Relational Properties Expressible with One Universal Quantifier Are Testable

Charles Jordan[⋆] and Thomas Zeugmann[⋆⋆]

Division of Computer Science
Hokkaido University, N-14, W-9, Sapporo 060-0814, Japan
{skip,thomas}@ist.hokudai.ac.jp

Abstract. In property testing a small, random sample of an object is taken and one wishes to distinguish with high probability between the case where it has a desired property and the case where it is *far* from having the property. Much of the recent work has focused on graphs. In the present paper three generalized models for testing relational structures are introduced and relationships between these variations are shown. Furthermore, the logical classification problem for testability is considered and, as the main result, it is shown that *Ackermann's class with equality is testable.*

Key words: property testing, logic

1 Introduction

Property testing is an application of induction. Given a large object such as a graph or database, we wish to state a conclusion about the entire object after examining a small, randomly selected sample. Lovász [19] has described it as the "third reincarnation" of this approach, after statistics and machine learning.

Property testers are probabilistic approximation algorithms that examine only a small part of their input. Our goal is always to distinguish inputs that have some desired property from inputs that are *far* from having it. We are especially interested in *classification*, i.e., the testability of large classes of properties.

The paper is structured as follows. In Subsection 1.1, we outline the history of testing, focusing on results that influence our approach. Much recent work has focused on graphs, while we seek a general framework that we call *relational property testing*. Definitions and notation are in Section 2. In Section 3 we show the relationships between variations of our framework. We use the framework from previous sections to state the *classification problem for testability* in Section 4, and show that Ackermann's class with equality is testable (cf. Theorem 4) in all of the variations considered in previous sections.

[⋆] Supported by a Grant-in-Aid for JSPS Fellows under Grant No. 2100195209.
[⋆⋆] Supported by MEXT Grand-in-Aid for Scientific Research on Priority Areas under Grant No. 21013001.

1.1 History of Property Testing

We begin with a brief history and overview of property testing. There are also a number of surveys of property testing, see for example Fischer [12] or Ron [25].

Property testing is a form of approximation where we trade accuracy for efficiency. Probabilistic machines seem to have been first formalized by de Leeuw *et al.* [18], who 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. An early example of such a result is Freivalds' [14] matrix multiplication checker.

Property testing itself began in program verification (see Blum *et al.* [8] and Rubinfeld and Sudan [26]). Goldreich *et al.* [16] first considered the testability of graph properties 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 [15]. Parnas and Ron [22] generalized this approach and attempted to move away from the functional representation of structures.

For other types of structures, Alon *et al.* [4] showed that the regular languages are testable and that there exist untestable context-free languages. Chockler and Kupferman [11] extended the positive result to the ω-regular languages.

There is also recent work on testing properties of (usually uniform) hypergraphs. In particular, Fischer *et al.* [13] defined a general model that is roughly equivalent to one of our models, namely $\mathcal{T}_r$ based on Definition 8, and showed that hypergraph partition problems are testable in this framework. However, much of the recent work has been focused on graphs and Alon and Shapira [6] survey some of the recent results in testing graph properties.

Alon *et al.* [2] began a *logical* characterization of the testable (graph) properties, see Section 4. Alon and Shapira [5] gave a (near) characterization of a natural subclass of the testable graph properties, which Rödl and Schacht [24] generalized to hypergraphs. Alon *et al.* [3] showed a combinatorial characterization of the graph properties testable with a constant number of queries.

2 Preliminaries

Instead of restricting our attention to, for example, graphs, we focus on property testing in a general setting. We begin by defining vocabularies.

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

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

Two examples 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* structure A of type τ *is an* $(s+1)$*-tuple*

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

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 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."

The set of all structures of type τ and universe size n is $STRUC^n(\tau)$ and the set of all structures of type τ is $STRUC(\tau) := \bigcup_{0 \le n} STRUC^n(\tau)$. A *property of type* τ is any subset of $STRUC(\tau)$. For $A \in P$, we say *A has P*. We use *language* to refer to string properties, P to denote properties and $P_1 \backslash P_2$ for set difference.

2.1 Property Testing Definitions

We wish to distinguish, with high probability, between inputs that have a desired property and inputs that are *far* from having the property. We begin by defining a *distance measure* between structures. Changing the definition of distance results in a different model for relational testing. The symbol $\oplus$ is exclusive-or.

Definition 3. *Let $A, B \in STRUC(\tau)$ be any structures such that $\#(A) = \#(B) = n$. The* distance *between structures A and B is*

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

The dist distance is the fraction of assignments on which the two structures disagree. It is equivalent to the definition that would result from mapping relational structures to binary strings and using the usual definitions for testing strings. We now give the remaining definitions for testing, and will then give alternatives to Definition 3 (cf. Definitions 8 and 12).

Definition 4. *Let P be a property of structures with vocabulary τ and let A be such a structure with a universe of size n. Then,*

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

Definition 5. *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* $\mathrm{dist}(A, P) \ge \varepsilon$.

Definition 6. *Property P is* testable *if for all $\varepsilon > 0$ there are ε-testers making a number of queries which is upper-bounded by a function depending only on ε.*

We allow different ε-testers for each $\varepsilon > 0$. The situation is similar to that familiar from circuit complexity (cf. Straubing [30]), where we have uniform and non-uniform cases, see, e.g., Alon and Shapira [7]. Our results hold in both cases and so we will not distinguish between them.

2.2 Logical Definitions

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 formulas $x_i = x_j$ 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 interpreted as usual and defines property $P := \{A \mid A \in \mathit{STRUC}(\tau) \text{ and } A \models \varphi\}$. Lower-case Greek letters φ, ψ and γ refer to first-order formulas and x, y, and z to first-order variables. Our classification definitions are from Börger *et al.* [9] except that we omit function symbols. The following is for completeness, where $\mathbb{N} = \{0, 1, \ldots\}$ is the set of natural numbers.

Definition 7. *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 either the special phrase 'all' or a sequence over* $\mathbb{N}$ *and the first infinite ordinal* ω*, and* e *is '=' or* λ*.*

The first-order sentence $\varphi := \pi_1 x_1 \pi_2 x_2 \ldots \pi_r x_r : \psi$ in prenex normal form, with quantifiers π_i and quantifier-free ψ, is a member of the prefix vocabulary class given by $[\Pi, (p_1, p_2, \ldots)]_e$, where $p_i \in \mathbb{N} \cup \{\omega\}$ iff

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. 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 if the equality symbol is permitted.

A prefix class is *testable* if every formula in it expresses a testable property for every vocabulary in which it is evaluable. 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 property that is testable in the context of* τ*, then the property of type* τ' *defined by* φ *is also testable.*

Proof. Let φ define property P of type τ and property P' of type τ'. 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 $A \in \mathit{STRUC}(\tau')$ has property P'. Removing the N predicate, the corresponding $A' \in \mathit{STRUC}(\tau)$ has property P and so T_ε^τ accepts with probability at least 2/3, as desired.

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

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

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

Testable properties remain testable when the vocabulary is extended. So it suffices to consider the minimal relevant vocabulary. A prefix class is *untestable* if it contains an untestable property. Simple modifications of the proof of Lemma 1 give the corresponding results for the variations considered in the next section.

3 Variations of Relational Property Testing

In Definition 3, any difference in low-arity relations is asymptotically dominated by the number of high-arity tuples. However, there are situations where this is not ideal. Consider (not necessarily admissible, vertex) 3-colored graphs with the vocabulary $\tau_C := (E^2, R^1, G^1, B^1)$. We might wish to test if the given coloring is admissible. In large graphs, this is equivalent to testing if the graph is 3-colorable and ignores the given coloring. We need a different model for our task.

Here we give two alternate definitions for the distance between structures. In testing we wish to distinguish structures that have a desired property and those that are far from the property, and so modifying the definition of distance changes the task of testing. As in Definition 3, the symbol $\oplus$ is exclusive-or.

Definition 8. *Let $A, B \in STRUC^n(\tau)$ be 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 3 gave equal weight to each *tuple* regardless of its arity, the above gives equal weight to each *relation.* However, loops in graphs and other *subtypes* of relations are similar to low-arity relations.

Definition 9. *Let R be a relation with arity a. The* subtypes *of R are the partitions of* $\{1, \dots, a\}$.

For example, $\{\{1\}, \{2\}\}$ is a subtype of the edge predicate E for graphs. This corresponds to the set of pairs of E for which the element in position 1 of the pair occurs only in position 1 and the element in position 2 occurs only in position 2. That is, this subtype is the set of edges that are not loops. The subtype $\{\{1, 2\}\}$ corresponds to the set of loops. This is more formally defined as follows.

Definition 10. *Let R be a relation with arity a and S be a subtype of R, i.e.,*

$$S = \left\{\{t_1^1, \dots, t_{b_1}^1\}, \dots, \{t_1^{|S|}, \dots, t_{b_{|S|}}^{|S|}\}\right\} \ .$$

Tuple $(x_1, \dots, x_a)$ belongs to S *if for all t_1^i, it is the case that $x_{t_1^i} = x_{t_j^i}$ for all j and, if $x_u = x_v$ for some u, v then u and v occur in the same element of S.*

We define the S-distance between structures, for a subtype S of relation R_i.

Definition 11. *Let $A, B \in STRUC^n(\tau)$ be structures with universe U, and let S be a subtype of relation $R_i \in \tau$. Then, the S-distance between A and B is*

$$S\text{-dist}(A, B) := \frac{|\{\mathbf{x} \mid \mathbf{x} \in U^{a_i}, \mathbf{x} \text{ belongs to } S \text{ and } R_i^A(\mathbf{x}) \oplus R_i^B(\mathbf{x})\}|}{n^{|S|}} \ .$$

If the S-dist for all subtypes of all relations is small, no query has a high probability of finding a difference. Denote the set of subtypes of R by $SUB(R)$.

Definition 12. *For $A, B \in STRUC^n(\tau)$, the* mrdist *is*

$$\mathrm{mrdist}(A, B) := \max_{1 \le i \le s} \max_{S \in SUB(R_i)} S\text{-}\mathrm{dist}(A, B) \ .$$

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. It is easy to show the following.

Theorem 1. *Let τ be a vocabulary and $A, B \in STRUC^n(\tau)$. Then,*

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

Assume a tester distinguishes between structures A having some property P and those for which $\mathrm{mrdist}(A, P) \ge \varepsilon$. Theorem 1 trivially implies that it also distinguishes between structures A that have P and those for which $\mathrm{rdist}(A, P) \ge \varepsilon$. 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, where $\overleftarrow{u}$ denotes the usual reversal of string u.

Theorem 2 (Alon *et al.* [4]). *The language $L = \{u\overleftarrow{u}v\overleftarrow{v}\}$, where u and v are strings over $\{0,1\}$ is not testable with complexity $o(\sqrt{n})$.*

In some vocabularies, e.g., binary strings and loop-free graphs, all three definitions are equivalent. However, we will show the following.

Theorem 3. $\mathcal{T}_{mr} \subset \mathcal{T}_r \subset \mathcal{T}$.

Proof. 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. 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. That is, A has P_1 if there is some $0 \le k \le n/2$ such that for all $0 \le i < k$, $S(i)$ is true iff $S(2k-1-i)$ is true and for all $0 \le j < (n-2k)/2$, $S(2k+j)$ 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 2. *Property P_1 is testable under the* dist *definition for distance.*

Proof. For any (even) n, 1^n is of the form $u\bar{u}v\bar{v}$. Given A, we create A' by changing all $S(i)$ assignments to be true. This involves at most n modifications and so $\mathrm{dist}(A, P_1) \leq \mathrm{dist}(A, A') = O(n)/\Theta(n^2) < \varepsilon$, 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 $\mathrm{dist}(A, P_1) \geq \varepsilon$, then $n < N(\varepsilon)$. In this case we query all assignments and reject with zero error. □ Lemma 2

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

Proof. Suppose there exist $\mathcal{T}_r$-type ε-testers T^ε for all $\varepsilon > 0$. We will show that the following is an ε-tester using *Definition* 3 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 $\mathrm{dist}(w, L) \geq \varepsilon$. Then, $\mathrm{rdist}(A, P_1) = \mathrm{dist}(w, L) \geq \varepsilon$ 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 3

Lemmata 2 and 3, 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.

An example is the property of graphs where the "loops" $E(i, i)$ encode the language from Theorem 2. We omit the details due to space. □ Theorem 3

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 4 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. Our main result is the testability of Ackermann's class with equality, which is of the form required by the lemma. A formula is *testable* if the property it defines is testable.

Lemma 4. *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 3, $\mathcal{T}_{mr}$ testability implies $\mathcal{T}_r$ testability. We prove $\mathcal{T}_r$ testability of such prefix classes implies $\mathcal{T}_{mr}$ testability using Lemma 5. In the following, $S(n,k)$ is the Stirling number of the second kind.

Lemma 5. *Let $\mathcal{C} = [\Pi, (p_1, p_2, \ldots)]_=$ be a prefix vocabulary class and let $q_j = \sum_{i \geq j} p_i S(i,j)$. If $\mathcal{C}' = [\Pi, (q_1, q_2, \ldots)]_=$ is $\mathcal{T}_r$ testable, then $\mathcal{C}$ is $\mathcal{T}_{mr}$ testable.*

Proof. Let $\varphi \in \mathcal{C}$ be arbitrary 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 $S(i,k)$ 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, we let t map the subtypes of a predicate to the sets of tuples comprising the subtypes. For our example of a binary predicate, $(0,1) \in t(\{\{1\},\{2\}\})$ and $(0,0) \in t(\{\{1,2\}\})$. 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

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

Note that $(x_1, \ldots, x_i) \in t(\mathcal{S})$ 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

$$([x = y \wedge E_1(x)] \vee [x \neq y \wedge E_2(x,y)]) .$$

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' 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 an A such that $\mathrm{mrdist}(A, P = \{B \mid B \models \varphi\}) \geq \varepsilon$, we simulate T^ε on an A' such that $\mathrm{rdist}(A', P' = \{B' \mid B' \models \varphi'\}) \geq \varepsilon$. The tester T^ε rejects with probability at least 2/3, as desired. □ Lemma 5

Proving $\mathcal{T}_r$ testability for $[\Pi, all]_=$ implies proving it for all $(q_1, \ldots)$ that are "images" of some $(p_1, \ldots)$ and so Lemma 5 is stronger than required. □ Lemma 4

4 The Classification Problem for Testability

Here we consider the classification problem of first-order logic for testability, inspired by the classification problem for decidability and results in testability such as those by Alon *et al.* [2]. The goal is a complete classification of the prefix vocabulary classes of first-order logic into testable and untestable classes.

We first outline the traditional classification problem, focusing on results with parallels to results in testing. See Börger *et al.* [9] for the complete classification and proofs. Then we prove the testability of Ackermann's class with equality.

4.1 Classification Similarities

Löwenheim [20] proved the decidability of monadic first-order logic, and McNaughton and Papert [21] showed that it (with ordering and some arithmetic) characterizes the star-free regular languages. The testability of this logic is then implied by a result of Alon *et al.* [4]. Using instead Büchi's [10] result that monadic *second*-order logic characterizes the regular languages, the parallel is with Skolem's [28] extension of Löwenheim's result to second-order logic.

Skolem [29] showed that $[\forall^*\exists^*, all]$ is a reduction class. Alon *et al.* [2] found an untestable graph property (an encoding of graph isomorphism) expressible in $[\forall^*\exists^*, (0,1)]_=$, a class close enough to Skolem's [29] to be interesting.

Alon *et al.* [2] also proved that $[\exists^*\forall^*, (0,1)]_=$ is testable. The class $[\exists^*\forall^*, all]_=$ is known as Ramsey's class and its decidability was shown by Ramsey [23].

[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. We define that $\mathbf{z} \notin Q$ where $\mathbf{z}$ is any tuple with at least one duplicate component and Q is any predicate symbol in φ'. The resulting map is injective but not necessarily surjective.

5 Ackermann's Class with Equality

Above, we saw several similarities between the classifications for decidability and testability. Here we give an additional example: $[\exists^*\forall\exists^*, all]_=$. Ackermann [1] proved the decidability of this class without equality. If we allow equality and a unary function symbol, the result is Shelah's class, which Shelah [27] proved decidable. Unlike the decidable classes above, Shelah's class does not have the finite model property and it would be interesting to determine if it is testable.

Kolaitis and Vardi [17] showed the satisfiability problem for Ackermann's class with equality is complete for NEXPTIME and that a 0-1 law holds for existential second-order logic where the first-order part belongs to $[\exists^*\forall\exists^*, all]_=$.

The main goal of this section is Theorem 4. Recalling Theorem 3, 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 colors. We define $\mathrm{Col}(A, c)$ to be the set of colors that occur at least c times in A.

Theorem 4. *All formulas in* $[\exists^*\forall\exists^*, all]_=$ *define properties that are in* $\mathcal{T}_{mr}$.

Proof. Ackermann's class with equality is $[\exists^*\forall\exists^*, all]_=$ and so it suffices to show the testability of property P of type $\tau = (R_1^{a_1}, \dots, R_s^{a_s})$ defined by formula $\varphi := \exists x_1 \dots \exists x_a \forall y \exists z_1 \dots \exists z_b : \psi$, where ψ is quantifier-free. 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 4 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.
3. Search over all $A' \in STRUC^\kappa(\tau)$. Accept if we find an A' such that $A' \models \varphi$ and the colors in our sample are a subset of $\mathrm{Col}(A', a+1)$.
4. Otherwise, reject.

We must show that the tester accepts if $A \models \varphi$ and rejects if $\mathrm{rdist}(A, P) > \varepsilon$, with probability at least 2/3 in both cases. We do this by showing that with probability at least 2/3, we get a "good" sample in step 2 (cf. Lemma 6). A "good" sample is one that contains all colors of A that occur on at least an $\varepsilon/(2 \cdot 2^{|M|})$ fraction of the elements of A and no colors that occur on at most a elements. We then show that the tester is correct if it obtains a good sample.

Lemma 6. *There is a constant k such that, with probability at least $2/3$, the tester obtains a sample that contains all colors that occur at least $\varepsilon n/(2 \cdot 2^{|M|})$ times in A and no colors that occur at most a times.*

Proof. 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 independent queries is at most $(1 - \varepsilon/(2 \cdot 2^{|M|}))^k$. There are at most $2^{|M|}$ such colors, and so the probability of our sample containing at least one representative of all such colors is at least

$$\left(1 - \left(1 - \frac{\varepsilon/2}{2^{|M|}}\right)^k\right)^{2^{|M|}} .$$

The $|M|$ is a constant, and we take k such that this probability is at least $\sqrt{2/3}$.

Next, the probability of a particular query seeing a fixed color that occurs at most a times is at most a/n and the probability that k independent queries miss it is at least $(1 - a/n)^k$. There are at most $2^{|M|}$ such colors, and so the probability that we miss all of them is at least $\left((1 - a/n)^k\right)^{2^{|M|}}$. The k and $|M|$ are now constant, and we let N be such that for $n > N$ this probability is at least $\sqrt{2/3}$.

The probability of a "good" sample is at least $\sqrt{2/3}^2 = 2/3$. □ Lemma 6

We now show that if $A \models \varphi$, the tester will accept if it obtains a good sample. We begin with Lemma 7.

Lemma 7. *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 structure 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 $w_1, \dots, w_b$ used for the j-th structure. We connect each $A_{i,j}$ to A_x in the same way as in A, modifying assignments on tuples $(A_x \cup A_{i,j})^{a_k}$.

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 $(x_1, \dots, x_a, y)$ occurs in A. We assume it is the k-th structure and use the elements of $A_{i+1 \mod 3,k}$ 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 $A_{i+1 \mod 3,k}$. The resulting structure, which we call A_1, is a model of φ. Before this step we have not modified any assignments "spanning" the "rows" $A_{i,j}$ 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[2]. 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 7

For structures A such that $\#(A) > N$, the colors in a good sample are a subset of $\mathrm{Col}(A, a+1)$. If $A \models \varphi$ and our tester obtains a good sample, then Lemma 7 implies that our tester will find an A' satisfying the conditions of step 3 and will therefore accept. The tester obtains a good sample with probability at least 2/3, and so the tester accepts such A with at least the same probability.

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. It is easiest 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 8, which we will use to "grow" smaller models.

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

Lemma 8. *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. 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 $(u_1, \ldots, u_a)$ 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 first modify the assignments needed to make the structure induced by $(x_1, \ldots, x_a, q)$ identical to that induced by $(x_1, \ldots, x_a, p)$. 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 $(v_1, \ldots, v_b)$ 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 $(q, v_1, \ldots, v_b)$ identical to that induced by $(p, v_1, \ldots, v_b)$[3]. 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. □ Lemma 8

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 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 8 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 occurring 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 8 to A, making $O(1)$ modifications to non-monadic assignments at each step. The resulting structure is B and is such that $B \models \varphi$.

We now show that $\mathrm{rdist}(A, B) < \varepsilon$. If R_i is a monadic relation, then the i-th term of the maximum in Definition 8 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 $o(1)$

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

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. □ Theorem 4

6 Conclusion

We considered a generalization of property testing which we call *relational property testing*. In Section 3 we showed the relationships between variations of our definitions. The "best" definition depends on the problem in question.

Relational databases are perhaps the most obvious example of massive structures where it would be promising to consider applications of property testing. Relational property testing is a natural way to characterize this problem. In addition, properties of databases are often given by queries written in formal languages such as SQL and so it is very natural to consider the testability of properties expressible in various syntactic restrictions of formal languages.

Finally, we used our framework to discuss the *classification problem for testability* in Section 4, inspired by the classical problem for decidability. The major result of Section 4 is the testability of Ackermann's class with equality, in each of the variations of relational property testing that we considered. This implies the corresponding result in the dense graph and hypergraph models.

References

[1] Ackermann, W.: Über die Erfüllbarkeit gewisser Zählausdrücke. Math. Annalen **100** (1928) 638–649
[2] Alon, N., Fischer, E., Krivelevich, M., Szegedy, M.: Efficient testing of large graphs. Combinatorica **20**(4) (2000) 451–476
[3] Alon, N., Fischer, E., Newman, I., Shapira, A.: A combinatorial characterization of the testable graph properties: It's all about regularity. In: STOC '06: Proc. 38th Ann. ACM Symp. on Theory of Comput., NY, USA, ACM (2006) 251–260
[4] Alon, N., Krivelevich, M., Newman, I., Szegedy, M.: Regular languages are testable with a constant number of queries. SIAM J. Comput. **30**(6) (2001) 1842–1862
[5] Alon, N., Shapira, A.: A characterization of the (natural) graph properties testable with one-sided error. In: Proc., 46th Ann. IEEE Symp. on Foundations of Comput. Sci., FOCS 2005, Washington, DC, USA, IEEE Comput. Soc. (2005) 429–438
[6] Alon, N., Shapira, A.: Homomorphisms in graph property testing. In Klazar, M., Kratochvíl, J., Loebl, M., Matoušek, J., Thomas, R., Valtr, P., eds.: Topics in Discrete Mathematics. Volume 26 of Algorithms and Combinatorics. Springer (2006) 281–313
[7] Alon, N., Shapira, A.: A separation theorem in property testing. Combinatorica **28**(3) (2008) 261–281
[8] Blum, M., Luby, M., Rubinfeld, R.: Self-testing/correcting with applications to numerical problems. J. of Comput. Syst. Sci. **47**(3) (1993) 549–595
[9] Börger, E., Grädel, E., Gurevich, Y.: The Classical Decision Problem. Springer-Verlag (1997)
[10] Büchi, J.R.: Weak second-order arithmetic and finite-automata. Z. Math. Logik Grundlagen Math. **6** (1960) 66–92

[11] Chockler, H., Kupferman, O.: ω-regular languages are testable with a constant number of queries. Theoret. Comput. Sci. **329**(1-3) (2004) 71–92
[12] Fischer, E.: The art of uninformed decisions. Bulletin of the European Association for Theoretical Computer Science **75** (October 2001) 97–126 Columns: Computational Complexity.
[13] Fischer, E., Matsliah, A., Shapira, A.: Approximate hypergraph partitioning and applications. Proc. 48th Ann. IEEE Symp. on Foundations of Comput. Sci., FOCS 2007 (2007) 579–589
[14] Freivalds, R.: Fast probabilistic algorithms. In: Mathematical Foundations of Computer Science 1979, Proc., 8th Symp., Olomouc, Czechoslovakia, September 3-7, 1979. Volume 74 of Lecture Notes in Computer Science., Springer-Verlag (1979) 57–69
[15] Goldreich, O., Ron, D.: Property testing in bounded degree graphs. Algorithmica **32** (2002) 302–343
[16] Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. J. ACM **45**(4) (1998) 653–750
[17] Kolaitis, P.G., Vardi, M.Y.: 0-1 laws and decision problems for fragments of second-order logic. Inf. Comput. **87**(1-2) (1990) 302–338
[18] Leeuw, K.d., Moore, E.F., Shannon, C.E., Shapiro, N.: Computability by probabilistic machines. In Shannon, C., McCarthy, J., eds.: Automata Studies. Princeton University Press, Princeton, NJ (1956) 183–212
[19] Lovász, L.: Some mathematics behind graph property testing. In Freund, Y., Györfi, L., Turán, G., Zeugmann, T., eds.: Algorithmic Learning Theory, 19th International Conference, ALT 2008, Budapest, Hungary, October 2008, Proc. Volume 5254 of Lecture Notes in Computer Science., Springer (2008) 3
[20] Löwenheim, L.: Über Möglichkeiten im Relativkalkül. Math. Annalen **76** (1915) 447–470
[21] McNaughton, R., Papert, S.: Counter-Free Automata. M.I.T. Press (1971)
[22] Parnas, M., Ron, D.: Testing the diameter of graphs. Random Struct. Algorithms **20**(2) (2002) 165–183
[23] Ramsey, F.P.: On a problem of formal logic. Proc. London Math. Soc. (2) **30** (1930) 264–286
[24] Rödl, V., Schacht, M.: Property testing in hypergraphs and the removal lemma. In: STOC '07: Proc. 39th Ann. ACM Symp. on Theory of Comput., NY, USA, ACM (2007) 488–495
[25] Ron, D.: Property testing. In Rajasekaran, S., Pardalos, P.M., Reif, J.H., Rolim, J., eds.: Handbook of Randomized Computing. Volume II. Kluwer Academic Publishers (2001) 597–649
[26] Rubinfeld, R., Sudan, M.: Robust characterizations of polynomials with applications to program testing. SIAM J. Comput. **25**(2) (1996) 252–271
[27] Shelah, S.: Decidability of a portion of the predicate calculus. Israel J. Math. **28**(1-2) (1977) 32–44
[28] Skolem, T.: Untersuchungen über die Axiome des Klassenkalküls und über Produktations und Summationsprobleme, welche gewisse Klassen von Aussagen betreffen. Videnskapsselskapets skrifter, I. Mat.-natur kl. (3) (1919) 37–71
[29] Skolem, T.: Logisch-kombinatorische Untersuchungen über die Erfüllbarkeit oder Beweisbarkeit mathematischer Sätze nebst einem Theorem über dichte Mengen. Videnskapsselskapets skrifter, I. Mat.-natur kl. (4) (1920) 1–26
[30] Straubing, H.: Finite Automata, Formal Logic, and Circuit Complexity. Birkhäuser (1994)