors which are not $(0, 0)$ share the same first component. Let a be this shared component. We make the structure induced by $\{e_1, \ldots, e_t\}$ identical to the a^{th} structure in our enumeration of t-element structures, requiring at most $\sum_{i=1}^{s} t^{a_i} = O(1)$ modifications. Next, for each element v_i that is colored (a, b) with $a, b \neq 0$, we modify the connections between v_i and $\{e_1, \ldots, e_t\}$ in order to make these connections identical to the b^{th} way of making such connections in our enumeration. This requires at most

$$(n-t) \sum_{i=1}^{r} \sum_{j=1}^{a_i - 1} \left[\binom{a_i}{j} t^{a_i - j} \right] = O(n)$$

additional modifications, all of which are to non-monadic subrelations. Binding x_i to e_i, the resulting structure is a model of φ. We made at most $O(1)$ modifications to monadic subrelations and $O(n)$ modifications to non-monadic subrelations, and so $\text{mrdist}(A, P) \leq \max\{O(1)/n, O(n)/\Omega(n^2)\} = o(1) < \varepsilon$, where the inequality holds for sufficiently large n.

Therefore, all such properties P are indistinguishable from instances of F-colorability, as desired. □ Lemma 11

Recall that a *hereditary property* of relational structures is one which is closed under taking induced substructures. F-colorability is clearly a hereditary property; if A is F-colorable, then so are its induced substructures. However, the definitions of Austin and Tao [11] are significantly different from ours and so we explicitly reduce the following translation in our setting to their result.

Theorem 6 (Translation of Austin and Tao)**.** *Let P be a hereditary property of relational structures which is closed under isomorphisms. Then, property P is testable in the sense of $\mathcal{T}_{mr}$ with one-sided error.*

Theorem 6 is, in a sense, the latest in a series of generalizations of Corollary 6.3 of Alon *et al.* [3], i.e., the testability of colorability problems for undirected loop-free graphs. The first such generalization was by Fischer [20], who extended the result to more general colorability problems with counting restrictions. This was followed by Alon and Shapira [8] who extended it to hereditary graph properties. Ishigami [37] extended the testability result to hereditary partite uniform hypergraph properties, and Rödl and Schacht [57] extended it to hereditary uniform hypergraph properties. These generalizations are closely related to extensions of Szemerédi's Regularity Lemma and the Removal Lemma, see the references in Section 1.1.

Before reducing Theorem 6 to its statement in [11], we first briefly introduce their definitions. All of the definitions in Subsection 4.2 are from Austin and Tao [11], although we omit definitions which are not necessary for our purposes.

Framework of Austin and Tao

We begin by introducing their analogue of vocabularies: *finite palettes.*

Definition 17. *A* finite palette K *is a sequence* $K := (K_j)_{j=0}^{\infty}$ *of finite sets, of which all but finitely-many are singletons. The singletons are called* points *and denoted* pt. *A point is called* trailing *if it occurs after all non-points.*

We will write $K = (K_0, \ldots, K_k)$, omitting trailing points and call k the *order* of K. We use the elements of K_j to *color* the j-ary edges in hypergraphs.

Definition 18. *A* vertex set V *is any set which is at most countable. If* V, W *are vertex sets, then a* morphism f *from* W *to* V *is any injective map* $f \colon W \to V$ *and the set of such morphisms is denoted* $\mathrm{Inj}(W, V)$. *For* $N \in \mathbb{N}$*, we denote the set* $\{1, \ldots, N\}$ *by* $[N]$.

Of course, $[N]$ is a vertex set. Our structures are finite so we are mostly interested in *finite* vertex sets. Next, we define the analogue of relational structures.

Definition 19. *Let V be a vertex set and K be a finite palette. A K-*colored hypergraph *G on V is a sequence $G := (G)_{j=0}^{\infty}$, where each $G_j \colon \mathrm{Inj}([j], V) \to K_j$ is a function. Let $K^{(V)}$ be the set of K-colored hypergraphs on V.*

Only finitely many of the K_j are not points, and so only finitely many G_j are non-trivial. The G_j assign colors from K_j to the morphisms in $\mathrm{Inj}([j], V)$. In our relational setting, this set of morphisms corresponds to the set of j-ary tuples $(x_1, \ldots, x_j)$ with pairwise distinct components.

Before defining hereditary K-properties, we need one last technical definition.

Definition 20. *Let V, W be vertex sets and $f \in \mathrm{Inj}(W, V)$ be a morphism from W to V. The* pullback map *$K^{(f)} \colon K^{(V)} \to K^{(W)}$ is*

$$\left(K^{(f)}(G)\right)_j (g) := G_j(f \circ g)\,,$$

for all $G = (G_j)_{j=0}^{\infty} \in K^{(V)}$, $j \geq 0$ and $g \in \mathrm{Inj}([j], W))$. If $W \subseteq V$ and $f \in \mathrm{Inj}(W, V)$ is the identity map on W, we abbreviate

$$G \downharpoonleft_W := K^{(f)}\,.$$

Abusing notation, the pullback map $K^{(f)}$ maps K-colored hypergraphs on V to those on W, by assigning the color of $f \circ g$ to g, for all tuples g. Note that $G \downharpoonleft_W$ is equivalent to the induced subhypergraph on W. For clarity, we reserve P for properties of relational structures and use $\mathcal{P}$ to denote properties of hypergraphs.

Definition 21. *Let $K = (K_j)_{j=0}^{\infty}$ be a finite palette. A* hereditary K-property $\mathcal{P}$ *is an assignment $\mathcal{P}\colon V \mapsto \mathcal{P}^{(V)}$ of a collection $\mathcal{P}^{(V)} \subseteq K^{(V)}$ of K-colored hypergraphs for every finite vertex set V such that for every morphism $f \in \mathrm{Inj}(W, V)$ between finite vertex sets,*

$$K^{(f)}(\mathcal{P}^{(V)}) \subseteq \mathcal{P}^{(W)}\,.$$

Finally, we state the definition of (one-sided error) testability used by Austin and Tao [11]. For vertex set V and $c \in \mathbb{N}$, we write $\binom{V}{c} := \{V' \mid V' \subseteq V, |V'| = c\}$ to denote the set of subsets of V with exactly c elements.

Definition 22. *Let K be a finite palette with order $k \geq 0$ and $\mathcal{P}$ be a hereditary K-property. Property $\mathcal{P}$ is* testable with one-sided error *if for every $\varepsilon > 0$, there exists $N \geq 1$ and $\delta > 0$ satisfying the following. For all vertex sets V with $|V| \geq N$, if $G \in K^{(V)}$ satisfies*

$$\frac{1}{|\binom{V}{N}|}\left|\left\{W \mid W \in \binom{V}{N}, G\restriction_W \in \mathcal{P}^{(W)}\right\}\right| \geq 1 - \delta\,, \tag{4.1}$$

then there exists a $G' \in \mathcal{P}^{(V)}$ satisfying

$$\frac{1}{|\binom{V}{k}|}\left|\left\{W \mid W \in \binom{V}{k}, G\restriction_W \neq G'\restriction_W\right\}\right| \leq \varepsilon\,. \tag{4.2}$$

To see that this is a variant of testability, it is easiest to consider the contrapositive. If there is a G' satisfying (4.2), then G is not ε-far from $\mathcal{P}$, using the implicit distance measure based on the fraction of differing induced subhypergraphs with size k. If there is no such G' (i.e., G is ε-far from $\mathcal{P}$) and $\mathcal{P}$ is testable, then (4.1) must not hold. That is, there are *many* induced subhypergraphs of size N that do not have $\mathcal{P}$. The definition is for hereditary $\mathcal{P}$, and so if G has $\mathcal{P}$, then so do all induced subhypergraphs. This allows the construction of testers.

Finally, we can state one of the main results of Austin and Tao [11].

Theorem 7 (Austin and Tao [11])**.** *Let K be a finite palette and let $\mathcal{P}$ be a hereditary K-property. Then, $\mathcal{P}$ is testable with one-sided error.*

In the following subsection we will map our vocabularies, structures and properties to this setting. We will then show that hereditary properties in our setting correspond to hereditary properties (in the sense of Definition 21 above) here, and that testability in the sense of this section (Definition 22) implies testability of the original relational properties. That is, we explicitly reduce our translation (Theorem 6) to Theorem 7.

Reducing Theorem 6 to Theorem 7

To begin, we map vocabulary $\tau = \{R_1^{a_1}, \dots, R_s^{a_s}\}$ to finite palette $K_\tau = (K_i)_{i=0}^{\infty}$. We use the color of a "tuple" to represent the set of assignments on it. The difference between the set of j-ary tuples over a finite universe U and $\mathrm{Inj}([j], U)$ is that the latter does not permit repeated components. If $S \in SUB(R_i^{a_i})$ is such that $|S| < a_i$, the corresponding subrelation consists of tuples with repeated components. We treat such S as relations with arity $|S|$ and no repeated components.

Recall that $\mathfrak{S}(n, k)$ is the Stirling number of the second kind.

For $a \geq 1$, let $P_a := \{R_i^{a_i} \mid R_i^{a_i} \in \tau, a_i = a\}$ be the set of predicate symbols with arity a. We now define palette K. Let $K_0 := \mathrm{pt}$ and $K_i := \left[2^{\sum_j^i |P_j| \mathfrak{S}(j,i)}\right]$. There are finitely-many predicate symbols and so only finitely-many $K_i \neq \mathrm{pt}$.

Let $S_a := \{S_a^i \mid S_a^i \in SUB(R_i^{a_i}), |S_a^i| = a, 1 \leq i \leq s\}$ be the set of subtypes with cardinality a for all $a \geq 1$. Now, $2^{|S_a|} = |K_a|$ and we have exactly enough colors to encode the set of assignments of the a-ary subtypes on a-ary tuples.

We will now define a map h from relational structures A on universe U to hypergraphs $G_A \in K^{(U)}$. For any $S_a^i \in S_a$, there is a bijection

$$r(S_a^i)\colon s^U(S_a^i) \to \{(x_1, \ldots, x_a) \mid x_i \in U, x_i \neq x_j \text{ for } i \neq j\}$$

from $s^U(S_a^i)$ to the a-ary tuples without duplicate components, formed by removing the duplicate components. That is, $r(S_a^i)$ maps $(x_1, \ldots, x_{a_i})$ to $(x_{i_1}, \ldots, x_{i_a})$ where $1 \leq i_1 < i_2 < \ldots < i_a \leq a_i$. We can now define $G_A = h(A)$.

For $j > 0$, we define $G_j\colon \mathrm{Inj}([j], U) \to K_j$ as follows. Assign to $f \in \mathrm{Inj}([j], U)$ the color encoding the set of assignments of the subtypes S_j on $(f(1), \ldots, f(j))$, using the inverses $(r(S_j^i))^{-1}$ to get assignments for subtypes of high-arity relations. For $j = 0$, $\mathrm{Inj}([j], U) = \emptyset$ and $K_0 = \mathrm{pt}$ and we can use a trivial map.

Of course, we extend the map to properties in the obvious way. If P is a property of relational structures, we let $\mathcal{P}^{(U)} := \{h(A) \mid A \in P\}$. Formally, we define $\mathcal{P}(U) := \mathcal{P}^{(U)}$, but there is a small technical point. We have identified finite universes with subsets of the naturals, allowing us to call $STRUC(\tau)$ a *set*. However, Definition 18 in this section allows a vertex set to be *any* finite set and Definition 21 requires hereditary hypergraph properties to be closed under bijections between vertex sets. To remedy this, for each finite vertex set W, we fix a[6] bijection $g^W \colon W \to \{0, \dots, |W| - 1\}$. We then define $\mathcal{P} := h(P)$ formally as

$$\mathcal{P}(W) := \begin{cases} \mathcal{P}^{(W)}, & \text{if } W = \{0, 1, \dots, |W| - 1\}; \\ K^{\left(g^W\right)}\left(\mathcal{P}^{(\{0,\dots,|W|-1\})}\right), & \text{otherwise.} \end{cases}$$

Hereditary relational properties are mapped to hereditary hypergraph properties, which are testable in the sense of this section (Definition 22 above) by Theorem 7.

Lemma 12. *If P is a hereditary property of relational structures, then $h(P)$ is a hereditary property of hypergraphs.*

Proof (Lemma 12). Let P be a hereditary property of relational structures with vocabulary τ. Assume that $\mathcal{P} := h(P)$ is not a hereditary K-property. Then, by Definition 21 above, there exist finite vertex sets V and W, and a morphism $f' \in \mathrm{Inj}(W, V)$ such that

$$K^{(f)}(\mathcal{P}^{(V)}) \not\subseteq \mathcal{P}^{(W)} . \tag{4.3}$$

Since f' exists, $\mathrm{Inj}(W, V)$ cannot be the empty set and so $|V| \geq |W|$. Let $U_V := \{0, \dots, |V| - 1\}$ and $U_W := \{0, \dots, |W| - 1\}$. By the definition of $\mathcal{P}$, we can fix bijections $g^V \colon V \to U_V$ and $g^W \colon W \to U_W$ such that $\mathcal{P}^{(V)} = K^{\left(g^V\right)}\left(\mathcal{P}^{(U_V)}\right)$ and $\mathcal{P}^{(W)} = K^{\left(g^W\right)}\left(\mathcal{P}^{(U_W)}\right)$. By the definition of $\mathcal{P} = h(P)$, this implies

$$K^{(f)}\left(K^{\left(g^V\right)}\left(\mathcal{P}^{(U_V)}\right)\right) \not\subseteq K^{\left(g^W\right)}\left(\mathcal{P}^{(U_W)}\right) .$$

[6] Our properties are closed under isomorphisms, so any fixed bijection is acceptable.

Bijections are invertible, and so this implies

$$K^{\left(g^V\circ f\circ\left(g^W\right)^{-1}\right)}\left(\mathcal{P}^{(U_V)}\right)\not\subseteq\mathcal{P}^{(U_W)}.$$

Rename $f' := g^V \circ f \circ \left(g^W\right)^{-1}$ and note $f' \in \operatorname{Inj}(U_W, U_V)$. Let $A' \in \mathcal{P}^{(U_V)}$ be such that $K^{(f')}(A') \notin \mathcal{P}^{(U_W)}$.

We defined $\mathcal{P}$ as $h(P)$ for a hereditary property P of relational structures. Property P is closed under isomorphisms, and so there is an

$$A := h^{-1}(A') \in P \cap STRUC^{|U_V|}(\tau)$$

such that the $|U_W|$-element structure induced by $\{a \mid a = f'(u) \text{ for some } u \in U_w\}$ does not have P. This contradicts the hereditariness of P and so $\mathcal{P}$ must be hereditary in the sense of this section (Definition 21).

□ Lemma 12

We mapped hereditary relational properties to hereditary hypergraph properties, which are testable by Theorem 7. We will show this implies testability of the original properties.

Definition 23. *Let $A, B \in STRUC^n(\tau)$ be structures with vocabulary τ and universe $U := \{0, \dots, n-1\}$ of size n, $k := \max_i a_i$ be the maximum arity of the predicate symbols, and $h\colon STRUC^n(\tau) \to K^{(U)}$ be the map defined above. The* h-distance *between A and B is*

$$\operatorname{hdist}(A,B) := \frac{1}{\left|\binom{U}{k}\right|}\left|\left\{W \mid W \in \binom{U}{k}, h(A)\restriction_W \neq h(B)\restriction_W\right\}\right|.$$

We now relate the two distances with the following simple lemma.

Lemma 13. *Let $A, B \in STRUC^n(\tau)$ be relational structures with vocabulary τ and size n. Then,* $\operatorname{hdist}(A, B) \geq \operatorname{mrdist}(A, B)$.

Proof (Lemma 13). Assume that $\operatorname{mrdist}(A, B) = \varepsilon$. Then, there exists a predicate symbol $R_i^{a_i} \in \tau$ and subtype $S \in SUB(R_i^{a_i})$ such that

$$\left|s^A(S) \triangle s^B(S)\right| / (n!/(n - |S|)!) = \varepsilon\,.$$

Let $k := \max_i a_i$ and let the universe of both structures be $U_n := \{0, \ldots, n - 1\}$.

Consider a random permutation of the universe (i.e., a bijection $r\colon U_n \to U_n$) chosen uniformly from the set of such permutations. The probability that the substructures induced on $\{r(0), \ldots, r(k - 1)\}$ differ in A and B is $\operatorname{hdist}(A, B)$. The probability that the tuple of the first $|S|$ elements, i.e. $(r(0), \ldots, r(|S| - 1))$, differ in $s^A(S)$ and $s^B(S)$ is ε and so $\operatorname{hdist}(A, B) \geq \varepsilon$. □ Lemma 13

Equality is obtained when $|S| = k$. It is possible to show that the two distances differ by at most a constant factor, and so the corresponding notions of testability are essentially equivalent. However, Lemma 13 suffices for our purposes.

Lemma 14. *Let $P \subseteq STRUC(\tau)$ be a property of relational structures which is mapped by h to a property of hypergraphs that is testable with one-sided error. Then, P is testable with one-sided error.*

Proof (Lemma 14). Let $\mathcal{P} := h(P)$ be the hypergraph property which P maps to. We show that the following is an ε-tester for P with one-sided error. Let $N \geq 1$,

$\delta > 0$ be the constants of Definition 22 above for ε. Assume that we are testing a structure $A \in STRUC^n(\tau)$ and recall that $U = \{0, \ldots, n-1\}$.

1. If $\#(A) \leq N$, query the entire structure and decide exactly whether $A \in P$.
2. Otherwise, repeat the following $q(\delta)$ times.
   (a) Uniformly select N elements and query the induced substructure.
   (b) If it has P, continue. Otherwise, reject.
3. Accept if all of the induced substructures had P.

If $A \in P$, then all induced substructures have P because P is hereditary and the tester accepts with probability 1. Next, assume $\mathrm{mrdist}(A, P) > \varepsilon$. We use Definition 22 above to show the tester will find a witness for $A \notin P$ with probability at least 2/3. By Lemma 13, $\mathrm{hdist}(A, P) \geq \mathrm{mrdist}(A, P) > \varepsilon$. We assumed $h(P)$ is hereditary, and so (by Theorem 7) it is testable in the sense of Definition 22. The probability that a uniformly chosen N-element substructure does not have P is at least δ. We use $q(\delta)$ to amplify the success probability from δ to 2/3.

$\square$ Lemma 14

This completes the proof of the testability of Ramsey's class (Theorem 5). All properties expressible in Ramsey's class are indistinguishable from instances of F-colorability. Indistinguishability preserves testability and so it sufficed to show that these instances are testable. All instances of F-colorability are hereditary relational properties, which are testable by Theorem 6, which we reduced to the statement by Austin and Tao [11].

4.3 Summary

We proved the testability of two classes in this chapter. In Section 4.1, we used model-theoretic techniques to prove testability for Ackermann's class with equality and explicitly constructed testers for each formula in the class. This result first appeared in [38] with two-sided error, and the improved (i.e., one-sided) version proven here will appear in [42].

In Section 4.2, we used a reduction to a strong result by Austin and Tao [11] to prove testability for Ramsey's class. This extends a result of Alon *et al.* [3] from loop-free, undirected graphs to relational structures, and answers a question of Fischer [20] (who asked whether the hypergraph properties expressible with prefix $\exists^*\forall^*$ are testable). This result first appeared in [39], and will also appear in [42].

It is interesting to consider the query complexities for these two classes. In particular, our construction for Ramsey's class gives a tremendous dependence on ε (towers of $1/\varepsilon$). Improving the query complexity for Ramsey's class is difficult, and is closely linked to open problems in extremal (hyper)graph theory (see the Conclusion in Chapter 6).

Chapter 5

Untestable Classes

In the previous chapter, we saw that two large syntactic classes of first-order logic are testable. However, there are also first-order properties that are not testable. In this chapter, we prove that several classes are untestable.

5.1 Untestable Properties in $[\forall^3\exists, (0,1)]_=$

As mentioned in Subsection 1.1.1, Alon *et al.* [3] proved that there exists an untestable property of undirected, loop-free graphs expressible with quantifier prefix $\forall^{12}\exists^5$. In this section, we simplify their untestable example and thereby show that the classes $[\forall^3\exists, (0,1)]_=$, $[\forall^2\exists\forall, (0,1)]_=$, $[\forall\exists\forall^2, (0,1)]_=$ and $[\forall\exists\forall\exists, (0,1)]_=$ are untestable. The focus on graphs is justified by recalling that monadic first-order logic is testable.

We will improve on three of these prefixes in Section 5.2, where we prove that there are untestable properties in $[\forall\exists\forall, (0,1)]_=$. However, we still need Theorem 8 for the prefix $[\forall^3\exists, (0,1)]_=$ and we present it first due to its relative simplicity.

The class $[\forall^3\exists, (0,1)]$ (usually without equality) is well-known in the literature. It is trivial to prove that this class does not have the finite model property. In addition, Kolaitis and Vardi [45] showed that a 0-1 law does not hold for second-order existential logic when the first order part is in this class (even without equality). However, it is an essentially finite class (i.e., it can only express a finite number of properties up to logical equivalence) and therefore decidable.

We will begin by defining property P, which is essentially the graph isomorphism problem for undirected loop-free graphs encoded in directed graphs that may contain loops. We will begin by showing in Lemma 15 that P is indistinguishable from property P_f (cf. Definition 25 below) which is expressible in any of the prefix vocabulary classes mentioned in Theorem 8 below. We will then show that P is not testable. Indistinguishability preserves testability and so this implies that P_f is also untestable, which will suffice to show the following theorem.

Theorem 8. *The following prefix classes are not testable:*

1. $[\forall\exists\forall\exists, (0,1)]_=$
2. $[\forall\exists\forall^2, (0,1)]_=$
3. $[\forall^2\exists\forall, (0,1)]_=$
4. $[\forall^3\exists, (0,1)]_=$

We define property P as follows. First, a graph that has property P must consist of an even number of vertices, of which exactly half have loops. The subgraph induced by the vertices with loops must be isomorphic to that induced by the vertices without loops, ignoring all loops, and there must be no edges connecting the vertices with loops to those without loops. Finally, all edges must be *undirected* (i.e., an edge from x to y implies an edge from y to x). We refer to such undirected edges as *paired edges.*

Definition 24. *A graph $G \in \mathcal{G}^n$ has P iff the following conditions are satisfied:*

1. *For some s, $n = 2s$.*
2. *There are exactly s vertices x satisfying $E(x, x)$. We will refer to the set of such vertices as H_1 and to the remaining s vertices as H_2.*
3. *The substructure induced by H_1 is isomorphic to that induced by H_2 when all loops are removed. That is, there is a bijection f from H_1 to H_2 such that for distinct $x, y \in H_1$, it is true that $G \models E(x, y)$ iff $G \models E(f(x), f(y))$.*
4. *There are no edges between H_1 and H_2.*
5. *All edges are paired.*

Graph isomorphism is not directly expressible in first-order logic, and so we use the following encoding where the bijection f is made explicit by adding n edges between H_1 and H_2. This of course reduces the complexity from the level of finding an isomorphism to the level of checking a given one, in order to achieve first-order expressivity. However, it maintains hardness for testability: essentially, our samples are too small to see any part of the given isomorphism.

Definition 25. *A graph $G \in \mathcal{G}^n$ has P_f iff the following conditions are satisfied:*

1. *For every vertex x, if $E(x, x)$ then there is an edge from x to exactly one y such that $\neg E(y, y)$.*

2. *For every vertex x, if $\neg E(x,x)$ then there is an edge from x to exactly one y such that $E(y,y)$.*

3. *For all vertices x and y, $E(x,y)$ iff $E(y,x)$.*

4. *For all pairwise-distinct vertices x_1, x_2, x_3, x_4, if $E(x_1,x_1)$, $\neg E(x_2,x_2)$, $E(x_3,x_3)$, $\neg E(x_4,x_4)$, $E(x_1,x_2)$ and $E(x_3,x_4)$, then $E(x_1,x_3)$ iff $E(x_2,x_4)$.*

Expressing Conditions 1 and 2 as "there is at most one such y" and "there is at least one such y," P_f can be expressed in each of the classes $[\forall\exists\forall\exists, (0,1)]_=$, $[\forall\exists\forall^2, (0,1)]_=$, $[\forall^2\exists\forall, (0,1)]_=$ and $[\forall^3\exists, (0,1)]_=$.

For example, in the class $[\forall^3\exists, (0,1)]_=$, we can express P_f by

$$\begin{aligned}
&\forall x_1 \forall x_3 \forall x_4 \exists x_2 : \Big[\\
&\quad \big((E(x_1,x_1) \leftrightarrow \neg E(x_2,x_2)) \wedge E(x_1,x_2) \big) \quad \wedge \\
&\quad \big[\big((E(x_1,x_1) \leftrightarrow \neg E(x_3,x_3)) \wedge (E(x_3,x_3) \leftrightarrow E(x_4,x_4)) \wedge \\
&\qquad E(x_1,x_3) \wedge E(x_1,x_4) \big) \rightarrow x_3 = x_4 \big] \quad \wedge \\
&\quad \big(E(x_1,x_3) \rightarrow E(x_3,x_1) \big) \quad \wedge \\
&\quad \big([E(x_1,x_1) \wedge E(x_3,x_3) \wedge x_1 \neq x_3 \wedge \neg E(x_4,x_4) \wedge E(x_3,x_4)] \rightarrow \\
&\qquad (\neg E(x_2,x_2) \wedge E(x_1,x_2) \wedge (E(x_1,x_3) \leftrightarrow E(x_2,x_4))) \big) \Big] .
\end{aligned}$$

To express P_f with prefixes $\forall^2\exists\forall$ and $\forall\exists\forall^2$, it suffices to reorder the quantifiers (keeping x_2 existential and x_1 first). The prefix $\forall\exists\forall\exists$ requires a few additional modifications.

The two properties P and P_f differ only in the edges which make the isomorphism explicit in P_f but are forbidden in P. There are at most n such edges, none of which are loops. This suffices to prove the following.

Lemma 15. *Properties P and P_f are indistinguishable.*

Proof (Lemma 15). Let $\varepsilon > 0$ be arbitrary and let $N_\varepsilon = \varepsilon^{-1}$. Assume that G is a structure that has property P and that $\#(G) > N_\varepsilon$. We will show that $\mathrm{mrdist}(G, P_f) < \varepsilon$.

Structure G has P and so there is a bijection f satisfying Condition 3 of Definition 24. For all $x \in H_1$, we add the edges $E(x, f(x))$ and $E(f(x), x)$ and call the result G'. Property P_f differs from P only in that the isomorphism is made explicit by the edges connecting loops and non-loops, and so G' has P_f. Indeed, it satisfies Conditions 1 and 2 of Definition 25 because G had no edges between loops and non-loops and we have connected each to exactly one of the other, following the bijection f. Next, G' satisfies Condition 3 of Definition 25 because G satisfied Condition 5 of Definition 24 and we added only paired edges. Finally, G' satisfies Condition 4 of Definition 25 because the edges between loops and non-loops follow the isomorphism f from Condition 3 of Definition 24.

We have added exactly n (directed) edges, none of which are loops and so we have $\mathrm{mrdist}(G, P) \leq \mathrm{mrdist}(G, G') = 0 + n/n^2 < \varepsilon$, where the inequality holds for $n > N_\varepsilon$. The converse is analogous; given a G that has property P_f, we simply remove the n edges between loops and non-loops after using them to construct the isomorphism f. □ Lemma 15

Properties P and P_f are indistinguishable. We saw in Section 3.3 that testability is preserved by indistinguishability (cf. Theorem 7) and thus showing that P is not testable suffices to prove that P_f is not testable (and therefore Theorem 8).

The proof closely follows that of Alon *et al.* [3]. The crucial lemma is the following, a combination of Lemmata 7.3 and 7.4 from Alon *et al.* [3]. We use $\mathrm{count}_H(T)$ to refer to the number of times that a graph T occurs as an induced subgraph in H. A *bipartite graph* is a graph where we can partition the vertices into two sets U_1 and U_2 such that there are no edges "internal" to the partitions. That is, for all $x_1, y_1 \in U_1$ and $x_2, y_2 \in U_2$, $\neg E(x_1, y_1)$ and $\neg E(x_2, y_2)$.

Lemma 16 (Alon *et al.* [3]). *There exists a constant $\varepsilon' > 0$ such that for every $D \in \mathbb{N}$, there exist two undirected bipartite graphs $H = H(D)$ and $H' = H'(D)$, and a number t satisfying the following conditions.*

1. *Both H and H' have a bipartition into classes U_1 and U_2, each of size t.*
2. *In both H and H', for all subgraphs X with size $t/3 \leq \#(X) \leq t$, there are more than $t^2/18$ undirected edges between X and the remaining part of the graph.*
3. *The minimum degree of both H and H' is at least $t/3$.*
4. $\mathrm{dist}(H, H') \geq \varepsilon'$.
5. *For all D-element graphs T, $\mathrm{count}_H(T) = \mathrm{count}_{H'}(T)$.*

Lemma 16 is due to Alon *et al.* [3], although they only sketch the proof. We include a complete proof here for completeness.

5.1.1 Proof of Lemma 16

We follow the proof outlined by Alon *et al.* [3] and begin with their proof of the following simple lemma.

Lemma 17 (Alon *et al.* [3]). *There exist constants ε and N such that every graph H with $n > N$ (labelled) vertices is ε-far from all but at most $2^{n^2/5}$ other graphs with the same vertex set.*

Proof (Lemma 17). Choose an $\varepsilon < 1/2$ such that $\left(\frac{e}{\varepsilon}\right)^{\varepsilon} < 2^{1/10}$ and an $N > \varepsilon^{-1}$ such that $n^n < 2^{n^2/10}$ for $n > N$. The number of graphs that are less than ε-far from a given H with $n > N$ vertices is at most

$$n! \sum_{i=0}^{\varepsilon n^2} \binom{\binom{n}{2}}{i} . \tag{5.1}$$

We apply the identity $\binom{a}{b} = \binom{a-1}{b-1} + \binom{a-1}{b}$ inductively to $\binom{n^2}{\varepsilon n^2}$. A simple inductive proof shows that applying this identity once to each term, repeating for a levels, gives $\binom{n^2}{\varepsilon n^2} = \sum_{i=0}^{a} \binom{a}{i}\binom{n^2-a}{\varepsilon n^2 - i} =: L(a)$, and so

$$\binom{n^2}{\varepsilon n^2} = L(n) = \sum_{i=0}^{n} \binom{n}{i}\binom{n^2-n}{\varepsilon n^2 - i} . \tag{5.2}$$

Recalling that $\sum_{i=0}^{n}\binom{n}{i} = 2^n$, there are $2^n > \varepsilon n^2$ total "terms" in the summation. Each term in the summation $\sum_{i=0}^{\varepsilon n^2} \binom{\binom{n}{2}}{i}$ can therefore be paired with a term from (5.2) that upper-bounds it. Combining with (5.1), we see that the number of graphs that are less than ε-far from H is less than

$$n! \binom{n^2}{\varepsilon n^2} < n^n \left(\frac{e}{\varepsilon}\right)^{\varepsilon n^2} < 2^{n^2/5} ,$$

as desired. □ Lemma 17

Next, we show that most (sufficiently large) bipartite graphs satisfy Conditions 2 and 3 of Lemma 16. We use the following statement of Chernoff bounds (see

Appendix A of Alon and Spencer [10]);

$$\Pr[X < a] \leq \mathrm{E}[e^{-\lambda X}]e^{\lambda a}\,, \tag{5.3}$$

where λ is chosen to optimize the bound, and also the following lemma.

Lemma 18 (Lemma A.1.5 in Alon and Spencer [10])**.**

$$\frac{e^{\lambda} + e^{-\lambda}}{2} = \cosh(\lambda) \leq e^{\lambda^2/2}\,. \tag{5.4}$$

Lemma 19 (Alon *et al.* [3])**.** *There exists an N' such that for $n > N'$, at least $\frac{1}{2}2^{n^2}$ of the bipartite graphs with a given (labeled) bipartition U_1, U_2 where $|U_1| = |U_2| = n$ satisfy both of the following conditions.*

(191) The minimum degree is at least $n/3$.

(192) For every subset X of $U_1 \cup U_2$ with size $n/3 \leq |X| \leq n$, there are more than $n^2/18$ edges between X and $(U_1 \cup U_2)\backslash X$.

Proof (Lemma 19). We let G be a random bipartite graph with a given, labeled bipartition U_1, U_2 chosen in the following way. Each possible edge $(u, v) \in U_1 \times U_2$ is placed independently and uniformly with probability 1/2. There are n^2 possible edges, and so each of the 2^{n^2} possible such bipartite graphs is generated with equal probability. The probability of G satisfying (19.1) and (19.2) above is (by definition)

$$\frac{|\{H \mid H \text{ is such a graph that satisfies (19.1) and (19.2)}\}|}{2^{n^2}}\,.$$

We want a lower-bound on the number of such graphs, or equivalently a lower-bound on $\Pr[G \text{ satisfies (19.1) and (19.2)}] \cdot 2^{n^2}$ that is greater than $\frac{1}{2}2^{n^2}$. It suffices therefore to show that this probability is at least 1/2. Using the union bound, $\Pr[G \text{ satisfies (19.1) and (19.2)}] \geq$

$$1 - \Pr[G \text{ does not satisfy (19.1)}] - \Pr[G \text{ does not satisfy (19.2)}]\,.$$

We will show in Claims 1 and 2 that this is at least $1 - o(1) > 1/2$, where the inequality holds for sufficiently large N'. We will let $U = U_1 \cup U_2$ be the set of all vertices.

Claim 1. $\Pr[G$ *does not satisfy (19.1)*$] = o(1)$.

Proof (Claim 1). Let $\deg(v)$ be the degree of a vertex v. By the union bound,

$$\Pr[G \text{ does not satisfy (19.1)}] \leq \sum_{v \in U} \Pr[\deg(v) \leq n/3].$$

Let N_{uv} be an "indicator" variable for the event that there is an edge $E(u,v)$ which is normalized to take the following values,

$$N_{uv} := \begin{cases} -1, & \text{if } \neg E(u,v); \\ 1, & \text{if } E(u,v). \end{cases} \tag{5.5}$$

Then, $\deg(v) \leq n/3$ iff

$$Y_v := \sum_{\{u \mid u \in U_1 \text{ if } v \in U_2,\ u \in U_2 \text{ if } v \in U_1\}} N_{uv} \leq -n/3\,,$$

and so $\sum_{v \in U} \Pr[\deg(v) \leq n/3] = \sum_{v \in U} \Pr[Y_v \leq -n/3]$. We apply (5.3) and Lemma 18, and so

$$\Pr[Y_v \leq -n/3] \leq \mathrm{E}[e^{-\lambda Y_v}]e^{-\lambda n/3} = \cosh(\lambda)e^{-\lambda n/3} \leq e^{\lambda^2 n/2 - \lambda n/3}.$$

Minimizing the bound by setting $\lambda = 1/3$ gives

$$\Pr[G \text{ does not satisfy (19.1)}] \leq (2n)e^{-n/18} = o(1),$$

as desired. □ Claim 1

Claim 2. $\Pr[G$ *does not satisfy (19.2)*$] = o(1)$.

Proof (Claim 2). By the union bound,

$$\Pr[G \text{ does not satisfy (19.2)}] \leq \sum_{\{X | X \subseteq U, n/3 \leq |X| \leq n\}} \Pr[G \text{ violates (19.2) with } X]. \tag{5.6}$$

Let $a := |X \cap U_1|$, $b := |X \cap U_2|$, $i := a + b = |X|$. As in (5.5), we let N_{uv} be a normalized indicator for the event $E(u,v)$. Let $Y_X := \sum_{\{u,v | u \in X, v \in U \setminus X\}} N_{uv}$. Then, G violates (19.2) with X iff $Y_X < -i(2n-i) + n^2/9$. Using again (5.3),

$$\begin{aligned}
&\sum_{\{X | X \subseteq U, n/3 \leq |X| \leq n\}} \Pr[G \text{ violates (19.2) with } X] &=\\
&\sum_{\{X | X \subseteq U, n/3 \leq |X| \leq n\}} \Pr[Y_X < -i(2n-i) + n^2/9] &\leq\\
&\sum_{\{X | X \subseteq U, n/3 \leq |X| \leq n\}} \mathrm{E}[e^{-\lambda Y_X}]e^{\lambda(-i(2n-i)+n^2/9)} &=\\
&\sum_{\{X | X \subseteq U, n/3 \leq |X| \leq n\}} \prod_{\{u,v | u \in X, v \in U \setminus X\}} \mathrm{E}[e^{-\lambda N_{uv}}]e^{\lambda(-i(2n-i)+n^2/9)}. &
\end{aligned} \tag{5.7}$$

We can divide the product into four cases, $\prod_{\{u,v|u\in X,v\in U\setminus X\}} \mathrm{E}[e^{-\lambda N_{uv}}] =$

$$\left(\prod_{\{u,v|u\in X\cap U_1,v\in U_1\setminus X\}} e^{\lambda}\right)\left(\prod_{\{u,v|u\in X\cap U_2,v\in U_2\setminus X\}} e^{\lambda}\right)\cdot$$
$$\left(\prod_{\{u,v|u\in X\cap U_1,v\in U_2\setminus X\}} \frac{e^{\lambda}+e^{-\lambda}}{2}\right)\left(\prod_{\{u,v|u\in X\cap U_2,v\in U_1\setminus X\}} \frac{e^{\lambda}+e^{-\lambda}}{2}\right).$$

Recalling Lemma 18 and combining with (5.7), $\Pr[G$ does not satisfy (19.2)$] \leq$

$$\sum_{\{X\subseteq U|n/3\leq|X|\leq n\}} e^{\lambda(a(n-a)+b(n-b))+\frac{\lambda^2}{2}(a(n-b)+b(n-a))+\lambda\left(-i(2n-i)+\frac{n^2}{9}\right)}. \tag{5.8}$$

There are at most $\binom{n}{a}\binom{n}{i-a}$ choices of X with size i when $a = |X\cap U_1|$ and $b = i-a = |X\cap U_2|$, and so after simplifying, (5.8) is at most

$$\sum_{i=\lceil n/3\rceil}^{n}\sum_{a=0}^{i}\binom{n}{a}\binom{n}{i-a}e^{\lambda\left(2ai+n^2/9-2a^2-in\right)+\frac{\lambda^2}{2}\left(2a^2+in-2ai\right)}. \tag{5.9}$$

Using the simple bound $\binom{n}{k}\leq\left(\frac{en}{k}\right)^k$, we get that (5.9) is at most

$$\sum_{i=\lceil n/3\rceil}^{n}\sum_{a=0}^{i}e^{i+a\ln(n/a)+(i-a)\ln(n/(i-a))+\lambda\left(2ai+n^2/9-2a^2-in\right)+\frac{\lambda^2}{2}\left(2a^2+in-2ai\right)}. \tag{5.10}$$

Let us consider $2ai+n^2/9-2a^2-in$. If $a\geq 5n/6$, then

$$\begin{aligned}2ai+\frac{n^2}{9}-2a^2-in &\leq 2in+\frac{n^2}{9}-\frac{25}{18}n^2-in\\ &\leq -\frac{5}{18}n^2 = -\Theta(n^2).\end{aligned}$$

If $a < 5n/6$, then the maximum of $2ai+n^2/9-2a^2-in$ occurs at $a = i/2$.

Therefore,

$$2ai+\frac{n^2}{9}-2a^2-in \le i^2+\frac{n^2}{9}-\frac{i^2}{2}-in=\frac{i^2}{2}+\frac{n^2}{9}-in$$
$$\le \frac{n^2}{18}+\frac{n^2}{9}-\frac{n^2}{3}=-\frac{n^2}{6}=-\Theta(n^2)\,,$$

because the maximum occurs at the boundary $i=n/3$. Applying these bounds, (5.10) is at most

$$\sum_{i=\lceil n/3\rceil}^{n}\sum_{a=0}^{i} e^{i+a\ln(n/a)+(i-a)\ln(n/(i-a))+\lambda\left(-n^2/6\right)+\frac{\lambda^2}{2}\left(2a^2+in-2ai\right)}\,. \tag{5.11}$$

Choosing the non-optimal $\lambda=1/\sqrt{n}$ and looking only at asymptotics, we see from (5.11) that Pr[G does not satisfy (19.2)] $\le$

$$\sum_{i=\lceil n/3\rceil}^{n}\sum_{a=0}^{i} e^{O(n)+O(n\ln n)-\Theta\left(n^{3/2}\right)+O(n)}=O(n^2)e^{-\Theta\left(n^{3/2}\right)}=o(1)\,.$$

$\square$ Claim 2

The two claims combine to give the lemma. $\square$ Lemma 19

We are now ready to complete the proof of Lemma 16. We let ε' be the ε of Lemma 17, and choose a sufficiently large $s=2n>\max(N,2N')$ where N and N' are from Lemmata 17 and 19 respectively. There are at most $E:=2^{\binom{D}{2}}$ graphs on D vertices and each appears at most s^D times as an induced subgraph in a graph on s vertices. An "appearance count" for a graph is an $2^{\binom{D}{2}}$-tuple giving, for each of the possible graphs on D vertices, the number of appearances as an induced subgraph. There are therefore at most $(s^D)^E=2^{DE\log s}$ many distinct appearance counts (tuples). By Lemma 19, there are at least $\frac{1}{2}2^{n^2}$ bipartite graphs satisfying the conditions of that lemma, and

$$\frac{1}{2}2^{n^2} = 2^{s^2/4-1} = 2^{s^2/20-1}2^{s^2/5} > 2^{DE\log s}2^{s^2/5},$$

where the inequality holds for sufficiently large s.

There are at most $2^{DE\log s}$ distinct appearance counts and so there must be some appearance count shared by *more* than $2^{s^2/5}$ of the above graphs. By Lemma 17, there must be two such graphs (satisfying the conditions of Lemma 19 and with the same appearance count) that are ε-far from each other. This completes the proof of Lemma 16.

5.1.2 Completing the Proof of Theorem 8

It is worth noting that the above is for undirected, loop-free graphs. However, bipartite graphs never have loops and "undirected" in our setting results in paired edges. It is easy to show that if two structures agree on the counts for all size D induced subgraphs, they agree on the counts for all induced subgraphs of size at most D. This is done by applying the following lemma inductively.

Lemma 20. *Let H and H' be two graphs, both of size s, and let $2 < D \leq s$. If for every graph T of size D, $\text{count}_H(T) = \text{count}_{H'}(T)$, then for every graph T' of size $D-1$, $\text{count}_H(T') = \text{count}_{H'}(T')$.*

Proof (Lemma 20). Assume H and H' satisfy the initial conditions of Lemma 20, but that there exists a T' of size $D-1$ such that $\text{count}_H(T') \neq \text{count}_{H'}(T')$. Let $C = \{T \mid \#(T) = D \text{ and } T \text{ contains } T' \text{ as an induced subgraph}\}$.

Note that $\sum_{T\in C}\operatorname{count}_H(T)\operatorname{count}_T(T') = \operatorname{count}_H(T')(s-D+1)$ and likewise for $\sum_{T\in C}\operatorname{count}_{H'}(T)\operatorname{count}_T(T')$. We assumed that H and H' satisfy $\operatorname{count}_H(T) = \operatorname{count}_{H'}(T)$ for $T\in C$, but $\operatorname{count}_H(T')\neq\operatorname{count}_{H'}(T')$, giving a contradiction and the Lemma follows. $\square$ Lemma 20

Lemma 21. *Property P is not testable.*

Proof (Lemma 21). Assume that P is testable. Then, there exists an ε-tester for

$$\varepsilon := \min\{\varepsilon'/8, 1/144\},$$

where ε' is the constant from Lemma 16 above. We can assume without loss of generality that the tester queries all edges in a random sample of $D := D(\varepsilon)$ vertices.

Consider the graph G which contains two copies of $H = H(D)$ from Lemma 16, where one of the copies is marked by loops on each vertex and there are no edges between the copies. This graph has property P, and so the tester must accept it with probability at least $2/3$. Next, consider the graph G' which contains one copy of H marked by loops and one copy of H', again where there are no edges between the two (induced) subgraphs. Graph G' is such that $\operatorname{dist}(G', P)\geq\varepsilon$ (cf. Lemma 22 above) and so it must be rejected with probability at least $2/3$. Both G and G' consist of two bipartite graphs, each of which has a bipartition into two classes of size t, and so $\#(G) = \#(G') = 4t$.

However, G and G' both contain exactly the same number of each induced subgraph with D vertices. This is because both have loops on exactly half of the vertices and the two halves are not connected by any edges. Some of the D

vertices must be in the first copy of H and the others in the second H (resp. H'). By Lemma 20 above, H and H' contain the same number of each induced subgraph with size at most D. The tester therefore obtains any fixed sample with the same probability in G and G' and is unable to distinguish between them. Hence, it is unable to accept G with probability 2/3 and also reject G' with probability 2/3. This completes the proof, taking into account Lemma 22 below. □ Lemma 21

Recall that testing is easiest under the dist definition, and so Lemma 21 also implies P is not testable under other definitions.

Lemma 22. *The graph G' is such that* $\mathrm{dist}(G', P) \geq \varepsilon$.

Proof (Lemma 22). Suppose that $\mathrm{dist}(G', P) < \varepsilon$. Then, there is an $M \in P$ such that $\mathrm{dist}(G', M) < \varepsilon$. Let M_1 be the set of vertices with loops in M and let M_2 be the set of vertices without loops. We will refer to the subgraph induced by the vertices with loops in G' as H and to that induced by those without loops as H'. Without loss of generality, assume that $|M_1 \cap H| \geq |M_1 \cap H'|$. Then, $|M_1 \cap H| \geq t$. We let α_1 be the set $M_1 \backslash H$ and α_2 be $M_2 \backslash H'$. Note that $|\alpha_1| = |\alpha_2|$ and $|\alpha_1| \leq t$ because $|M_1 \cap H| \geq t$.

Informally, M is formed by moving vertices α_1 from H' to H and vertices α_2 from H to H', and then possibly making other changes. There are three cases, which we will consider in order.

1. $|\alpha_1| = 0$.

2. $|\alpha_1| \geq t/3$.

3. $0 < |\alpha_1| < t/3$.

If $|\alpha_1| = 0$, then we can construct M from G' without exchanging vertices between H and H', and in particular, construct H' from H (ignoring loops), by making less than $\varepsilon(4t)^2$ modifications. However, $\mathrm{dist}(H, H') \geq \varepsilon'$ by Lemma 16 above and so this must require at least $\varepsilon'(2t)^2$ modifications. By definition, $\varepsilon < \varepsilon'/4$ so $\varepsilon(4t)^2 < \varepsilon'(2t)^2$. The first case is therefore not possible.

Recall that $|\alpha_1| \leq t$. If $|\alpha_1| \geq t/3$, then by Condition 2 of Lemma 16 there exists at least $t^2/18$ undirected edges between α_1 and $H'\backslash\alpha_1$ and between α_2 and $H\backslash\alpha_2$. All of these edges must be removed to satisfy P because each would connect a vertex with a loop to a vertex without a loop. Therefore,

$$\mathrm{dist}(G', M) \;\geq\; \frac{4t^2/18}{(4t)^2} \;=\; 1/72\,.$$

But, $\varepsilon < 1/72$ and so the second case is not possible.

Therefore, it must be that $0 < |\alpha_1| < t/3$. Here, we will show that it must be the case that α_1 and α_2 are relatively far apart. If they are not far apart, then it is possible to modify them instead of swapping them. This essentially results in the first case considered above. Condition 3 of Lemma 16 requires that each vertex has relatively high degree. These edges can be either internal to α_1 (resp. α_2) or connecting α_1 (α_2) with $H'\backslash\alpha_1$ ($H\backslash\alpha_2$). If α_1 and α_2 are relatively far apart, then we will see that this forces too many edges "outside" of α_1 (resp. α_2), resulting in a similar situation to the second case considered above.

We have assumed that $\mathrm{dist}(G', M) < \varepsilon$ and that we can construct M from G' by making less than $\varepsilon(4t)^2$ modifications if we move α_1 to H and α_2 to H'. This entails the following modifications.

1. Removing all edges connecting α_1 to $H'\backslash\alpha_1$.

2. Removing all edges connecting α_2 to $H\backslash\alpha_2$.

3. Adding any required edges between α_1 and $H\backslash\alpha_2$.

4. Adding any required edges between α_2 and $H'\backslash\alpha_1$.

5. Changing α_1, α_2, $H\backslash\alpha_2$ and $H'\backslash\alpha_1$ to their final forms.

We can assume that the total number of modifications is less than $\varepsilon(4t)^2$. It must be that $\text{dist}(\alpha_1,\alpha_2)|\alpha_1|^2/(4t)^2+\varepsilon \geq \varepsilon'/4$. If this does not hold, then we could first modify α_1 to make it identical to α_2 and then make H' identical to M_2. Next, M_2 is identical to M_1, which we could make identical to H. This would require less than $\varepsilon'(2t)^2$ modifications, which would violate Lemma 16. Therefore,

$$\text{dist}(\alpha_1,\alpha_2) \;\geq\; \frac{16(\varepsilon'/4-\varepsilon)t^2}{|\alpha_1|^2}\,. \tag{5.12}$$

If both α_1 and α_2 are complete graphs then they cannot be far apart. Given that all vertices in α_1 (α_2 is analogous) have degree at least $t/3$, then there must be at least

$$|\alpha_1|(t/3-|\alpha_1|+1)+2r$$

edges connecting α_1 to $H'\backslash\alpha_1$, where r is the number of edges internal to α_1 that must be omitted to satisfy (5.12). The simple lower bound on r, the number of edges needed for two graphs with at most r edges to be $\text{dist}(\alpha_1,\alpha_2)$-far, that follows from $\text{dist}(\alpha_1,\alpha_2) \leq 2r/|\alpha_1|^2$ is sufficient. Finally, combining this with Inequality (5.12) yields

$$r \;\geq\; 8(\varepsilon'/4-\varepsilon)t^2\,. \tag{5.13}$$

The number of edges connecting α_1 to $H'\backslash\alpha_1$ is therefore, by (5.13), at least

$$|\alpha_1|(t/3 - |\alpha_1| + 1) + 16(\varepsilon'/4 - \varepsilon)t^2 \;\geq\; 16(\varepsilon'/4 - \varepsilon)t^2 \,.$$

All of these edges must be removed to move α_1 (resp. α_2), and so

$$\mathrm{dist}(G', M) \;\geq\; \frac{16(\varepsilon'/4 - \varepsilon)t^2}{(4t)^2} \;=\; \frac{\varepsilon'}{4} - \varepsilon \,.$$

We have defined $\varepsilon \leq \varepsilon'/8$ and so $\mathrm{dist}(G', M) \geq \varepsilon$, a contradiction.

The cases are exhausted and so $\mathrm{dist}(G', P) \geq \varepsilon$ as desired. $\square$ Lemma 22

5.2 The Kahr-Moore-Wang Class with Equality

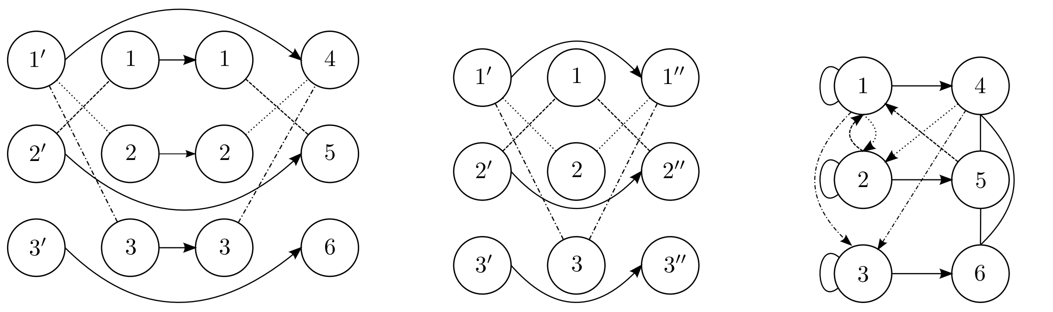
In Section 5.1, we saw an untestable first-order property that is an encoding of graph isomorphism similar to the untestable property of Alon *et al.* [3]. However, expressing graph isomorphism seems to require us to use four quantifiers, as we'd like to say that some edge is present if some other (in general, disjoint) edge is present. In this section we prove the untestability of a prefix with length three, using a variant of graph isomorphism that is closely related to Boolean function isomorphism.

The main result of this section is Theorem 9.

Theorem 9. *The prefix class* $[\forall\exists\forall, (0,1)]_=$ *is not testable.*

5.2.1 From Four to Three Quantifiers

We begin by introducing the idea that allows us to remove one quantifier from the prefix while maintaining the hardness of our property for testing. Recall that graph isomorphism is generally hard for testing (see, e.g., Fischer and Matsliah [21]). In fact, restricting the properties to checking an explicitly given isomorphism between undirected, bipartite graphs (see Figure 5.1(a)) maintains hardness for testing.



(a) A graph with P_b (b) A graph with P_e (c) A graph with P_f

Figure 5.1. Properties P_b, P_e and P_f

Sharing one of the partitions (see Figure 5.1(b)) would seem to remove the need for four quantifiers. The resulting property is perhaps closer to a variant of *function* isomorphism, e.g., for functions $f, g\colon [n] \to \{0,1\}^n$ where bit i of $f(j)$ is 1 if there is an edge from j in the leftmost partition to i in the middle partition and likewise for $g(j)$ and the right partition. This property is not first-order expressible, but there is a somewhat tedious first-order encoding that is similar (see Figure 5.1(c) and the formula in Subsection 5.2.2 below).

This connection with function isomorphism allows us to leverage recent work on the testability of (Boolean) function isomorphism and use recent ideas and techniques from Alon and Blais [2] to prove Lemma 23.

5.2.2 Proof of Theorem 9

We begin by outlining the proof. First, we define P_f, a property expressible in the class $[\forall\exists\forall,(0,1)]_=$ which, as described above, is in some sense a somewhat tedious but first-order expressible variant of checking (explicit) isomorphism of undirected bipartite graphs in tripartite graphs. We then define a variant P_2, in which the isomorphism is not explicitly given and we must test whether there exists some suitable isomorphism. Although this increases the complexity of deciding the problem from checking an isomorphism to finding one, it does not change hardness for testing. We will show that P_2 and P_f are indistinguishable and so P_2 is testable iff P_f is testable. Finally, we prove directly that P_2 is untestable, even with $o(\sqrt{n})$ queries, using an argument based on a recent proof by Alon and Blais [2].

Proof (Theorem 9). We begin by defining P_f. Formally, it is the set of graphs satisfying the following conjunction of four clauses (see Figure 5.1(c) for an example).

$$\begin{aligned}
\forall x\exists y\forall z: \quad & \{\, ((\neg E(x,x)\wedge\neg E(z,z)\wedge x\neq z)\rightarrow E(x,z))\\
\wedge \quad & (E(x,x)\rightarrow(E(x,y)\wedge\neg E(y,y)\wedge[(\neg E(z,z)\wedge E(x,z))\rightarrow y=z]))\\
\wedge \quad & (\neg E(x,x)\rightarrow(E(y,x)\wedge E(y,y)\wedge[(E(z,z)\wedge E(z,x))\rightarrow y=z]))\\
\wedge \quad & ((E(x,x)\wedge E(z,z))\rightarrow[\neg E(y,y)\wedge E(x,y)\wedge(E(x,z)\leftrightarrow E(y,z))])\,\}
\end{aligned}$$

A graph satisfies this formula if the following conditions are all satisfied.

1. The nodes without loops form a complete subgraph.
2. For every node x with a loop, there is exactly one y without a loop such that there is an edge from x to y.

3. For every node y without a loop, there is exactly one x with a loop such that there is an edge from x to y.

4. For all nodes x, z with loops, and y the unique node without a loop such that $E(x, y)$, it holds that $E(x, z)$ iff $E(y, z)$.

Property P_2 below is similar to P_f, except that the isomorphism is not explicitly given.

Definition 26. *A graph $G = (V, E)$ has P_2 if it satisfies the following conditions.*

1. *There is a partition*[1] $V_1, V_2 \subseteq V$ *such that* $|V_1| = |V_2|$, *there are loops* $(E(x, x))$ *on all* $x \in V_1$ *and no loops* $(\neg E(x, x))$ *for all* $x \in V_2$.

2. *The nodes without loops form a complete subgraph.*

3. *There are no edges from a node with a loop to a node without a loop.*

4. *There exists a bijection* $b \colon V_1 \to V_2$ *such that if* x, z *have loops, then* $E(x, z)$ *iff* $E(b(x), z)$.

It is not difficult to show that properties P_f and P_2 are indistinguishable.

Claim 3. *Properties P_f and P_2 are indistinguishable.*

Proof (Claim 3). Let $\varepsilon > 0$ be arbitrary and let $N_\varepsilon = \varepsilon^{-1}$. Assume that G has property P_2 and that $\#(G) > N_\varepsilon$. We will show that $\mathrm{mrdist}(G, P_f) < \varepsilon$.

Graph G has P_2 and so there is a bijection satisfying Condition 4 of Definition 26. We therefore add the edges $E(i, b(i))$ making the isomorphism (from V_1 to V_2) explicit. The resulting graph G_f has P_f.

[1] V_1, V_2 partition V if $V_1 \cap V_2 = \emptyset$ and $V_1 \cup V_2 = V$.

We have made exactly $n/2$ modifications, all to non-loops, and $n - 1 \geq N_\varepsilon$, so $\mathrm{mrdist}(G, P_f) \leq \mathrm{mrdist}(G, G_f) = 1/2(n-1) < \varepsilon$.

The converse is analogous; given a G that has P_f, simply remove the $n/2$ edges from loops to non-loops after using them to construct a suitable bijection b.

$\square$ Claim 3

Properties P_f and P_2 are indistinguishable and so (by Lemma 7), it suffices to show that P_2 is is untestable. Lemma 23 below is stronger than necessary, and actually implies a $\Omega(\sqrt{n})$ lower bound for testing P_f per the discussion following Lemma 7. $\square$ Theorem 9

Lemma 23. *Fix $0 < \varepsilon < 1/2$. Any ε-tester for P_2 must perform $\Omega(\sqrt{n})$ queries.*

Proof (Lemma 23). The proof is via Yao's Principle (cf. Principle 1), and so we define two distributions, D_{no} and D_{yes} and show that all deterministic testers have an error-rate greater than $1/3$ for property P_2 when the input is chosen randomly from D_{no} with probability $1/2$ and from D_{yes} with probability $1/2$.

In the following, we consider a distribution over graphs of sufficiently large size $2n$, and an arbitrary fixed partition of the vertices into V_1 and V_2 such that $|V_1| = |V_2| = n$ (for example, let the vertices be the integers $V := [2n]$, $V_1 := [n]$ and $V_2 := V \backslash V_1$).

We begin with D_{no}, defined as the following distribution.

1. Place a loop on each vertex in V_1 and place no loops in V_2.
2. Place all edges (except loops) in $V_2 \times V_2$.

3. Place each possible edge (except loops) in $V_1 \times V_1$ and $V_2 \times V_1$ uniformly and independently with probability 1/2.

That is, D_{no} is the uniform distribution of graphs (with this particular partition) satisfying the first three conditions of P_2.

Next, we define D_{yes} as the following.

1. Choose uniformly a random bijection $\pi\colon V_1 \to V_2$.
2. Place a loop on each vertex in V_1 and place no loops in V_2.
3. Place all edges (except loops) in $V_2 \times V_2$.
4. For each possible edge $(i, j \neq i) \in V_1 \times V_1$, uniformly and independently place *both* (i, j) and $(\pi(i), j)$ with probability 1/2 (otherwise place *neither*).

It is easy to see that D_{yes} generates only positive instances. Next, we show that D_{no} generates negative instances with high probability.

Lemma 24. *Fix $0 < \varepsilon < 1/2$ and let n be sufficiently large. Then,*

$$\Pr_{G\sim D_{\text{no}}}[\operatorname{dist}(G, P_2) \leq \varepsilon] = o(1)\,.$$

Proof (Lemma 24). D_{no} is the uniform distribution over graphs of size $2n$ with a particular partition satisfying the first three conditions of P_2. Let G_ε be the set of graphs G' of size $2n$ satisfying these conditions and such that $\operatorname{dist}(G', P_2) \leq \varepsilon$ (regardless of partition).

Counting the number of such graphs shows

$$|G_\varepsilon| \leq \binom{2n}{n} 2^{n(n-1)} n! \sum_{i=0}^{\lceil \varepsilon 2n^2 \rceil} \binom{2n^2}{i} \leq \binom{2n}{n} 2^{n(n-1)} n! 2^{H(\epsilon)2n^2}\,,$$

where $H(\varepsilon) := -\varepsilon \log \varepsilon - (1-\varepsilon)\log(1-\varepsilon)$ is the binary entropy function (cf. Lemma 16.19 in Flum and Grohe [23] for the bound on the summation).

Distribution D_{no} produces each of $2^{n(n-1)}2^{n^2}$ graphs with equal probability, so

$$\begin{aligned}\Pr_{G\sim D_{\text{no}}}[\text{dist}(G,P_2)\le\varepsilon] &\le \frac{|G_\varepsilon|}{2^{n(n-1)+n^2}} \le \binom{2n}{n} n! 2^{H(\epsilon)2n^2}/2^{n^2} \\ &\approx \frac{4^n n! 2^{H(\epsilon)2n^2}}{\sqrt{\pi n}2^{n^2}} = o(1)\,.\end{aligned}$$

The approximation is asymptotically tight, which suffices. □ Lemma 24

We have shown that D_{yes} generates only positive instances and that (with high probability) D_{no} generates instances that are ε-far from P_2. Next, we show that (again, with high probability) the two distributions look the same to testers making only $o(\sqrt{n})$ queries.

The proof is similar to a proof by Alon and Blais [2]. We begin by defining two random processes, P_{no} and P_{yes}, which answer queries from testers and generate instances according to D_{no} and D_{yes}, respectively.

Process P_{no} is defined in the following way.

1. Choose uniformly a random bijection $\pi\colon V_1 \to V_2$.
2. Intercept all queries from the tester and respond as follows.
   (a) To queries $E(i,i)$ with $i \in V_1$: respond 1.
   (b) To queries $E(i,i)$ with $i \in V_2$: respond 0.
   (c) To queries $E(i,j)$ with $i \in V_1$ and $j \in V_2$: respond 0.
   (d) To queries $E(i,j)$ with $i \neq j \in V_2$: respond 1.

(e) To queries $E(i,j)$ with $i \neq j \in V_1$: quit if we have queried $E(\pi(i),j)$, otherwise respond 1 or 0 randomly with probability 1/2 in each case.

(f) To queries $E(i,j)$ with $i \in V_2$ and $j \in V_1$: quit if we have queried $E(\pi^{-1}(i),j)$, otherwise respond 1 or 0 randomly with probability 1/2 in each case.

3. When the process has quit or the tester has finished its queries, complete the generated instance in the following way. First, fix the edges that were queried according to our answer. Next, place loops on each vertex in V_1, no loops in V_2, all non-loop edges in V_2 and no edges from V_1 to V_2. For each remaining possible edge, place it (uniformly, independently) with probability 1/2, *ignoring* π.

We define P_{yes} in the same way, except for the final step. When P_{yes} quits or the tester finishes, it fixes the edges that were queried according to its answers, and *also* fixes the corresponding edges (when relevant) according to π. More precisely, for each fixed $E(i,j)$ with $i \neq j \in V_1$, we also fix $E(\pi(i),j)$ and for fixed $E(i,j)$ with $i \in V_2, j \in V_1$, we also fix $E(\pi^{-1}(i),j)$, in both cases the same as our response to $E(i,j)$ (not randomly). The remaining edges are placed as in P_{no}.

Note that P_{no} generates instances according to D_{no} and P_{yes} generates instances according to D_{yes}. In addition, P_{yes} and P_{no} behave identically until they quit or answer all queries. In particular, if a tester does not cause the process to quit, the distribution of responses of its queries is identical for the two processes. We show that, with high probability, a tester that makes $o(\sqrt{n})$ queries does not cause either process to quit.

Lemma 25. *Let T be a deterministic tester which makes $o(\sqrt{n})$ queries, and let T interact with P_{yes} or P_{no}. In both cases,*

$$\Pr[T \text{ causes the process to quit}] = o(1)\,.$$

Proof (Lemma 25). The condition causing the process to quit is identical in P_{yes} and P_{no}. The probability that any pair of queries $E(i,j)$ and $E(i',j')$ cause the process to quit is at most

$$\Pr[i' = \pi(i) \text{ or } i = \pi(i')] \le \frac{(n-1)!}{n!} = 1/n\,.$$

The tester makes at most $o(\sqrt{n})$ queries and so

$$\Pr[T \text{ causes the process to quit}] \le o(\sqrt{n})^2 O(1/n) = o(1)\,.$$

□ Lemma 25

Any deterministic tester T which makes $o(\sqrt{n})$ queries can only distinguish between D_{yes} and D_{no} with probability $o(1)$, but it must accept D_{yes} with probability $2/3$, and reject D_{no} with probability $2/3 - o(1)$. It is impossible for T to satisfy both conditions, and the lemma follows from Principle 1. □ Lemma 23

5.3 Untestable Classes without Equality

We now consider classes without equality. Of course, any prefix class that is testable with equality remains testable without equality (because we are not forced to use equality). However, we now show that there are some untestable properties that are expressible in first-order logic even without equality.

Theorem 10. *There are properties in* $[\forall\exists\forall, (0,1)]$ *that are not* $\mathcal{T}_{mr}$*-testable, even given* $o(\sqrt{n})$ *queries.*

Note that Theorem 10 is restricted to $\mathcal{T}_{mr}$-testability. It is trivial to modify the proof for $\mathcal{T}_r$-testability of $[\forall\exists\forall, (1,1)]$. It remains open whether this class is $\mathcal{T}$-testable, however we suspect that $[\forall\exists\forall, (0,1)]$ is not $\mathcal{T}$-testable.

Proof (Theorem 10). The proof is similar to the proof of Theorem 9. We will begin by defining a property P_f that is expressible in our class. We will then define a property P which is indistinguishable from P_f, and use Yao's Principle to show that P is not $\mathcal{T}_{mr}$-testable.

A graph has property P_f if it satisfies the following conditions.

1. For every x with a loop, there is an outgoing edge to at least one y without a loop.

2. For every x without a loop, there is an incoming edge from at least one y without a loop.

3. There are no edges between vertices without loops.

4. For every x with a loop, there is an edge to at least one y without a loop such that for all z with loops, the following holds. There is a directed edge from y to z iff there are an odd number[2] of directed edges between x and z.

[2]There is an odd number of edges between x and z if there is a directed edge from x to z or from z to x, but not both. Note that a loop is counted as an even number of edges.

5. For every x without a loop, there is an incoming edge from at least one y with a loop such that for all z with loops, the following holds. There is a directed edge from x to z iff there are an odd number of directed edges between y and z.

More formally, P_f is the set of graphs that satisfy the following formula.

$$\begin{array}{ll}
\forall x \exists y \forall z : & \{\, (E(x,x) \to (\neg E(y,y) \wedge E(x,y))) \\
\wedge & \quad (\neg E(x,x) \to (E(y,y) \wedge E(y,x))) \\
\wedge & \quad ((\neg E(x,x) \wedge \neg E(z,z)) \to \neg E(x,z)) \\
\wedge & \quad ((E(x,x) \wedge E(z,z)) \to ((E(x,z) \oplus E(z,x)) \leftrightarrow E(y,z))) \\
\wedge & \quad ((\neg E(x,x) \wedge E(z,z)) \to ((E(y,z) \oplus E(z,y)) \leftrightarrow E(x,z))) \,\}
\end{array}$$

Next, we define a property P that we will show to be indistinguishable from P_f. A graph has property P if it satisfies the following conditions.

1. There is a partition of the vertices into (non-empty) V_1, V_2.

2. All vertices in V_1 have loops and no vertices in V_2 have loops.

3. There are no edges in $V_2 \times V_2$.

4. There exist functions $f\colon V_1 \to V_2$ and $g\colon V_2 \to V_1$ satisfying the following. For all $x, z \in V_1$, there is an edge from $f(x)$ to z iff there are an odd number of directed edges between x and z. For all $x \in V_2$ and $z \in V_1$, there is an edge from x to z iff there are an odd number of directed edges between $g(x)$ and z.

It is not difficult to show that P and P_f are indistinguishable.

Lemma 26. *Properties P and P_f are indistinguishable.*

Proof (Lemma 26). Let G be graph with property P_f and let $\varepsilon > 0$ be arbitrary. Then, G also has property P, that is $\mathrm{mrdist}(G, P) = 0$. In the other direction, if G has property P, then we can satisfy property P_f by adding at most $O(n)$ (non-loop) edges from x to $f(x)$ and $g^{-1}(y)$ to y. Thus, $\mathrm{mrdist}(G, P_f) \leq O(n)/\Theta(n^2) = o(1) < \varepsilon$ for sufficiently large graphs. □ Lemma 26

Indistinguishability preserves testability (cf. Lemma 7) and so it suffices to show that P is untestable. Lemma 27 below is stronger than necessary and actually implies a $\Omega(\sqrt{n})$ lower bound for testing P_f per the discussion following Lemma 7 in Section 3.3. □ Theorem 10

Lemma 27. *There is an $0 < \varepsilon < 1/2$ such that any $\mathcal{T}_{mr}$-style ε-tester for P must perform $\Omega(\sqrt{n})$ queries.*

Proof (Lemma 27). The proof is via Yao's Principle (cf. Principle 1) and so we must define a distribution of inputs and show that all deterministic ε-testers have an error rate greater than $1/3$ for P on inputs from the distribution. For our distribution, we will draw from a distribution D_{no} with probability $1/2$ and from a distribution D_{yes} with probability $1/2$.

In the following, we consider distributions over graphs with sufficiently large vertex set $[2n]$ and an arbitrary fixed partition of the vertices into V_1 and V_2 such that $|V_1| = |V_2| = n$.

We begin with D_{no}, defined as the following distribution.

1. Place loops on all vertices in V_1 and no loops in V_2.
2. For each ordered pair in $V_1 \times V_2$, place a directed edge with probability $1/2$.
3. For each unordered pair $(i, j \neq i) \in V_1 \times V_1$, with probability $1/2$ place no edge, and with probability $1/2$ place a single directed edge, from i to j if $i \leq j$ and from j to i if $j \leq i$.
4. For each ordered pair in $V_2 \times V_1$, with probability $1/2$ place the directed edge and with probability $1/2$ do not.

Note that D_{no} is the uniform distribution of graphs that satisfy the following conditions.

1. The vertex set is $[2n]$ and the vertices with loops follow the given partition.
2. There are no undirected edges between vertices with loops (when a loop is not considered an undirected edge).
3. There are no edges between vertices without loops.
4. All directed edges (i, j) between vertices with loops satisfy $i \leq j$.

Next, we define D_{yes}.

1. Place loops on all vertices in V_1 and no loops in V_2.
2. For each ordered pair in $V_1 \times V_2$, place a directed edge with probability $1/2$.
3. Choose uniformly a random bijection $\pi \colon V_1 \to V_2$.

4. For each unordered pair in $(i, j \neq i) \in V_1 \times V_1$, with probability $1/2$ do the following. Place directed edges $(\pi(i), j)$ and $(\pi(j), i)$, and then place either (i, j) (if $i \leq j$) *or* (j, i) (if $j \leq i$). Otherwise, do not place any edges right now.

Distribution D_{yes} generates only positive instances for P. Now, we show that with high probability, D_{no} generates instances that are ε-far.

Lemma 28. *Let $\varepsilon > 0$ be sufficiently small and n be sufficiently large. Then,*

$$\Pr_{G \sim D_{no}} [\mathrm{mrdist}(G, P) \leq \varepsilon] = o(1)\,.$$

Proof (Lemma 28). Distribution D_{no} is the uniform distribution over graphs of size $2n$ with a fixed partition V_1, V_2 satisfying the following.

1. All vertices in V_1 have loops and no vertices in V_2 have loops.

2. There are no undirected edges between vertices with loops.

3. There are no edges between vertices without loops.

4. All directed edges (x, y) between vertices with loops satisfy $x \leq y$.

We want a small upper-bound on the probability of a graph being drawn from D_{no} that is not ε-far from P. D_{no} is the uniform distribution over a certain class of graphs, and so this probability is

$$\frac{|\{G \mid G \sim D_{\text{no}}, \mathrm{mrdist}(G, P) \leq \varepsilon\}|}{|\{G \mid G \sim D_{\text{no}}\}|}\,.$$

The number of distinct graphs produced by D_{no} is $2^{\binom{n}{2}} 2^{2n^2} = 2^{2.5n^2 - n/2}$.

Let G_{2n} be the set of graphs with vertices $[2n]$ that have property P and are not ε-far from all graphs in D_{no}. Then,

$$\Pr_{G\sim D_{\text{no}}}[\text{mrdist}(G,P)\le\varepsilon]\le\frac{|G_{2n}|\sum_{i=0}^{\lfloor 4\varepsilon n^2\rfloor}\binom{4n^2}{i}}{2^{2.5n^2-n/2}}\,. \tag{5.14}$$

Any graph G that is not ε-far from all graphs in D_{no} must have loops on $n-\varepsilon n\le j\le n+\varepsilon n$ vertices. Therefore, $|G_{2n}|\le$

$$\sum_{j=n-\varepsilon n}^{n+\varepsilon n}\binom{2n}{j}4^{\binom{j}{2}}2^{j(2n-j)}j^{2n-j}\le(2\varepsilon n+1)\binom{2n}{n}2^{2\binom{n+\varepsilon n}{2}+(n+\varepsilon n)^2+(n+\varepsilon n)\log{(n+\varepsilon n)}}\,. \tag{5.15}$$

Using the (asymptotically tight) $\binom{2n}{n}\approx 4^n/\sqrt{\pi n}$, we see that (5.15) is approximately

$$\frac{(2\varepsilon n+1)}{\sqrt{\pi n}}2^{2n+(n+\varepsilon n)^2+(n+\varepsilon n)(n+\varepsilon n-1)+(n+\varepsilon n)\log(n+\varepsilon n)}\,.$$

Combining this with (5.14) and using $\sum_{i=0}^{\lfloor\varepsilon 4n^2\rfloor}\binom{4n^2}{i}\le 2^{H(\varepsilon)4n^2}$, where $H(\varepsilon)=-\varepsilon\log\varepsilon-(1-\varepsilon)\log(1-\varepsilon)$ is the binary entropy function (cf. Lemma 16.19 in Flum and Grohe [23]), we get $\Pr_{G\sim D_{\text{no}}}[\text{mrdist}(G,P)\le\varepsilon]\le$

$$\frac{2\varepsilon n+1}{\sqrt{\pi n}}2^{-n^2/2+4H(\varepsilon)n^2+3/2n+2(\varepsilon^2+\varepsilon)n^2+\varepsilon n+(n+\varepsilon n)\log{(n+\varepsilon n)}}=o(1)\,,$$

because the $-n^2/2$ in the exponent dominates when ε is sufficiently small.

□ Lemma 28

We have shown that D_{yes} generates only positive instances and, with high probability, D_{no} generates ε-far instances. Next, we show that, with high probability, the two distributions look identical to testers making only $o(\sqrt{n})$ queries. The proof is similar to a proof by Alon and Blais [2].

We begin by defining two random processes, P_{no} and P_{yes}, which answer queries from testers and generate instances according to D_{no} and D_{yes}, respectively.

Process P_{no} is defined in the following way.

1. Choose uniformly a random bijection $\pi \colon V_1 \to V_2$.
2. Intercept all queries from the tester and respond as follows.
   (a) To queries $E(i, i)$ with $i \in V_1$, respond 1.
   (b) To queries $E(i, i)$ with $i \in V_2$, respond 0.
   (c) To queries $E(i, j)$ with $i \in V_2, j \in V_2$, respond 0.
   (d) To queries $E(i, j)$ with $i \in V_1, j \in V_2$, randomly respond 1 or 0 with probability 1/2 in each case.
   (e) To queries $E(i, j)$ with $i > j \in V_1$, respond 0.
   (f) To queries $E(i, j)$ with $i < j \in V_1$, quit if we have queried $E(\pi(i), j)$ or $E(\pi(j), i)$. Otherwise randomly respond 1 or 0 with probability 1/2 in each case.
   (g) To queries $E(i, j)$ with $i \in V_2, j \in V_1$, quit if we have queried $E(\pi^{-1}(i), j)$ or $E(j, \pi^{-1}(i))$. Otherwise randomly respond 1 or 0 with probability 1/2 in each case.
3. When the process has quit, or the tester has finished its queries, complete the generated instance in the following way. First, fix the edges that were queried according to our answers. Next, place loops on all vertices in V_1, no loops in V_2 and no edges internal to V_2. Place each edge in $V_1 \times V_2$ uniformly

and independently with probability 1/2. For each remaining possible edge $(i, j) \in V_1 \times V_1$, place the edge uniformly and independently with probability 1/2 if $i < j$ and do not place the edge if $i > j$. For each remaining possible edge in $V_2 \times V_1$, place the edge uniformly and independently with probability 1/2 (ignoring π).

We define P_{yes} in the same way, except for the final step. When P_{yes} quits or the tester finishes, it fixes the edges that were queried according to its answers, and also fixes the corresponding edges (when relevant) according to π. More precisely, for each fixed $E(i, j)$ with $i \neq j \in V_1$, we also fix $E(\pi(i), j)$ and $E(j, \pi(i))$, and for fixed $E(i, j)$ with $i \in V_2, j \neq \pi^{-1}(i) \in V_1$, we also fix $E(\pi^{-1}(i), j)$ and $E(j, \pi^{-1}(i))$, in both cases according to our previous decision. The remaining edges are placed as in P_{no}.

Note that P_{no} generates instances according to D_{no} and P_{yes} generates instances according to D_{no}. In addition, P_{yes} and P_{no} behave identically until they quit or answer all queries. In particular, if a tester does not cause the process to quit, the distribution of responses to queries is identical for the two processes. We show that, with high probability, a tester that makes $o(\sqrt{n})$ queries does not cause either process to quit.

Lemma 29. *Let T be a deterministic tester which makes $o(\sqrt{n})$ queries, and let T interact with P_{yes} or P_{no}. In both cases,*

$$\Pr[T \text{ causes the process to quit}] = o(1)\,.$$

Proof (Lemma 29). The condition causing the process to quit is identical in P_{yes}

and P_{no}. The probability that any fixed pair of queries $E(i,j)$ and $E(i',j')$ cause the process to quit is at most

$$\Pr[i' = \pi(i) \text{ or } i' = \pi(j)] \leq \frac{2(n-1)!}{n!} = 2/n\,.$$

The tester makes at most $o(\sqrt{n})$ queries and so

$$\Pr[T \text{ causes the process to quit}] \leq o(\sqrt{n})^2 O(1/n) = o(1)\,.$$

$\square$ Lemma 28

Any deterministic tester T which makes $o(\sqrt{n})$ queries can only distinguish between D_{yes} and D_{no} with probability $o(1)$, but it must accept D_{yes} with probability at least $2/3$ and reject D_{no} with probability at least $2/3 - o(1)$. It is impossible for T to satisfy both conditions, so the lemma follows from Principle 1.

$\square$ Lemma 27

5.4 Summary

In this chapter, we proved that several classes of first-order properties of directed graphs are not testable. We used a different variant of graph isomorphism in each case. We began with the prefix $\forall^3\exists$ in Section 5.1. This result first appeared in [40], and we thank an anonymous referee from LATA for improving it significantly. This result will also appear in [42].

In Section 5.2, we proved the untestability of $\forall\exists\forall$ by using a variant of graph isomorphism related to Boolean function isomorphism. The connection with Boolean function isomorphism allows us to leverage recent ideas from Alon and Blais [2]. This result first appeared in [41].

Finally, we considered classes without equality in Section 5.3. There, we showed that, even without equality, $\forall\exists\forall$ can express untestable properties of directed graphs. However, this proof is currently limited to $\mathcal{T}_{mr}$-style testing and it remains open whether this class is $\mathcal{T}_r$-testable. Our suspicion is that it is untestable in all of our models. This result appears here for the first time.

We're grateful to Neil Immerman for pointing out that removing equality in our untestable properties does not really change the "spirit" of why they're untestable. Section 5.3 formalizes this for one of our classes; we suspect that similar arguments will also allow us to remove equality from the classes in Section 5.1. We're also grateful to Hiro Ito for pointing out an omission in a previous version of the proof of Lemma 23.

Chapter 6

Conclusions

In this thesis, we focused on the testability of prefix-vocabulary classes of first-order logic, extending work that was initiated by Alon *et al.* [3]. They first considered the idea of testing syntactic subclasses of first-order logic, and showed that all properties of undirected, loop-free graphs expressible in first-order sentences with quantifier pattern $\exists^*\forall^*$ are testable, while there exists an untestable property expressible with quantifier pattern $\forall^*\exists^*$.

Their proof of the latter result implies upper bounds of twelve, five and seventeen for the minimum number of universal, existential and total quantifiers, respectively, sufficient to express an untestable property. One of our goals was to optimize these bounds and find the minimum number of universal and existential quantifiers, as well as quantifiers in total, sufficient to express an untestable property. Our results imply that these minima are two universal, one existential and three total quantifiers, respectively. In addition, we remove the restriction to undirected, loop-free graphs and focus on relational structures.

Our main results are as follows. First, we proved that all properties expressible in Ackermann's class with equality ($[\exists^*\forall\exists^*, \mathit{all}\,]_=$) are testable. Then, we extended the positive result of Alon *et al.* [3] from undirected, loop-free graphs to relational structures by using a result from Austin and Tao [11]. This answers a question of Fischer [20] on the testability of hypergraph properties expressible with quantifier pattern $\exists^*\forall^*$, although much of the work for this case is by Austin and Tao [11]. Next, we simplified the untestable property of Alon *et al.* [3] and showed that there are untestable properties of directed graphs expressible with quantifier prefixes $\forall^3\exists$. Finally, we used a variant of graph isomorphism related to Boolean function isomorphism to prove that there are untestable properties of directed graphs expressible with prefix $\forall\exists\forall$ (for $\mathcal{T}_{mr}$ testability, this remains true without equality).

The current classification of prefix-vocabulary classes for testability is the following.

- Testable classes
  1. Monadic first-order logic: $[\mathit{all}\,, (\omega)]_=$.
  2. Ackermann's class with equality: $[\exists^*\forall\exists^*, \mathit{all}\,]_=$.
  3. Ramsey's class: $[\exists^*\forall^*, \mathit{all}\,]_=$.
- Untestable classes
  1. $[\forall^3\exists, (0, 1)]_=$.
  2. $[\forall\exists\forall, (0, 1)]_=$.
  3. $[\forall\exists\forall, (0, 1)]$[1].

[1]We only prove this for $\mathcal{T}_{mr}$ testability, although we suspect that it also holds for our other models.

It is interesting to compare this classification for testability with known (complete) classifications for other properties. For example, the current classification for testability is consistent with the classifications for the finite model property (see, e.g., Chapter 6 of Börger *et al.* [15]), for docility[2] (see Kolaitis and Vardi [46]) and for 0-1 laws for fragments of existential second-order logic (see Kolaitis and Vardi [46]). These classifications may be helpful in providing guidance in the classification for testability.

This similarity between classifications may indicate a deeper connection between these seemingly distinct properties. We would like to know which (if any) of the traditional classifications coincides with the classification for testability, and hope to understand the connections between testability and other properties of prefix-vocabulary classes.

As concrete open problems, we are especially interested in the testability of $[\forall^3\exists, (0,1)]$ (without equality) and variants of the Gödel class (i.e., classes whose prefix contain at least $\forall^2\exists$). Determining the testability of these classes may suffice to complete the classification. We are also interested in the special case of predicate logic with equality.

There are *many* possible variations of the classification for testability. For example, one could be more interested in classes which are *constructively* testable, i.e., where it is possible to compute an ε-tester given ε and a formula from the class. However, in the present paper we are fortunate that many of the possible classifications coincide. Namely, all of our positive results are for constructive (and therefore uniform) testability in the most-restricted model ($\mathcal{T}_{mr}$) that we con-

[2]A class is said to be *docile* (or decidable for finite satisfiability) if given an arbitrary formula from the class, one can decide if there exists a *finite* model of the property.

sider, while all of the negative results hold even for non-uniform testability in the least-restricted model ($\mathcal{T}$).

As is common in the literature, we have focused on *testable* properties, i.e., those which can be approximated with a number of queries (or time) depending on ε but not on n. Given the deep connection between dense graph property testing and applications of strong versions of Szemerédi's regularity lemma (see Alon *et al.* [4]), it is not surprising that there are testable properties with enormous query complexity (e.g., towers of $1/\varepsilon$ with height growing in $1/\varepsilon$). Even relatively simple properties such as triangle-freeness[3] are testable, but not with query complexity polynomial in $1/\varepsilon$[4]. However, in practice we are perhaps more interested in properties that can be tested with a number of queries polynomial in $1/\varepsilon$.

Our construction for Ramsey's class (cf. Section 4.2) results in a rather large query complexity. However, this class contains the problems of triangle-freeness and $\mathcal{R}$-freeness (for finite sets $\mathcal{R}$ of finite structures). Significantly improving the query complexity of these problems is considered to be a difficult open problem, and so significantly improving the query complexity for Ramsey's class (in general) is likely to be challenging.

[3]A graph is triangle-free if it contains no 3-cliques.

[4]See Alon and Shapira [6] for the lower-bound. The best known upper-bound, a tower of $1/\varepsilon$ of height logarithmic in $1/\varepsilon$ is due to Fox [24].

Bibliography

[1] Wilhelm Ackermann. Über die Erfüllbarkeit gewisser Zählausdrücke. *Mathematische Annalen*, 100:638–649, 1928.

[2] Noga Alon and Eric Blais. Testing Boolean function isomorphism. In *Approximation, Randomization, and Combinatorial Optimization, Algorithms and Techniques, 13th International Workshop, APPROX 2010, and 14th International Workshop, RANDOM 2010, Barcelona, Spain, September 2010, Proceedings*, volume 6302 of *Lecture Notes in Computer Science*, pages 394–405. Springer, 2010.

[3] Noga Alon, Eldar Fischer, Michael Krivelevich, and Mario Szegedy. Efficient testing of large graphs. *Combinatorica*, 20(4):451–476, 2000.

[4] Noga Alon, Eldar Fischer, Ilan Newman, and Asaf Shapira. A combinatorial characterization of the testable graph properties: It's all about regularity. In *STOC '06: Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 251–260, New York, NY, USA, 2006. ACM.

[5] Noga Alon, Michael Krivelevich, Ilan Newman, and Mario Szegedy. Regular languages are testable with a constant number of queries. *SIAM Journal on*

Computing, 30(6):1842–1862, 2001.

[6] Noga Alon and Asaf Shapira. A characterization of easily testable induced subgraphs. *Combinatorics, Probability and Computing*, 15(6):791–805, 2006.

[7] Noga Alon and Asaf Shapira. Homomorphisms in graph property testing. In M. Klazar, J. Kratochvíl, M. Loebl, J. Matoušek, R. Thomas, and P. Valtr, editors, *Topics in Discrete Mathematics*, volume 26 of *Algorithms and Combinatorics*, pages 281–313. Springer, 2006.

[8] 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.

[9] Noga Alon and Asaf Shapira. A separation theorem in property testing. *Combinatorica*, 28(3):261–281, 2008.

[10] Noga Alon and Joel H. Spencer. *The Probabilistic Method.* Wiley, third edition, 2008.

[11] Tim Austin and Terence Tao. On the testability and repair of hereditary hypergraph properties. *Random Structures & Algorithms*, 36(4):373–463, 2010.

[12] Paul Bernays and Moses Schönfinkel. Zum Entscheidungsproblem der mathematischen Logik. *Mathematische Annalen*, 99(1):342–372, 1928.

[13] Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. In *Proceedings, 26th Annual IEEE Confer-*

ence on Computational Complexity, CCC 2011, 8–10 June 2011, San Jose, California, pages 210–220. IEEE Computer Society, 2011.

[14] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993.

[15] Egon Börger, Erich Grädel, and Yuri Gurevich. *The Classical Decision Problem*. Springer-Verlag, 1997.

[16] J. Richard Büchi. Weak second-order arithmetic and finite-automata. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 6:66–92, 1960.

[17] Hana Chockler and Orna Kupferman. ω-regular languages are testable with a constant number of queries. *Theoretical Computer Science*, 329(1-3):71–92, 2004.

[18] Christian Fermüller and Gernot Salzer. Ordered paramodulation and resolution as decision procedure. In *Logic Programming and Automated Reasoning, 4th International Conference, LPAR'93, St. Petersburg, Russia, July 13-20, 1993, Proceedings*, volume 698 of *Lecture Notes in Computer Science*, pages 122–133. Springer, 1993.

[19] Eldar Fischer. The art of uninformed decisions. *Bulletin of the European Association for Theoretical Computer Science*, 75:97–126, October 2001. Columns: Computational Complexity.

[20] Eldar Fischer. Testing graphs for colorability properties. *Random Structures & Algorithms*, 26(3):289–309, 2005.

[21] Eldar Fischer and Arie Matsliah. Testing graph isomorphism. *SIAM Journal on Computing*, 38(1):207–225, 2008.

[22] Eldar Fischer, Arie Matsliah, and Asaf Shapira. Approximate hypergraph partitioning and applications. *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2007*, pages 579–589, 2007.

[23] Jörg Flum and Martin Grohe. *Parametrized Complexity Theory*. Springer, 2006.

[24] Jacob Fox. A new proof of the graph removal lemma. *Annals of Mathematics*, 174(1):561–579, 2011.

[25] Rūsiņš Freivalds. Fast probabilistic algorithms. In *Mathematical Foundations of Computer Science 1979, Proceedings, 8th Symposium, Olomouc, Czechoslovakia, September 3-7, 1979*, volume 74 of *Lecture Notes in Computer Science*, pages 57–69. Springer-Verlag, 1979.

[26] John Gill. Computational complexity of probabilistic Turing machines. *SIAM Journal on Computing*, 6(4):675–695, 1977.

[27] O. Goldreich and D. Ron. Property testing in bounded degree graphs. *Algorithmica*, 32:302–343, 2002.

[28] Oded Goldreich. Introduction to testing graph properties. Technical Report TR10-082, Electronic Colloquium on Computational Complexity (ECCC), May 2010.

[29] 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.

[30] W. T. Gowers. Hypergraph regularity and the multidimensional Szemerédi theorem. *Annals of Mathematics*, 166(3):897–946, 2007.

[31] Erich Grädel. Satisfiability of formulae with one $\forall$ is decidable in exponential time. *Archive for Mathematical Logic*, 29:256–276, 1990.

[32] Yuri Gurevich. The decision problem for the logic of predicates and operations. *Algebra i Logika*, 8:284–308, 1969. In Russian.

[33] Yuri Gurevich. The decision problem for the logic of predicates and operations. *Algebra and Logic*, 8:160–174, 1969.

[34] Yuri Gurevich. On the classical decision problem. *Bulletin of the European Association for Theoretical Computer Science*, pages 140–150, October 1990.

[35] Joseph Y. Halpern, Robert Harper, Neil Immerman, Phokion G. Kolaitis, Moshe Y. Vardi, and Victor Vianu. On the unusual effectiveness of logic in computer science. *The Bulletin of Symbolic Logic*, 7(2):213–236, 2001.

[36] John E. Hopcroft and Jefferey D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Adison-Wesley Publishing Company, Reading, Massachusets, USA, 1979.

[37] Yoshiyasu Ishigami. Removal lemma for infinitely-many forbidden hypergraphs and property testing. arXiv:math/0612669v2, 2008.

[38] Charles Jordan and Thomas Zeugmann. Relational properties expressible with one universal quantifier are testable. In Osamu Watanabe and Thomas Zeugmann, editors, *Stochastic Algorithms: Foundations and Applications, 5th International Symposium, SAGA 2009, Sapporo, Japan, October 2009, Proceedings*, volume 5792 of *Lecture Notes in Computer Science*, pages 141–155. Springer, 2009.

[39] Charles Jordan and Thomas Zeugmann. A note on the testability of Ramsey's class. In Jan Kratochvíl, Angsheng Li, Jiří Fiala, and Petr Kolman, editors, *Theory and Applications of Models of Computation, 7th International Conference, TAMC 2010, Prague, Czech Republic, June 2010, Proceedings*, volume 6108 of *Lecture Notes in Computer Science*, pages 296–307. Springer, 2010.

[40] Charles Jordan and Thomas Zeugmann. Untestable properties expressible with four first-order quantifiers. In Adrian-Horia Dediu, Henning Fernau, and Carlos Martín-Vide, editors, *Language and Automata Theory and Applications, 4th International Conference, LATA 2010, Trier, Germany, May 2010, Proceedings*, volume 6031 of *Lecture Notes in Computer Science*, pages 333–343. Springer, 2010.

[41] Charles Jordan and Thomas Zeugmann. Untestable properties in the Kahr-Moore-Wang class. In Lev D. Beklemishev and Ruy de Queiroz, editors, *Logic, Language, Information and Computation, 18th International Workshop, WoLLIC 2011, Philadelphia, PA, USA, May 18-21, 2011, Proceedings*, volume 6642 of *Lecture Notes in Artificial Intelligence*, pages 176–186. Springer, 2011.

[42] Charles Jordan and Thomas Zeugmann. Testable and untestable classes of first-order formulae. *Journal of Computer and System Sciences*, to appear, 2012.

[43] Skip Jordan and Thomas Zeugmann. Indistinguishability and first-order logic. In Manindra Agrawal, Dingzhu Du, Zhenhua Duan, and Angsheng Li, editors, *Theory and Applications of Models of Computation, 5th International Conference, TAMC 2008, Xi'an, China, April 2008, Proceedings*, volume 4978 of *Lecture Notes in Computer Science*, pages 94–104. Springer, 2008.

[44] P. Kolaitis and M. Vardi. The decision problem for the probabilities of higher-order properties. In *STOC '87: Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 425–435, New York, NY, USA, 1987. ACM.

[45] Phokion G. Kolaitis and Moshe Y. Vardi. 0-1 laws and decision problems for fragments of second-order logic. *Information and Computation*, 87(1-2):302–338, 1990.

[46] Phokion G. Kolaitis and Moshe Y. Vardi. 0-1 laws for fragments of existential second-order logic: A survey. In Mogens Nielsen and Branislav Rovan, editors, *Mathematical Foundations of Computer Science 2000, 25th International Symposium, MFCS 2000, Bratislava, Slovakia, August/September 2000, Proceedings*, volume 1893 of *Lecture Notes in Computer Science*, pages 84–98. Springer, 2000.

[47] K. de Leeuw, E. F. Moore, C. E. Shannon, and N. Shapiro. Computability by probabilistic machines. In C.E. Shannon and J. McCarthy, editors, *Automata Studies*, pages 183–212. Princeton University Press, Princeton, NJ, 1956.

[48] Harry R. Lewis. Complexity results for classes of quantificational formulas. *Journal of Computer and System Sciences*, 21(3):317–353, 1980.

[49] Leonid Libkin. Expressive power of SQL. *Theoretical Computer Science*, 296:379–404, 2003.

[50] László Lovász. Some mathematics behind graph property testing. In Yoav Freund, László Györfi, György Turán, and Thomas Zeugmann, editors, *Algorithmic Learning Theory, 19th International Conference, ALT 2008, Budapest, Hungary, October 2008, Proceedings*, volume 5254 of *Lecture Notes in Computer Science*, page 3. Springer, 2008.

[51] Leopold Löwenheim. Über Möglichkeiten im Relativkalkül. *Mathematische Annalen*, 76:447–470, 1915.

[52] Robert McNaughton and Seymour Papert. *Counter-Free Automata.* M.I.T. Press, 1971.

[53] J. v. Neumann. Zur Theorie der Gesellschaftsspiele. *Mathematische Annalen*, 100(1):295–320, 1928.

[54] Eugenio Omodeo and Alberto Policriti. The Bernays-Schönfinkel-Ramsey class for set theory: Semidecidability. *Journal of Symbolic Logic*, 75(2):459–480, June 2010.

[55] Michal Parnas and Dana Ron. Testing the diameter of graphs. *Random Structures & Algorithms*, 20(2):165–183, 2002.

[56] F. P. Ramsey. On a problem of formal logic. *Proceedings of the London Mathematical Society, series 2*, 30:264–286, 1930.

[57] Vojtěch Rödl and Mathias Schacht. Generalizations of the removal lemma. *Combinatorica*, 29(4):467–501, 2009.

[58] Vojtěch Rödl and Mathias Schacht. Regular partitions of hypergraphs: Regularity lemmas. *Combinatorics, Probability and Computing*, 16(6):833–885, 2007.

[59] Vojtěch Rödl and Mathias Schacht. Regularity lemmas for graphs. In Gyula O. H. Katona, Alexander Schrijver, Tamás Szőnyi, and Gábor Sági, editors, *Fete of Combinatorics and Computer Science*, volume 20 of *Bolyai Society Mathematical Studies*, pages 287–325, 2010.

[60] Dana Ron. Property testing. In Sanguthevar Rajasekaran, Panos M. Pardalos, John H. Reif, and José Rolim, editors, *Handbook of Randomized Computing*, volume II, chapter 15, pages 597–649. Kluwer Academic Publishers, 2001.

[61] Dana Ron. Property testing: A learning theory perspective. *Foundations and Trends in Machine Learning*, 1(3):307–402, 2008.

[62] Dana Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends in Theoretical Computer Science*, 5(2):73–205, 2009.

[63] Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.

[64] Saharon Shelah. Decidability of a portion of the predicate calculus. *Israel Journal of Mathematics*, 28(1-2):32–44, 1977.

[65] Stephen Simons. Minimax theorems and their proofs. In Ding-Zhu Du and Panos M. Pardalos, editors, *Minimax and Applications*, pages 1–23. Kluwer Academic Publishers, 1995.

[66] Th. Skolem. Untersuchungen über die Axiome des Klassenkalküls und über Produktations und Summationsprobleme, welche gewisse Klassen von Aussagen betreffen. *Videnskapsselskapets skrifter, I. Matematisk-naturvidenskabelig klasse*, (3):37–71, 1919.

[67] Th. Skolem. Logisch-kombinatorische Untersuchungen über die Erfüllbarkeit und Beweisbarkeit mathematischer Sätze nebst einem Theoreme über dichte Mengen. *Videnskapsselskapets skrifter, I. Matematisk-naturvidenskabelig klasse*, (4):1–36, 1920.

[68] Terence Tao. A variant of the hypergraph removal lemma. *Journal of Combinatorial Theory, Series A*, 113(7):1257–1280, 2006.

[69] Andrew Chi-Chih Yao. Probabilistic computations: Toward a unified measure of complexity. In *18th Annual Symposium on Foundations of Computer Science*, pages 222–227. IEEE Computer Society, 1977.