Innovations in Computer Science 2011

Property Testing via Set-Theoretic Operations

Victor Chen[1] Madhu Sudan[2*] Ning Xie[3†]
[1]Princeton University
[2] Microsoft Research New England
[3] MIT CSAIL
vychen@princeton.edu madhu@microsoft.com ningxie@csail.mit.edu

Abstract: Given two testable properties $\mathcal{P}_1$ and $\mathcal{P}_2$, under what conditions are the union, intersection or set-difference of these two properties also testable? We initiate a systematic study of these basic set-theoretic operations in the context of property testing. As an application, we give a conceptually different proof that linearity is testable, albeit with much worse query complexity. Furthermore, for the problem of testing disjunction of linear functions, which was previously known to be one-sided testable with a super-polynomial query complexity, we give an improved analysis and show it has query complexity $O(1/\epsilon^2)$, where ϵ is the distance parameter.

Keywords: property testing, self-correction, set theory.

1 Introduction

During the last two decades, the size of data sets has been increasing at an exponential rate, rendering a linear scan of the whole input an unaffordable luxury. Thus, we need *sublinear time* algorithms that read a vanishingly small fraction of their input and still output something intelligent and non-trivial about the properties of the input. The model of property testing [22,33] has been very useful in understanding the power of sublinear time. Property testing is concerned with the existence of a sublinear time algorithm that queries an input object a small number of times and decides correctly with high probability whether the object has a given property or whether it is "far away" from having the property.

We model input objects as strings of arbitrary length, which can also be viewed as a function on arbitrarily large domain. Formally, let $\mathcal{R}$ be a finite set and $\mathcal{D} = \{D_n\}_{n>0}$ be a parametrized family of domains. $\mathcal{R}^{\mathcal{D}}$ denote the set of all functions mapping from $\mathcal{D}$ to $\mathcal{R}$. A *property* $\mathcal{P}$ is simply specified by a family of functions $\mathcal{P} \subseteq \mathcal{R}^{\mathcal{D}}$. A *tester* for property $\mathcal{P}$ is a randomized algorithm which, given the oracle access to an input function $f \in \mathcal{R}^{\mathcal{D}}$ together with a distance parameter ϵ, distinguishes with high probability (say, 2/3) between the case that f satisfies $\mathcal{P}$ and the case that f is ϵ-*far* from satisfying $\mathcal{P}$. Here, distance between functions $f, g : \mathcal{D} \to \mathcal{R}$, denoted $\mathrm{dist}(f, g)$, is simply the probability that $\Pr_{x\in\mathcal{D}}[f(x) \neq g(x)]$, where x is chosen uniformly at random from $\mathcal{D}$, and $\mathrm{dist}(f, \mathcal{P}) = \min_{g\in\mathcal{P}}\{\mathrm{dist}(f, g)\}$. We say f is ϵ-*far* from $\mathcal{P}$ if $\mathrm{dist}(f, \mathcal{P}) \geqslant \epsilon$ and ϵ-*close* otherwise. The central parameter associated with a tester is the number of oracle queries it makes to the function f being tested.

Property testing was first studied by Blum, Luby and Rubinfeld [18] and was formally defined by Rubinfeld and Sudan [33]. The systematic exploration of property testing for combinatorial properties was initiated by Goldreich, Goldwasser, and Ron [22]. Subsequently, a rich collection of properties have been shown to be testable [3-5,7,8,19,25,26,31].

Perhaps the most fundamental question in property testing is the following: which properties have *local* testing algorithms whose running time depends only on the distance parameter ϵ? Are there any attributes that make a property locally testable? Questions of this type in the context of graph property testing were first raised in [22] and later received a lot of attention. Some very general results have been obtained [2,3,7,8,19.21], leading to an (almost) complete qualitative understanding of which graph prop-

* Research supported in part by NSF Award CCR-0514915.

† Research supported in part by NSF Award CCR-0728645.

erties are efficiently testable in the dense graph model (see [14] for some recent progress in the sparse graph model). In addition, for an important class of properties, namely H-freeness for fixed subgraphs H, it is known exactly for which H, testing H-freeness requires the query complexity to be super-polynomial in $1/\epsilon$ and for which only a polynomial number of queries suffice: This was shown by Alon [1] for one-sided error testers and by Alon and Shapira [6] for general two-sided error testers. Progress toward similar understanding has also been made for hypergraph properties [7,9,32].

However, much less is known for algebraic properties. In a systematic study, Kaufman and Sudan [27] examined the query complexity of a broad class of algebraic properties based on the invariance of these properties under linear transformations. Roughly speaking, they showed that any locally-characterized linear-invariant and *linear*[1] properties are testable with query complexities polynomial in $1/\epsilon$. Non-linear linear-invariant properties were first shown to be testable by Green [24] and were formally studied in [15]. The properties studied in [15,24] are "pattern-freeness" of Boolean functions, which has been attracting considerable attention [15,16,24,29,34], as such a study may lead to a complete characterization of testability for functions, analogous to the setting of graphs.

1.1 Motivation for set-theoretic operations

In this paper we propose a new paradigm to systematically study algebraic property testing. First, decompose a natural algebraic property into the union or intersection (or some other set operation) of a set of "atomic properties". Second, try to show that each of these atomic properties is testable. Finally, prove that some "composite" property obtained from applying some set theoretic operations on the (testable) atomic properties is also testable. A prominent example is the set of low-degree polynomials [4,25,26]. It is easy to see that the property of being a degree-d polynomial over GF(2) is simply the *intersection* of $2^{2^{d+1}-2}$ atomic properties. Indeed, let $\mathbb{P}_d$ denote the set of n-variate polynomials of degree at most d. Then, by the characterization of low-degree polynomials (see, e.g., [4]), $f \in \mathbb{P}_d$ if and only if for every $x_1, \ldots, x_{d+1} \in \mathbb{F}_2^n$,

$$\sum_{\emptyset \neq S \subseteq [d+1]} f(\sum_{i \in S} x_i) \equiv 0 \pmod 2.$$

Now fix an ordering of the non-trivial subsets of $[d+1] = \{1, 2, \ldots, d+1\}$. Let $\vec{b}$ be a bit-string of length $2^{2^{d+1}-1}$ with an odd number of ones and $\mathbb{P}_{d,\vec{b}}$ denote the set of functions f such that the string $\langle f(\sum_{i \in S} x_i) \rangle_{\emptyset \neq S \subseteq [d+1]}$ is not equal to $\vec{b}$. By definition, $\mathbb{P}_d$ is the intersection of $2^{2^{d+1}-2}$ "$\vec{b}$-free" properties $\mathbb{P}_{d,\vec{b}}$'s.[2]

In order to carry out this program of decomposing an algebraic properties into atomic ones, one must have a solid understanding of how basic set-theoretic operations affect testability. For instance, given two testable properties, is the union, intersection, or set-difference also testable? Previously, Goldreich, Goldwasser and Ron considered such questions in their seminal paper [22]. They observed that the union of two testable properties is always testable (cf. Section 3.1) but also provided examples showing that in general, testability is not closed under other set-theoretic operations. Thus, current understanding of testability via set-theoretic operations seems insufficient to carry out the above mentioned program of attack.

1.2 Our results

In this paper, we show more positive results for these basic set-theoretic operations and illustrate several applications. We now describe our contribution in more detail.

Set-theoretic operations We provide sufficient conditions that allow local testability to be closed under intersection and set difference. Given two locally testable properties, we show that if the two properties (minus their intersection) are sufficiently far apart, then their intersection is also locally testable. For set difference, a similar statement can also be made, albeit with more technicality, requiring that one of the properties must be "tolerantly testable".

A more detailed treatment of these set operations appears in Section 3. We remark that in the general case, testability is not closed under most set operations. Thus, putting restrictions on these properties is not unwarranted.

[1] A property $\mathcal{F}$ is linear if for any f and g that are in $\mathcal{F}$ necessarily implies that $f + g$ is in $\mathcal{F}$.

[2] In fact, some of these $2^{2^{d+1}-2}$ properties are identical since the set of non-trivial subsets generated by x_i is invariant under permutation of the x_i's.

Applications of these set-theoretic considerations appear in Sections 4.2 and 4.3. Furthermore, Section 4.3 demonstrates the simplicity that comes from these set-theoretic arguments. There, via set theory, we define a new property from an established one, and show that the new property's testability, in terms of both upper and lower bounds, is inherited from the previous property.

Disjunction of linear functions In addition to set theory, it is also natural to ask whether testability is preserved under the closure of some fundamental, unary operations. For instance, given a testable property $\mathcal{P}$, under what condition is its additive closure $\oplus\mathcal{P}$ testable? A similar question can also be asked for the disjunctive operator $\wedge$, which is one of the most basic operations used to combine formulas. Given a testable property $\mathcal{P}$, is its disjunctive closure $\wedge\mathcal{P}$ testable?

Trivially, if $\mathcal{P}$ is linear, then $\oplus\mathcal{P} = \mathcal{P}$ and testability is preserved. Furthermore, if $\mathcal{P}_1$ and $\mathcal{P}_2$ are both linear and linear-invariant as introduced by Kaufman and Sudan [27], then their sumset $\mathcal{P}_1 + \mathcal{P}_2$ is testable. However, in general, not much can be said about how these basic operations affect testability.

Here we focus on disjunction's effect on one specific property, namely the set of linear functions. Before we describe our result, we note some previous works in testing where disjunction played a role. For the disjunction of monomials, Parnas et. al. [31] gave a testing algorithm for s-term monotone DNF with query complexity $\tilde{O}(s^2/\epsilon)$. Diakonikolas et. al. [20] generalized Parnas et. al.'s result to general s-term DNF with query complexity $\tilde{O}(s^4/\epsilon^2)$.

We take a different direction and ask how disjunction affects the testability of the set of linear functions. The property of being a linear Boolean function (see next section for a full discussion), first studied by Blum, Luby and Rubinfeld [18], is testable with query complexity $O(1/\epsilon)$. As observed in [15], the class of disjunction of linear functions is equal to the class of 100-free functions (see Preliminaries for a definition). There they showed that a sufficiently rich class of "pattern-free" functions is testable, albeit with query complexity a tower of 2's whose height is a function of $1/\epsilon$. In a different context, the authors in [23] showed implicitly[3] that the disjunction of linear functions is testable with query complexity polynomial in $1/\epsilon$, but with two-sided error.

[3]We thank an anonymous reviewer from ICS 2011 for pointing this out.

Since both [15] and [23] seek to describe rich classes of testable Boolean functions, the bounds from both works do not adequately address how disjunction affects the query complexity of the underlying property, the set of linear functions. In Section 4.1, we give a direct proof, showing that the disjunction of linear functions is testable with query complexity $O(1/\epsilon^2)$ and has one-sided error. Thus, the blowup from the disjunctive operator is $O(1/\epsilon)$. It will be interesting to see if the blowup is optimal for this problem.

A different proof for linearity testing Linearity testing, first proposed by Blum, Luby and Rubinfeld [18], is arguably the most fundamental and extensively studied problem in property testing of Boolean functions. Due to its simplicity and important applications in PCP constructions, much effort has been devoted to the study of the testability of linearity [10-12,18,28].

For linearity, we indeed are able to carry out the program of decomposing an algebraic property into atomic pattern-free properties, and thus obtain a novel new proof that linearity is testable in Section 4.2. In particular, linearity is easily seen to be equal to the intersection of two atomic properties, namely triangle-freeness (see Section 2) and disjunction of linear functions, which are both testable.

The query complexity of linearity in our proof is of the tower-type, drastically worse than the optimal $O(1/\epsilon)$ bound, where ϵ is the distance parameter. We note that our effort in obtaining a new proof lies not in improving the parameters, but in understanding the relationships among these atomic, testable properties. In fact, we believe that despite the poor upper bound, our new proof is conceptually simple and gives evidence that set theory may uncover new testable properties.

1.3 Techniques

Our new proof that linearity is testable is built on the testability results for triangle freeness (see definition in Section 2) and the disjunction of linear functions. The latter was already shown to be testable in [15]. However, in this work, we give a completely different proof using a BLR-styled approach. Our proof is a novel variant of the classical self-correction method. Consequently, the query upper bound we obtain (quadratic in $1/\epsilon$) is significantly better than the tower-type upper bound shown in [15]. In fact, to the best of our knowledge, this is the first and only polynomial query upper bound for testing pattern-

freeness properties. All other analysis for testing pattern-freeness properties apply some type of "regularity lemma", thus making tower-type query upper bounds unavoidable.

We believe that both the self-correction technique and the investigation of set-operations may be useful in the study of testing pattern-freeness. From the works developed in [29,34], we know that for every d, the property $\mathbb{P}_{d,\vec{1}}$ is testable.[4] However, for an arbitrary $\vec{b}$, the testability of $\mathbb{P}_{d,\vec{b}}$ remains open. And in general very little can be said about the testability of an arbitrary intersection of these properties. Since $\mathbb{P}_d$ is known to be testable using self-correction [4], we believe that self-correction, applied in conjunction with set-theory, may be useful for understanding these pattern-free properties.

2 Preliminaries

We now describe some basic notation and definitions that we use throughout the paper. We let $\mathbb{N} = \{0, 1, \ldots\}$ denotes the set of natural numbers and $[n]$ the set $\{1, \ldots, n\}$. We view elements in $\mathbb{F}_2^n$ as n-bit binary strings, that is elements of $\{0,1\}^n$. For $x \in \mathbb{F}_2^n$, we write $x_i \in \{0,1\}$ for the i^{th} bit of x. If x and y are two n-bit strings, then $x + y$ denotes bitwise addition (i.e., XOR) of x and y, and $x \cdot y = \sum_{i=1}^{n} x_i y_i \pmod 2$ denotes the inner product between x and y. We write (x, y) to denote the concatenation of two bit strings x and y. For convenience, sometimes we view a n-bit binary string as a subset of $[n]$, that is, for every $x \in \mathbb{F}_2^n$ there is a corresponding subset $S_x \subseteq [n]$ such that $x_i = 1$ iff $i \in S_x$ for every $1 \leqslant i \leqslant n$. We write $|x|$ to indicate the Hamming weight of x, i.e., the number of coordinates i such that $x_i = 1$. Equivalently, this is also the cardinality of subset S_x. By abuse of notation, we use parentheses to denote *multisets*; for instance, we write (a, a, b, b, b) for the multiset which consists of two a's and three b's.

Let $f : \mathbb{F}_2^n \to \{0,1\}$ be a Boolean function. The *support* of f is $\text{supp}(f) = \{x \in \mathbb{F}_2^n : f(x) = 1\}$. Recall that for two functions f and g defined over the same domain, the (fractional) *distance* between these two functions is $\text{dist}(f,g) \stackrel{\text{def}}{=} \Pr_{x \in \mathcal{D}}[f(x) \neq g(x)]$. Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two properties defined over the same domain $\mathcal{D}$, then the *distance* between these two properties, $\text{dist}(\mathcal{P}_1, \mathcal{P}_2)$, is simply defined to be $\min_{f \in \mathcal{P}_1, g \in \mathcal{P}_2}\{\text{dist}(f,g)\}$.

A Boolean function $f : \mathbb{F}_2^n \to \{0,1\}$ is *linear* if for all x and y in $\mathbb{F}_2^n$, $f(x) + f(y) = f(x+y)$. We denote the set of linear function by $\mathcal{P}_{\mathbf{LIN}}$. Throughout this paper, we will be working with the pattern generated by the triple $(x, y, x+y)$. To this end, we say that a Boolean function $f : \mathbb{F}_2^n \to \{0,1\}$ is $(1,0,0)$-*free* if for all x and y in $\mathbb{F}_2^n$, $(f(x), f(y), f(x+y)) \neq (1,0,0)$, where here and after we view $(f(x), f(y), f(x+y))$ as well as $(1,0,0)$ as multisets[5]. We denote the set of $(1,0,0)$-free functions by $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$. Similarly, a $(1,1,0)$-*free* Boolean function is defined analogously. Lastly, we say that a Boolean function $f : \mathbb{F}_2^n \to \{0,1\}$ is *triangle-free* if for all x and y in $\mathbb{F}_2^n$, $(f(x), f(y), f(x+y)) \neq (1,1,1)$. We denote the set of triangle-free functions by $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$. Note that $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ is *monotone*: if $f \in \mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ and we modify f by setting some of the points in $\mathbb{F}_2^n$ from 1 to 0, then the new function is clearly also triangle-free. We encapsulate this observation into the following statement:

Observation 1. *Let f and g be two Boolean functions such that supp$(f) \subseteq$ supp(g). Then*

$$\textit{dist}(f, \mathcal{P}_{(111)\text{-}\mathbf{FREE}}) \leqslant \textit{dist}(g, \mathcal{P}_{(111)\text{-}\mathbf{FREE}}).$$

For concreteness, we provide a formal definition of a tester.

Definition 1(Testability). Let $\mathcal{R}$ be a finite set and $\mathcal{D} = \{D_n\}_{n>0}$ be a parametrized family of domains. Let $\mathcal{P} \subseteq \mathcal{R}^{\mathcal{D}}$ be a property.

We say a (randomized) algorithm T is a *tester* for $\mathcal{P}$ with *query complexity* $q(\epsilon, n)$ if for any distance parameter $\epsilon > 0$, input size n and function $f : D_n \to R$, T satisfies the following:

- T queries f at most $q(\epsilon, n)$ times;
- (completeness) if $f \in \mathcal{P}$, then $\Pr[T \text{ accepts}] = 1$;
- (soundness) if $\text{dist}(f, \mathcal{P}) \geqslant \epsilon$, then $\Pr[T \text{ accepts}] \leqslant \frac{1}{3}$, where the probabilities are taken over the internal randomness used by T.

We say that a property is *locally testable* if it has a tester whose query complexity is a function depending only on ϵ, independent of n. In this work, we actually use the word testability to describe the stronger notion of *local testability*. For our main results, we will work

[4] Actually, stronger theorems were proved in [29,34], but to state their works in full, definitions not needed in this work will have to be introduced.

[5] That is, for example, we do not distinguish the case $\langle f(x), f(y), f(x+y)\rangle = \langle 1,0,0\rangle$ from $\langle f(x), f(y), f(x+y)\rangle = \langle 0,1,0\rangle$.

with the model case when $\mathcal{D}_n = \mathbb{F}_2^n$ and $\mathcal{R} = \{0, 1\}$.

3 Basic theory of set operations

In this section, we present some basic testability results based on set-theoretic operations such as union, intersection, complementation, and set-difference. The proofs here are fairly standard and are thus deferred to the Appendix.

3.1 Union

It is well known that the union of two testable properties remains testable. This folklore result first appeared in [22]; for completeness, a proof is included in Appendix A.

Proposition 1(Folklore). *Let $\mathcal{P}_1, \mathcal{P}_2 \subseteq \mathcal{R}^{\mathcal{D}}$ be two properties defined over the same domain $D = \{D_n\}_{n>0}$. For $i = 1, 2$, suppose $\mathcal{P}_i$ is testable with query complexity $q_i(\epsilon)$. Then the union $\mathcal{P}_1 \cup \mathcal{P}_2$ is testable with query complexity $O(q_1(\epsilon) + q_2(\epsilon))$.*

3.2 Intersection

The case of set intersection is more complicated than union. Goldreich et al. showed in [22] (see Proposition 4.2.2) that there exist testable properties whose intersection is not testable. Thus, in general, testability does not follow from the intersection operation. However, testability may still follow in restricted cases. In particular, we show that if two testable properties $\mathcal{P}_1$ and $\mathcal{P}_2$ minus their intersection are sufficiently far from each other, then their intersection remains testable as well. A proof is included in Appendix B.

Proposition 2. *Let $\mathcal{P}_1, \mathcal{P}_2 \subseteq \mathcal{R}^{\mathcal{D}}$ be two properties defined over the same domain $\mathcal{D} = \{D_n\}_{n>0}$. Suppose $dist(\mathcal{P}_1 \setminus \mathcal{P}_2, \mathcal{P}_2 \setminus \mathcal{P}_1) \geqslant \epsilon_0$ for some absolute constant ϵ_0, and for $i = 1, 2$, $\mathcal{P}_i$ is testable with query complexity $q_i(\epsilon)$. Then the intersection $\mathcal{P}_1 \cap \mathcal{P}_2$ is testable with query complexity $O(q_1(\epsilon) + q_2(\epsilon))$,*

3.3 Complementation

Here we examine the effect complementation has on the testability of a property. As it turns out, all three outcomes – both $\mathcal{P}$ and $\bar{\mathcal{P}}$ are testable, only one of $\mathcal{P}$ and $\bar{\mathcal{P}}$ is testable, and neither $\mathcal{P}$ nor $\bar{\mathcal{P}}$ is testable – are possible!

The first outcome is the easiest to observe. Note that the property $\mathcal{D}^{\mathcal{R}}$ and the empty property are complements of each other, and both are trivially testable. The second outcome is observed in Proposition 4.2.3 in [22]. To our knowledge, the third outcome has not been considered before. In fact, previous constructions of non-testable properties, e.g. [13,22], are *sparse*. Hence, the complements of these non-testable properties are trivially testable (by the tester that accepts all input functions). One may wonder if in general the complement of a non-testable property must also be testable. We disprove this in the following proposition.

Proposition 3. *There exists some property $\mathcal{P} \subseteq \mathcal{R}^{\mathcal{D}}$ where $\mathcal{R} = \{0, 1\}$ and $\mathcal{D} = \{\mathbb{F}_2^n\}_{n>0}$, such that neither $\mathcal{P}$ nor $\overline{\mathcal{P}}$ is testable for any $\epsilon < 1/8$.*

By utilizing coding theory, we can bypass the sparsity condition to prove Proposition 3. Essentially, property $\mathcal{P}$ consists of neighborhoods around functions that have degree $n/2 - 1$ as polynomials over $\mathbb{F}_{2^n}$. Its complement contains functions that are polynomials of degree $n/2$. Since, for any $d \geqslant 0$, $d+1$ evaluations are needed to specify a polynomial of degree d, any tester for $\mathcal{P}$ or $\overline{\mathcal{P}}$ needs (roughly) at least $n/2$ queries. Using a standard argument involving code concatenation, one can construct $\mathcal{P}$ and $\overline{\mathcal{P}}$ to be *binary* properties that require testers of query complexity $\Omega(2^{n/2})$. A formal proof can be found in Appendix C. However, it is still open whether there exist properties $\mathcal{P}$ such that both $\mathcal{P}$ and $\overline{\mathcal{P}}$ require *linear* number of queries to test, like those hard-to-test properties constructed in [13,22].

3.4 Difference

Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two properties and let $\mathcal{P} = \mathcal{P}_1 \setminus \mathcal{P}_2$ denote the set difference of the two properties. In this section, we confine our attention to the simple case that $\mathcal{P}_2 \subset \mathcal{P}_1$. Since complementation is a special case of set-difference, from Section 3.3, we know that in general we can infer nothing about the testability of $\mathcal{P}$ from the fact that both $\mathcal{P}_1$ and $\mathcal{P}_2$ are testable. However, under certain restrictions, we still can show that $\mathcal{P}$ is testable.

First we observe a simple case in which $\mathcal{P}_1 \setminus \mathcal{P}_2$ is testable. This simple observation, which is obvious and whose proof we omit, is utilized in the proof of Theorem 4 in Section 4.3.

Observation 2. *Let $\mathcal{P}_2 \subset \mathcal{P}_1$ be two testable proper-*

ties defined over the same domain $\mathcal{D} = \{D_n\}_{n>0}$. If for every $f \in \mathcal{P}_2$, there is some $g \in \mathcal{P}_1 \setminus \mathcal{P}_2$ such that $dist(f, g) = o(1)$, then $\mathcal{P}_1 \setminus \mathcal{P}_2$ is testable by the same tester which tests property $\mathcal{P}_1$.

Our second observation on set difference relies on the notion of *tolerant testing*, introduced by Parnas, Ron, and Rubinfeld [30] to investigate testers that are guaranteed to accept (with high confidence) not only inputs that satisfy the property, but also inputs that are sufficiently close to satisfying it.

Definition 2(Tolerant Tester[30]) Let $0 < \epsilon_1 < \epsilon_2 < 1$ denote two distance parameters and $\mathcal{P} \subseteq \mathcal{R}^{\mathcal{D}}$ be a property defined over the domain $\mathcal{D} = \{D_n\}_{n>0}$. We say that property $\mathcal{P}$ is (ϵ_1, ϵ_2)-*tolerantly testable* with query complexity $q(\epsilon_1, \epsilon_2)$ if there is a tester T that makes at most $q(\epsilon_1, \epsilon_2)$ queries, if for all f with $\mathrm{dist}(f, \mathcal{P}) \leqslant \epsilon_1$, T rejects f with probability at most $1/3$, and for all f with $\mathrm{dist}(f, \mathcal{P}) \geqslant \epsilon_2$, T accepts f with probability at most $1/3$.

We record in the following proposition that if $\mathcal{P}$ and $\mathcal{P}_2$ are sufficiently far apart and $\mathcal{P}_2$ is tolerantly testable, then $\mathcal{P}$ is also testable. We include a proof in Appendix D.

Propostion 4. *Let $\epsilon_1 < \epsilon_2 < \epsilon_0$ be three absolute constants. Let $\mathcal{P}_2 \subset \mathcal{P}_1 \subseteq \mathcal{R}^{\mathcal{D}}$ be two properties defined over the same domain $\mathcal{D} = \{D_n\}_{n>0}$. If for every $\epsilon > 0$, P_1 is testable with query complexity $q_1(\epsilon)$, P_2 is (ϵ_1, ϵ_2)-tolerantly testable with query complexity $q_2(\epsilon_1, \epsilon_2)$, and $dist(\mathcal{P}_1 \setminus \mathcal{P}_2, \mathcal{P}_2) \geqslant \epsilon_0$, then $\mathcal{P}_1 \setminus \mathcal{P}_2$ is testable with query complexity $O(q_1(\epsilon) + q_2(\epsilon_1, \epsilon_2))$ (and completeness $2/3$).*

We note that since $\mathcal{P}_2$ is tolerantly testable, it does not have completeness 1. Thus, the set difference $\mathcal{P}_1 \setminus \mathcal{P}_2$ is not guaranteed to have one-sided error, either.

4 Main results

In this section we show two applications of the results developed in Section 3. We stress that set theoretic arguments may be used to show both upper bound results (some properties are testable with only a few number of queries) and lower bound results (some properties can not be tested by any tester with less than certain number of queries).

4.1 Testing disjunction of linear functions

In this section, we employ a BLR-style analysis to show that the class of disjunction of linear functions is testable with query complexity $O(1/\epsilon^2)$. We first recall from [15] that a function is a disjunction of linear functions iff it is $(1, 0, 0)$-free.

Propostion 5([5]). *A function $f : \mathbb{F}_2^n \to \{0, 1\}$ is (1,0,0)-free if and only if f is the disjunction (OR) of linear functions (or the all 1 function).*

Proof. The reverse direction is obvious. For the forward direction, let $S = \{x \in \mathbb{F}_2^n : f(x) = 0\}$. If S is empty, then f is the all 1 function. Otherwise let x and y be any two elements in S (not necessarily distinct). Then if f is $(1, 0, 0)$-free, it must be the case that $x+y$ is also in S. Thus S is a linear subspace of $\mathbb{F}_2^n$. Suppose the dimension of S is k with $k \geqslant 1$. Then there are k linearly independent vectors $a_1, \ldots, a_k \in \mathbb{F}_2^n$ such that $z \in S$ iff $\{z \cdot a_1 = 0\} \bigwedge \cdots \bigwedge \{z \cdot a_k = 0\}$. Therefore, by De Morgan's law, $f(z) = 1$ iff $z \in \bar{S}$ iff $\{z \cdot a_1 = 1\} \bigvee \cdots \bigvee \{z \cdot a_k = 1\}$, which is equivalent to the claim.

Recall that $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$ is the set of Boolean functions that are free of $(1, 0, 0)$-patterns for any x, y and $x+y$ in $\mathbb{F}_2^n$. $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$ was shown to be testable with a tower-type query upper bound in [15]. Observe that the testability of property $\mathcal{P}_{(110)\text{-}\mathbf{FREE}}$ is the same as the testability of $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$. We now give a direct proof that $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$ is testable.

Theorem 1. *For every distance parameter $\epsilon > 0$, the property $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$ is testable with query complexity $O(1/\epsilon^2)$.*

Proof. Suppose we have oracle access to some Boolean function $f : \mathbb{F}_2^n \to \{0, 1\}$. A natural 3-query test T for $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$ proceeds as follows. T picks x and y independently and uniformly at random from $\mathbb{F}_2^n$, and accepts iff $(f(x), f(y), f(x + y)) \neq (1, 0, 0)$.

Let $R \stackrel{\text{def}}{=} \Pr_{x,y}[(f(x), f(y), f(x + y)) \neq (1, 0, 0)]$ be the rejection probability of T. If $f \in \mathcal{P}_{(100)\text{-}\mathbf{FREE}}$, then $R = 0$, i.e., T has completeness 1. For soundness, in a series of steps, we shall show that for every $\epsilon > 0$, if $R < \epsilon^2/128$, then there exists a Boolean function g such that **(1)** g is well-defined, **(2)** $\mathrm{dist}(f, g) < \epsilon$, and **(3)** g is in $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$.

Let μ_0 denote $\Pr_x[f(x) = 0]$. Suppose $\mu_0 < 63\epsilon/64$. Then $\mathrm{dist}(f, \vec{1}) < 63\epsilon/64$, where $\vec{1}$ is the all-ones func-

tion. Then trivially, taking $g = \vec{1}$ completes the proof. Thus, henceforth we assume that $\mu_0 \geqslant 63\epsilon/64$.

For a fixed $x \in \mathbb{F}_2^n$, let p_{00}^x denote $\Pr_y[(f(y), f(x+y)) = (0,0)]$, and and p_{10}^x is defined similarly. We define $g : \mathbb{F}_2^n \to \{0,1\}$ as follows:

$$g(x) = \begin{cases} 0, & \text{if } p_{00}^x \geqslant \epsilon/4; \\ 1, & \text{if } p_{10}^x \geqslant \epsilon/4; \\ f(x), & \text{otherwise.} \end{cases}$$

Proof of (1). g is well-defined.

Suppose not, then there exists some $x \in \mathbb{F}_2^n$ such that $p_{00}^x, p_{10}^x \geqslant \epsilon/4$. Pick y and z independently and uniformly at random from $\mathbb{F}_2^n$. Let E be the event that

$$\text{at least one of } (f(y), f(z), f(y+z)) \text{ and } (f(x+y), f(x+z), f(y+z)) \text{ is } (1,0,0).$$

By assumption, with probability at least $\epsilon^2/16$, $f(y) = 1, f(x+y) = 0$ and $f(z), f(x+z) = 0$, which will imply that – regardless of the value of $f(y+z)$ – event E must occur. Thus, $\epsilon^2/16 \leqslant \Pr[E]$. On the other hand, by the union bound, $\Pr[E] \leqslant 2R < \epsilon^2/64$, a contradiction. □

Proof of (2). $\mathrm{dist}(f,g) < \frac{\epsilon}{32}$.

Suppose x is such that $f(x) \neq g(x)$. By construction, $\Pr_y[f(x), f(y), f(x+y)] \geqslant \epsilon/4$. This implies that the rejection probability R is at least $\mathrm{dist}(f,g) \cdot \epsilon/4$. Since $R < \epsilon^2/128$, $\mathrm{dist}(f,g) < \epsilon/32$. □

Before proving **(3)**, we first note that for every $x \in \mathbb{F}_2^n$,

$$\Pr_y[(g(x), g(y), g(x+y)) = (1,0,0)] < \frac{5\epsilon}{16}.$$

To see this, note that by construction of g, for every $x \in \mathbb{F}_2^n$, $\Pr_y[(g(x), f(y), f(x+y)) = (1,0,0)] < \epsilon/4$. Since $\mathrm{dist}(f,g) < \epsilon/32$, by the union bound, we can deduce that the probability that g has a $(1,0,0)$-pattern at $(x, y, x+y)$ is less than $\epsilon/4 + 2 \cdot \epsilon/32$.

Proof of (3). g is in $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$.

Suppose not, that there exist $x, y \in \mathbb{F}_2^n$ such that $g(x) = 1$, $g(y), g(x+y) = 0$. Pick z uniformly at random from $\mathbb{F}_2^n$. Let E denote the event that

- at least one of $(g(x), g(z), g(x+z)), (g(y), g(z), g(y+z))$,
- and $(g(x+y), g(x+z), g(y+z))$

is $(1,0,0)$.

A case by case analysis reveals that if $g(z) = 0$, then event E must occur. Note that the probability that $g(z) = 0$ is at least $63\epsilon/64 - \epsilon/32 = 61\epsilon/64$, since $f(z) = 0$ occurs with probability at least $63\epsilon/64$ and $\mathrm{dist}(f,g) < \epsilon/32$. On the other hand, by union bound, we have $\Pr[g(z) = 0] \leqslant \Pr[E] \leqslant 3 \cdot 5\epsilon/16$, implying that $61\epsilon/64 \leqslant 15\epsilon/16$, an absurdity.

Therefore, we have shown that on any input function that is ϵ-far from $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$, the rejection probability of T is always at least $\epsilon^2/128$. By repeating the basic test T independently $O(1/\epsilon^2)$ times, we can boost the rejection probability of T to $2/3$, and thus completing the proof. □

4.2 A new proof that linearity is testable

As an application of our results in Section 3.2, we give a new proof that linear functions are testable based on a set-theoretic argument. To this end, note that the set of linear functions equals to the intersection of $(1,1,1)$-free functions and $(1,0,0)$-free functions, i.e.,

$$\mathcal{P}_{\mathbf{LIN}} = \mathcal{P}_{(111)\text{-}\mathbf{FREE}} \cap \mathcal{P}_{(100)\text{-}\mathbf{FREE}}.$$

From the previous section, we know that $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$ is testable. The following theorem due to Green [24] asserts that $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ is also testable.

Theorem 2([24]). *The property $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ is testable with query complexity $W(poly(1/\epsilon))$, where for every $t > 0$, $W(t)$ denotes the tower of 2's of height $\lceil t \rceil$.*

By Proposition 2, to show that linearity is testable, it suffices to show that the two properties $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ and $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$ are essentially far apart. To this end, let us define a new property $\mathcal{P}_{\mathbf{NLTF}}$, where NLTF stands for *non-linear triangle-freeness*:

$$\mathcal{P}_{\mathbf{NLTF}} \stackrel{\text{def}}{=} \mathcal{P}_{(111)\text{-}\mathbf{FREE}} \setminus \mathcal{P}_{\mathbf{LIN}}.$$

Lemma 1. *We have that $\mathcal{P}_{(100)\text{-}\mathbf{FREE}} \setminus \mathcal{P}_{\mathbf{LIN}}$ is $\frac{1}{4}$-far from $\mathcal{P}_{\mathbf{NLTF}}$.*

We first establish a weaker version of Lemma 1.

Propostion 6. *Suppose f is a disjunction of*

exactly two non-trivial linear functions. Then $dist(f, \mathcal{P}_{(111)\text{-}\mathbf{FREE}})$ *is at least* $\frac{1}{4}$.

Proof. Set $N = 2^n$. Write $f(x) = (\alpha \cdot x) \bigvee (\beta \cdot x)$, where $\alpha \neq \beta \in \mathbb{F}_2^n$ denote two n-bit vectors not equal to 0^n. We say that a tuple $(x, y, x+y)$ where $x, y \in \mathbb{F}_2^n$ is a triangle in f if $f(x), f(y), f(x+y) = 1$. We shall show that **(1)** f has $N^2/16$ triangles and **(2)** for every x, the number of $y's$ such that $(x, y, x+y)$ is a triangle in f is $N/4$. Together, **(1)** and **(2)** will imply that $\text{dist}(f, \mathcal{P}_{(111)\text{-}\mathbf{FREE}})$ is at least $1/4$, since changing the value of f at one point removes at most $N/4$ triangles.

To prove these two assertions, let $A = \{x \in \mathbb{F}_2^n : \alpha \cdot x = 1\}$ and $B = \{x \in \mathbb{F}_2^n : \beta \cdot x = 1\}$. Since $\text{supp}(f) = A \cup B$, for every triangle $(x, y, x+y)$ in f, each of the three points $x, y, x+y$ must fall in one of the following three disjoint sets:

$$A \setminus B, (A \cap B), B \setminus A.$$

Furthermore, each of the three points must fall into distinct sets. To see this, suppose that $x, y \in A \setminus B$. Then by definition, $\alpha(x+y) = \alpha(x) + \alpha(y) = 0$ and $\beta(x+y) = 0$, implying that $f(x+y) = 0$, a contradiction. So $A \setminus B$ cannot contain two points of a triangle, and by symmetry, neither can $B \setminus A$. The same calculation also reveals that $A \cap B$ cannot contain two points of a triangle.

Thus, a triangle $(x, y, x+y)$ in f must be such that $x \in A \setminus B$, $y \in A \cap B$, and $x+y \in B \setminus A$. In addition, it is easy to check that given two points p_1, p_2 from two distinct sets (say $A \setminus B$ and $A \cap B$), their sum $p_1 + p_2$ must be in the third set $(B \setminus A)$. Since these three sets $A \setminus B, (A \cap B), B \setminus A$ all have size $N/4$, this implies that the number of triangles in f is $N^2/16$, proving **(1)**.

(2) also follows easily given the above observations. Suppose $x \in A \setminus B$. For every $y \in A \cap B$, $(x, y, x+y)$ forms a triangle. Since any triangle that has x as a point must also contain a point in $A \cap B$ (with the third point uniquely determined by the first two), the number of triangles in f containing x is $N/4$. The case when $x \in B \setminus A$ or $x \in A \cap B$ is similar. This completes the proof. □

Now we prove Lemma 1.

Proof of Lemma 1. Let $f \in \mathcal{P}_{(100)\text{-}\mathbf{FREE}} \setminus \mathcal{P}_{\mathbf{LIN}}$ and write $f = f_1 \vee f_2$, where f_1 is a disjunction of exactly two linear functions. By Proposition 6, it follows that $\text{dist}(f_1, \mathcal{P}_{(111)\text{-}\mathbf{FREE}})$ is at least $1/4$. Since $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ is monotone and $\text{supp}(f_1) \subseteq \text{supp}(f)$, by Observation 1, we know that $\text{dist}(f, \mathcal{P}_{(111)\text{-}\mathbf{FREE}}) \geqslant 1/4$. Since $\mathcal{P}_{\mathbf{NLTF}} \subset \mathcal{P}_{(111)\text{-}\mathbf{FREE}}$, $\text{dist}(f, \mathcal{P}_{\mathbf{NLTF}}) \geqslant \text{dist}(f, \mathcal{P}_{(111)\text{-}\mathbf{FREE}})$, completing the proof. □

By Theorem 2 and Theorem 1, both $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ and $\mathcal{P}_{(100)\text{-}\mathbf{FREE}}$ are testable. Now by combining Proposition 2 and Lemma 1, we obtain the following:

Theorem 3. $\mathcal{P}_{\mathbf{LIN}}$ *is testable.*

We remark that the query complexity for testing linearity in Theorem 3 is of the tower type (of the form $W(\text{poly}(1/\epsilon))$ because of Theorem 2. This is much worse than the optimal linear query upper bound obtained in [10,18].

4.3 A lower bound for testing non-linear triangle-freeness

We first show that $\mathcal{P}_{\mathbf{LIN}}$ is a "thin strip" around $\mathcal{P}_{\mathbf{NLTF}}$.

Propostion 7. *For any Boolean function* f, $dist(f, \mathcal{P}_{(111)\text{-}\mathbf{FREE}}) \geqslant dist(f, \mathcal{P}_{\mathbf{NLTF}}) - 2^{-n}$.

Proof. The statement is trivially true if $\text{dist}(f, \mathcal{P}_{(111)\text{-}\mathbf{FREE}}) = \text{dist}(f, \mathcal{P}_{\mathbf{NLTF}})$. Since $\mathcal{P}_{\mathbf{NLTF}}$ is a proper subset of $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$, we can assume that $\text{dist}(f, \mathcal{P}_{\mathbf{NLTF}})$ is strictly larger than $\text{dist}(f, \mathcal{P}_{(111)\text{-}\mathbf{FREE}})$, implying that the function in $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ that has minimum distance to f is actually in $\mathcal{P}_{\mathbf{LIN}}$. Call this function g. Then it is easy to see that there exists some function h in $\mathcal{P}_{\mathbf{NLTF}}$ such that $\text{dist}(g, h) = 2^{-n}$. To this end, note that if g is the all-zero function, we can define h such that $h(x) = 1$ for some $x \neq 0^n$ and 0 everywhere else. By construction h is non-linear but triangle-free. If g is a non-trivial linear function, then we can pick any $x \in \text{supp}(g)$ and define $h(x) = 0$ and $h(y) = g(y)$ for all $y \neq x$. By construction h is non-linear, and since $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ is monotone, h remains triangle-free.

Thus, by Triangle inequality, we know that $\text{dist}(f, \mathcal{P}_{(111)\text{-}\mathbf{FREE}}) = \text{dist}(f, g)$ is at least $\text{dist}(f, h) - 2^{-n}$. This implies that $\text{dist}(f, \mathcal{P}_{(111)\text{-}\mathbf{FREE}}) \geqslant \text{dist}(f, \mathcal{P}_{\mathbf{NLTF}}) - 2^{-n}$. □

Since any linear function is 2^{-n}-close to a function in $\mathcal{P}_{\mathbf{NLTF}}$, intuitively we expect $\mathcal{P}_{\mathbf{NLTF}}$, which is obtained by deleting the strip $\mathcal{P}_{\mathbf{LIN}}$ from $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$, to inherit the testability features of $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$. Indeed, we record this next by using the set-theoretic machinery set up in Section 3.

Theorem 4. *$\mathcal{P}_{\mathbf{NLTF}}$ is testable, but any non-adaptive* [6] *tester (with one-sided error) for $\mathcal{P}_{\mathbf{NLTF}}$ requires $\omega(1/\epsilon)$ queries.*

Proof. We first observe that $\mathcal{P}_{\mathbf{NLTF}}$ is testable with one-sided error. By Proposition 7 and Observation 2, the testing algorithm for $\mathcal{P}_{\mathbf{NLTF}}$ is simply the same as the tester for $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ [24].

Next we show that the lower bound for the query complexity of $\mathcal{P}_{\mathbf{NLTF}}$ is the same as $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$. As shown in [17], any one-sided, non-adaptive tester for $\mathcal{P}_{(111)\text{-}\mathbf{FREE}}$ requires $\omega(1/\epsilon)$ queries.[7] Suppose $\mathcal{P}_{\mathbf{NLTF}}$ is testable with one-sided error and has query complexity $O(1/\epsilon)$. Since $\mathcal{P}_{\mathbf{LIN}}$ is testable with query complexity $O(1/\epsilon)$ [18], by Proposition 1 $\mathcal{P}_{(111)\text{-}\mathbf{FREE}} = \mathcal{P}_{\mathbf{LIN}} \cup \mathcal{P}_{\mathbf{NLTF}}$ is testable with one-sided error and has query complexity $O(1/\epsilon)$, a contradiction. □

5 Concluding remarks

We have initiated a general study of the closure of testability under various set operations. Our results show that such a study can lead to both upper and lower bound results in property testing. We believe our answers are far from complete, and further investigation may lead to more interesting results. For example, the symmetric difference between two properties $\mathcal{P}_1$ and $\mathcal{P}_2$ is defined to be $\mathcal{P}_1 \triangle \mathcal{P}_2 \stackrel{\text{def}}{=} (\mathcal{P}_1 \setminus \mathcal{P}_2) \cup (\mathcal{P}_2 \setminus \mathcal{P}_1)$. Under what conditions is the property $\mathcal{P}_1 \triangle \mathcal{P}_2$ testable if both $\mathcal{P}_1$ and $\mathcal{P}_2$ are testable? Another natural generalization of our approach is to examine properties resulting from a finitely many application of some set-theoretic operations.

Our proof that the class of disjunction of linear functions is testable employs a BLR-style self-correction approach. We believe that this technique may be useful in analyzing other non-monotone, pattern-free properties. In particular, it will be interesting to carry out our approach of decomposing an algebraic property into atomic ones for higher degree polynomials. This will, in addition to giving a set-theoretic proof for testing low-degree polynomials, sheds light on how pattern-free properties relate to one another.

Finally, our quadratic query complexity upper bound for the disjunction of linear functions opens up a number of directions. In our work, the blowup in query complexity from the disjunction is $O(1/\epsilon)$. One may vary the underlying properties and the operators to measure the blowup in query complexity. Of particular interest may be understanding how the disjunction affects the testability of low-degree polynomials.

[6] A tester is non-adaptive if all its query points can be determined before the execution of the algorithm, i.e., the locations where a tester queries do not depend on the answers to previous queries.

[7] The specific lower bound shown in [17] is $\Omega((\frac{1}{\epsilon})^{1.704\cdots})$ but can be improved to be $\Omega((\frac{1}{\epsilon})^{2.423\cdots})$ as observed independently by Eli Ben-Sasson and the third author of the present paper.

Acknowledgments

We thank the anonymous referees for numerous suggestions and the reference to [23].

References

[1] Noga Alon. Testing subgraphs in large graphs. *Random Structures and Algorithms*, 21(3-4):359–370, 2002.

[2] Noga Alon, Eldar Fischer, Michael Krivelevich, and Mario Szegedy. Efficient testing of large graphs. *Combinatorica*, 20(6):451–476, 2000.

[3] 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, 2006.

[4] Noga Alon, Tali Kaufman, Michael Krivelevich, Simon Litsyn, and Dana Ron. Testing low-degree polynomials over GF(2). In *Proceedings of Random 2003*, pages 188–199, 2003.

[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, 2000.

[6] Noga Alon and Asaf Shapira. Testing subgraphs in directed graphs. *Journal of Computer and System Sciences*, 69(3):354–382, 2004.

[7] Noga Alon and Asaf Shapira. A characterization of the (natural) graph properties testable with one-sided error. In *FOCS'05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 429–438, 2005.

[8] Noga Alon and Asaf Shapira. Every monotone graph property is testable. In *STOC'05: Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 128–137, 2005.

[9] Tim Austin and Terence Tao. On the testability and repair of hereditary hypergraph properties. http://arxiv.org/abs/0801.2179, 2008.

[10] Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos A.Kiwi,and Madhu Sudan. Linearity testing over characteristic two. *IEEE Transactions on Information Theory*,42(6)1781–1795, 1996.

[11] Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, PCPs, and nonapproximability—towards tight results. *SIAM Journal on Computing*, 27(3):804–915, 1998.

[12] Mihir Bellare, Shafi Goldwasser, Carsten Lund, and Alexander Russell. Efficient probabilistically checkable proofs and applications to approximation. In *STOC'93: Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, pages 304–294, 1993.

[13] Eli Ben-Sasson, Prahladh Harsha, and Sofya Raskhodnikova. Some 3CNF properties are hard to test. *SIAM Journal on Computing*, 35(1):1–21, 2005. Early version in STOC'03.

[14] Itai Benjamini, Oded Schramm, and Asaf Shapira. Every minor-closed property of sparse graphs is testable. In *STOC'08: Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 393–402, 2008.

[15] Arnab Bhattacharyya, Victor Chen, Madhu Sudan, and Ning Xie. Testing linear-invariant nonlinear properties. In *STACS'09*, pages 135–146, 2009.

[16] Arnab Bhattacharyya, Elena Grigorescu, and Asaf Shapira. A unified framework for testing linear-invariant properties. In *FOCS'10: Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, 2010.

[17] Arnab Bhattacharyya and Ning Xie. Lower bounds for testing triangle-freeness in Boolean functions. In *SODA'10: Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 87–98, 2010.

[18] 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.

[19] Christian Borgs, Jennifer T. Chayes, László Lovász, Vera T. Sós, Balázs Szegedy, and Katalin Vesztergombi. Graph limits and parameter testing. In *STOC'06: Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 261–270, 2006.

[20] Ilias Diakonikolas, Homin K. Lee, Kevin Matulef, Krzysztof Onak, Ronitt Rubinfeld, Rocco Servedio, and Andrew Wan. Testing for concise representations. In *FOCS'07: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 549–558, 2007.

[21] Eldar Fischer and Ilan Newman. Testing versus estimation of graph properties. *SIAM Journal on Computing*, 37(2):482–501, 2007.

[22] 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.

[23] Parikshit Gopalan, Ryan O'Donnell, Rocco A. Servedio, Amir Shpilka, and Karl Wimmer. Testing Fourier dimensionality and sparsity. In *ICALP (1)*, pages 500–512, 2009.

[24] Ben Green. A Szemerédi-type regularity lemma in abelian groups, with applications. *Geom. Funct. Anal.*, 15(2):340–376, 2005.

[25] Charanjit S. Jutla, Anindya C. Patthak, Atri Rudra, and David Zuckerman. Testing low-degree polynomials over prime fields. In *FOCS'04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 423–432, 2004.

[26] Tali Kaufman and Dana Ron. Testing polynomials over general fields. In *FOCS'04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 413–422, 2004.

[27] Tali Kaufman and Madhu Sudan. Algebraic property testing: The role of invariance. In *STOC'08: Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 403–412, 2008.

[28] Marcos Kiwi. Algebraic testing and weight distributions of codes. *Theoretical Computer Science*, 299(1-3):81–106, 2003. Earlier version appeared as ECCC TR97-010, 1997.

[29] Dan Král, Oriol Serra, and Lluis Vena. A removal lemma for systems of linear equations over finite fields, 2008.

[30] Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012–1042, 2006.

[31] Michal Parnas, Dana Ron, and Alex Samorodnitsky. Testing basic Boolean formulae. *SIAM Journal on Discrete Mathematics*, 16(1):20–46, 2003.

[32] Vojtěch Rödl and Mathias Schacht. Generalizations of the removal lemma. *Combinatorica*, To appear. Earlier version in STOC'07.

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

[34] Asaf Shapira. Green's conjecture and testing linear-invariant properties. In *STOC'09: Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, pages 159–166, 2009.

A Proof of Proposition 1

Let T_1 be the tester for $\mathcal{P}_1$ with query complexity $q_1(\epsilon, n)$ and T_2 be the tester for $\mathcal{P}_2$ with query complexity $q_2(\epsilon, n)$. We may assume that both T_1 and T_2 have soundness $1/6$ with a constant blowup in their query complexity. Define T to be the tester which, on input function f, first simulates T_1 and then T_2. If at least one of the two testers T_1 and T_2 accepts f, T accepts f. Otherwise, T rejects.

Clearly the query complexity of T is $O(q_1+q_2)$. For completeness, note that if f is in $\mathcal{P}$, then by definition f is in at least one of $\mathcal{P}_1$ and $\mathcal{P}_2$. Thus, T accepts f with probability 1. Now suppose $\mathrm{dist}(f, \mathcal{P}) \geqslant \epsilon$. Then we have both $\mathrm{dist}(f, \mathcal{P}_1) \geqslant \epsilon$ and $\mathrm{dist}(f, \mathcal{P}_2) \geqslant \epsilon$. By the union bound, the probability that at least one of T_1 and T_2 accepts f is at most $1/6 + 1/6 = 1/3$. □

B Proof of Proposition 2

Let T_1 be the tester for $\mathcal{P}_1$ with query complexity $q_1(\epsilon)$, and T_2 be the tester for $\mathcal{P}_2$ with query complexities $q_2(\epsilon)$. First we convert T_1 into another tester T_1' for $\mathcal{P}_1$ such that, on input distance parameter ϵ, T_1' makes $Q_1'(\epsilon)$ queries, where

$$Q_1'(x) = \begin{cases} q_1(x), & \text{if } x < \dfrac{\epsilon_0}{2}; \\ \max\{q_1(x), q_i(\dfrac{\epsilon_0}{2})\}, & \text{otherwise.} \end{cases}$$

In other words, T_1' can be obtained from T_1 by making more queries when x is larger than $\epsilon_0/2$. Similarly, we can construct T_2' from T_2 in the same manner. Since ϵ_0 is a constant, we have $Q_1'(\epsilon) = O(q_1(\epsilon))$ and $Q_2'(\epsilon) = O(q_2(\epsilon))$.

Define T to be the tester that on input function f, first simulates T_1' and then T_2'. If both testers T_1' and T_2' accept, then T accepts f. Otherwise, it rejects. The query complexity of T is $Q_1'(\epsilon) + Q_2'(\epsilon)$, which is $O(q_1(\epsilon) + q_2(\epsilon))$.

For the completeness, if $f \in \mathcal{P}$, then both $f \in \mathcal{P}_1$ and $f \in \mathcal{P}_2$ hold. Therefore, T accepts with probability at least 1. For the soundness, suppose $\mathrm{dist}(f, \mathcal{P}) \geqslant \epsilon$. We distinguish between two cases.

Case 1. $\epsilon \leqslant \frac{\epsilon_0}{2}$.

It suffices to show that f is ϵ-far from at least one of $\mathcal{P}_1$ or $\mathcal{P}_2$. This fact then implies that T, in simulating T' and T_2', accepts f with probability at most $1/3$.

To show the f is far from at least one of the two properties, suppose not, that we have both $\mathrm{dist}(f, \mathcal{P}_1) < \epsilon$ and $\mathrm{dist}(f, \mathcal{P}_2) < \epsilon$. That is, there exist $g_1 \in \mathcal{P}_1$ and $g_2 \in \mathcal{P}_2$ such that $\mathrm{dist}(f, g_1) < \epsilon$ and $\mathrm{dist}(f, g_2) < \epsilon$.

Since $\mathrm{dist}(f, \mathcal{P}) \geqslant \epsilon$, $g_1, g_2 \notin \mathcal{P}$ and therefore $g_1 \in \mathcal{P}_1 \setminus \mathcal{P}$ and $g_2 \in \mathcal{P}_2 \setminus \mathcal{P}$. By triangle inequality, $\mathrm{dist}(g_1, g_2) < 2\epsilon \leqslant \epsilon_0$, and consequently $\mathrm{dist}(\mathcal{P}_1 \setminus \mathcal{P}_2, \mathcal{P}_2 \setminus \mathcal{P}_1) < \epsilon_0$, contradicting our assumption.

Case 2. $\epsilon > \frac{\epsilon_0}{2}$.

There are three sub-cases depending on where f is located. We analyze each of them separately below. Note that in each of the sub-cases, f is at least $\epsilon_0/2$-far from one of $\mathcal{P}_1$ and $\mathcal{P}_2$.

1. $f \in \mathcal{P}_1 \setminus \mathcal{P}$. Then by our assumption on the distance between $\mathcal{P}_1 \setminus \mathcal{P}_2$ and $\mathcal{P}_2 \setminus \mathcal{P}_1$, $\mathrm{dist}(f, \mathcal{P}_2 \setminus \mathcal{P}) \geqslant \epsilon_0$. It follows that
$$\begin{aligned} \mathrm{dist}(f, \mathcal{P}_2) &= \min\{\mathrm{dist}(f, \mathcal{P}), \mathrm{dist}(f, \mathcal{P}_2 \setminus \mathcal{P})\} \\ &\geqslant \min\{\epsilon, \epsilon_0\} \\ &\geqslant \epsilon_0/2. \end{aligned}$$
2. $f \in \mathcal{P}_2 \setminus \mathcal{P}$. Analogous to the case above, we have $\mathrm{dist}(f, \mathcal{P}_1) \geqslant \epsilon_0/2$.
3. $f \notin \mathcal{P}_1 \cup \mathcal{P}_2$. Then by triangle inequality, $\max\{\mathrm{dist}(f, \mathcal{P}_1 \setminus \mathcal{P}), \mathrm{dist}(f, \mathcal{P}_2 \setminus \mathcal{P})\} \geqslant \epsilon_0/2$. So there is some $i \in \{1, 2\}$ such that $\mathrm{dist}(f, \mathcal{P}_i \setminus \mathcal{P}) \geqslant \epsilon_0/2$. Since $\mathrm{dist}(f, \mathcal{P}) \geqslant \epsilon$, it follows that

$$\mathrm{dist}(f, \mathcal{P}_i) \geqslant \min\{\epsilon, \epsilon_0/2\} = \epsilon_0/2.$$

Thus, we conclude that there is some $i \in \{1, 2\}$ such that $\mathrm{dist}(f, \mathcal{P}_i) \geqslant \epsilon_0/2$. This implies that T'_i, which makes at least $Q'_i(\epsilon) \geqslant q_i(\epsilon_0/2)$ queries, accepts f with probability at most $1/3$. Hence, T accepts f with probability at most $1/3$ as well, completing the proof. □

C Proof of Proposition 3

We shall define a property $\mathcal{P} = \{P_{2k}\}_{k>0}$, where $\mathcal{P}_{2k} \subseteq \{0,1\}^{\mathbb{F}_2^{2k}}$ is a collection of Boolean functions defined over $\mathbb{F}_2^{2k}$, such that neither P_{2k} nor $\overline{P_{2k}}$ is testable. Recall that a property $\mathcal{P}$ is said to be *testable* if there is a tester for $\mathcal{P}$ whose query complexity is independent of the sizes of the inputs to the functions (in our case, independent of k).

First, let the Hadamard encoding $\mathrm{Had} : \mathbb{F}_2^k \times \mathbb{F}_2^k \to \{0,1\}$ be $\mathrm{Had}(\alpha, x) = \alpha \cdot x$. Note that $\mathbb{F}_{2^k}$ is isomorphic to $\mathbb{F}_2^k$, so for every function $g : \mathbb{F}_{2^k} \to \mathbb{F}_{2^k}$, the Hadamard concatenation of g can be written as $\mathrm{Had} \circ g : \mathbb{F}_2^{2k} \to \{0,1\}$ where $(\mathrm{Had} \circ g)(x, y) \stackrel{\mathrm{def}}{=} \mathrm{Had}(g(x), y)$.

We now define P_{2k} as follows. Let $f \in P_{2k}$ if there exists a polynomial $p : \mathbb{F}_{2^k} \to \mathbb{F}_{2^k}$ of degree at most $2^{k-1} - 1$ such that $\mathrm{dist}(f, \mathrm{Had} \circ p) < 1/8$. An important fact is that if $g : \mathbb{F}_{2^k} \to \mathbb{F}_{2^k}$ is a polynomial of degree 2^{k-1}, then $\mathrm{Had} \circ g$ is not in P_{2k}. To see this, note that by the Schwartz-Zippel Lemma, if $q : \mathbb{F}_{2^k} \to \mathbb{F}_{2^k}$ is a polynomial of degree at most 2^{k-1}, then $\Pr_x[q(x) = 0] \leqslant 1/2$. Therefore, for any polynomial p of degree at most $2^{k-1} - 1$, $\mathrm{dist}(p, g) \geqslant 1/2$. This implies that $\mathrm{dist}(\mathrm{Had} \circ p, \mathrm{Had} \circ g) \geqslant 1/4$, since the Hadamard encoding has relative distance $1/2$.[8] Because the Hadamard encoding of g is at least $1/4$-far from the Hadamard encoding of any degree $2^{k-1} - 1$ polynomials, by construction of P_{2k}, it follows that we not only have $\mathrm{Had} \circ g \in \overline{P_{2k}}$, but also have $\mathrm{dist}(\mathrm{Had} \circ g, P_{2k}) > 1/8$. Similarly, for any polynomial $h : \mathbb{F}_{2^k} \to \mathbb{F}_{2^k}$ of degree at most $2^{k-1} - 1$, $\mathrm{dist}(\mathrm{Had} \circ h, \overline{P_{2k}}) > 1/8$.

Now we show that neither P_{2k} nor its complement is testable for any distance parameter $\epsilon < 1/8$. By polynomial interpolation, for every set of 2^{k-1} points, there exists a polynomial h of degree $2^{k-1} - 1$ that agrees with these points. So any tester that distinguishes between members of P_{2k} and members that are at least ϵ-far away from P_{2k} needs at least $2^{k-1} + 1$ queries. Similarly, as we have just shown that $\mathrm{Had} \circ g \in \overline{P_{2k}}$ when g is a degree-2^{k-1} polynomial, it follows that any tester that distinguishes between members of $\overline{P_{2k}}$ and functions that are at least ϵ-far away from $\overline{P_{2k}}$ also needs at least $2^{k-1} + 2$ queries. To conclude, we have constructed a class properties $\mathcal{P}$ defined over domains of sizes $|\mathcal{D}| = 2^{2k}$ but testing $\mathcal{P}$ and $\overline{\mathcal{P}}$ both require $\Omega(2^k) = \Omega(|\mathcal{D}|^{1/2})$ queries. Thus, neither the property or its complement is testable with a query complexity independent of the sizes of the domains, completing the proof. □

[8] In other words, suppose $x \in \mathbb{F}_{2^k}$ satisfies that $p(x) \neq g(x)$. Then the number of y's such that $\mathrm{Had}(p(x), y) \neq \mathrm{Had}(g(x), y)$ is exactly 2^{k-1}.

D Proof of Proposition 4

Let T_1 be the tester for $\mathcal{P}_1$ with query complexity $q_1(\epsilon)$ and let T_2 be the tolerant tester for $\mathcal{P}_2$ with query complexity $q_2(\epsilon_1, \epsilon_2)$. First we convert T_1 into another tester T'_1 such that, on input distance parameter ϵ, T'_1 makes $Q'_1(\epsilon)$ queries, where

$$Q'_1(x) = \begin{cases} q_1(x), & \text{if } x < \epsilon_1; \\ \max\{q_1(x), q_i(\epsilon_1)\}, & \text{otherwise.} \end{cases}$$

Set $P = \mathcal{P}_1 \setminus \mathcal{P}_2$ and define its tester T as follows: on input function f, T first simulates T_1 and then T_2. T accepts iff T_1 accepts and T_2 rejects. Since ϵ_1 is a constant, $Q'_1(\epsilon)) = O(\epsilon)$, and T has query complexity $O(q_1 + q_2)$.

For completeness, if $f \in \mathcal{P}$, then by assumption $f \in \mathcal{P}_1$ and $\mathrm{dist}(f, \mathcal{P}_2) \geqslant \epsilon_0 > \epsilon_2$. This implies that T_1 always rejects f, T_2 accepts f with probability at most $1/3$, and thus by a union bound argument T accepts f with probability at least $2/3$.

For soundness, suppose $\mathrm{dist}(f, \mathcal{P}) \geqslant \epsilon$. We consider two cases and note that in both of them, T accepts f with probability at most $1/3$.

Case 1. $\mathrm{dist}(f, \mathcal{P}_2) \leqslant \epsilon_1$.

Since T_2 is a tolerant tester, T_2 rejects f with probability at most $1/3$. Thus, T accepts with probability at most $1/3$ as well.

Case 2. $\mathrm{dist}(f, \mathcal{P}_2) > \epsilon_1$.

Since $\mathcal{P}_1$ is the union of $\mathcal{P}$ and $\mathcal{P}_2$, we can conclude that $\mathrm{dist}(f, \mathcal{P}_1) = \min\{\mathrm{dist}(f, \mathcal{P}), \mathrm{dist}(f, \mathcal{P}_2)\}$, which is at least $\min\{\epsilon, \epsilon_1\}$. Since T'_1 makes at least $\max\{q_1(\epsilon), q_1(\epsilon_1)\}$ queries, we know that T'_1 accepts f with probability at most $1/3$, and hence, T accepts f with probability at most $1/3$ as well. □