Testing Intersecting and Union-Closed Families

Xi Chen
Columbia University, New York, NY, USA

Anindya De
University of Pennsylvania, Philadelphia, PA, USA

Yuhao Li
Columbia University, New York, NY, USA

Shivam Nadimpalli
Columbia University, New York, NY, USA

Rocco A. Servedio
Columbia University, New York, NY, USA

Abstract

Inspired by the classic problem of Boolean function monotonicity testing, we investigate the testability of other well-studied properties of combinatorial finite set systems, specifically *intersecting* families and *union-closed* families. A function $f : \{0,1\}^n \to \{0,1\}$ is intersecting (respectively, union-closed) if its set of satisfying assignments corresponds to an intersecting family (respectively, a union-closed family) of subsets of $[n]$.

Our main results are that – in sharp contrast with the property of being a monotone set system – the property of being an intersecting set system, and the property of being a union-closed set system, both turn out to be information-theoretically difficult to test. We show that:

- For $\varepsilon \geq \Omega(1/\sqrt{n})$, any non-adaptive two-sided ε-tester for intersectingness must make $2^{\Omega(n^{1/4}/\sqrt{\varepsilon})}$ queries. We also give a $2^{\Omega(\sqrt{n \log(1/\varepsilon)})}$-query lower bound for non-adaptive one-sided ε-testers for intersectingness.
- For $\varepsilon \geq 1/2^{\Omega(n^{0.49})}$, any non-adaptive two-sided ε-tester for union-closedness must make $n^{\Omega(\log(1/\varepsilon))}$ queries.

Thus, neither intersectingness nor union-closedness shares the $\mathrm{poly}(n, 1/\varepsilon)$-query non-adaptive testability that is enjoyed by monotonicity.

To complement our lower bounds, we also give a simple $\mathrm{poly}(n^{\sqrt{n \log(1/\varepsilon)}}, 1/\varepsilon)$-query, one-sided, non-adaptive algorithm for ε-testing each of these properties (intersectingness and union-closedness). We thus achieve nearly tight upper and lower bounds for two-sided testing of intersectingness when $\varepsilon = \Theta(1/\sqrt{n})$, and for one-sided testing of intersectingness when $\varepsilon = \Theta(1)$.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms; Mathematics of computing → Combinatorics

Keywords and phrases Sublinear algorithms, property testing, computational complexity, monotonicity, intersecting families, union-closed families

Digital Object Identifier 10.4230/LIPIcs.ITCS.2024.33

Related Version *Full Version*: `https://arxiv.org/abs/2311.11119`

Funding *Xi Chen*: NSF grants IIS-1838154, CCF-2106429, and CCF-2107187.
Anindya De: NSF grants CCF-1910534 and CCF-2045128.
Yuhao Li: NSF grants IIS-1838154, CCF-2106429 and CCF-2107187.
Shivam Nadimpalli: NSF grants IIS-1838154, CCF-2106429, CCF-2211238, CCF-1763970, and CCF-2107187.
Rocco A. Servedio: NSF grants IIS-1838154, CCF-2106429, and CCF-2211238.

Acknowledgements This work was partially completed while some of the authors were visiting the Simons Institute for the Theory of Computing at UC Berkeley.

1 Introduction

Monotonicity testing is among the oldest and most intensively studied problems in property testing (see e.g. [30, 21, 26, 32, 9, 11, 12, 18, 17, 2, 19, 3, 34, 13, 4, 37, 5, 8] and the numerous references contained therein). The simplicity with which the core monotonicity testing problem can be formulated – given query access to an unknown $f : \{0,1\}^n \to \{0,1\}$, output "yes" if f is monotone and "no" if f is far in Hamming distance from every monotone function – belies the wealth of sophisticated technical ingredients and ideas (such as combinatorial shifting [30, 21], multidimensional limit theorems [18, 17], and isoperimetric inequalities [11, 34, 3, 37, 5, 8]) which have been deployed in both algorithms and lower bounds for this problem. Thanks to this body of work the basic problem of monotonicity testing is now fairly well understood: [34] gave an $\tilde{O}(\sqrt{n}/\varepsilon^2)$-query non-adaptive testing algorithm, and [19] gave an $\tilde{\Omega}(n^{1/3})$-query lower bound which holds even for adaptive algorithms.

Monotonicity testing has several intriguing features as a property testing problem:

- Since the class of all monotone functions is of doubly exponential size[1], the results mentioned above tell us that the query complexity of testing this class, which contains $N = 2^{2^{\Theta(n)}}$ functions, is $(\log \log N)^c$ for some constant $\frac{1}{3} \leq c \leq \frac{1}{2}$. This is an interesting contrast with both the $O(\log N)$ query complexity which suffices to test any class of N functions[2] and the constant query complexity (independent of N and depending only on the error parameter ε) of a number of other well-studied property testing problems such as linearity testing [6], testing linear separability [36], and testing dictatorship [38].
- The monotonicity of $f : \{0,1\}^n \to \{0,1\}$ is equivalent to having all pairs of inputs x, y satisfy a simple "pair condition," which is that

$$x \leq y \implies f(x) \leq f(y). \tag{1}$$

  Given this, it is natural to consider "pair testers" for monotonicity which work by drawing a pair of inputs $\boldsymbol{x}, \boldsymbol{y} \in \{0,1\}^n$ with $\boldsymbol{x} \leq \boldsymbol{y}$ according to some distribution over such pairs, and checking whether the pair violates monotonicity. Indeed, all known algorithms for testing monotonicity, including the state-of-the-art algorithm of [34], work in this fashion.
- Finally, we observe that a monotone function $f : \{0,1\}^n \to \{0,1\}$ can alternately be viewed as an *upward-closed* set system: this is a collection of subsets $\mathcal{S} \subseteq 2^{[n]}$, corresponding to the satisfying assignments of f, which has the property that for every subset $S \subseteq [n]$, if $S \in \mathcal{S}$ then $S \cup \{i\} \in \mathcal{S}$ for every $i \in [n]$.

This Work. Motivated by monotonicity testing, we propose to study other combinatorial property testing problems of a similar flavor. In particular, we are interested in the testability of properties which (a) are "very large" (meaning that the number of functions with the property is doubly exponential in n); (b) are defined by a natural condition on pairs or triples of inputs; and (c) correspond to well-studied properties of set systems. We focus on two specific properties of this sort, namely *intersecting* and *union-closed* set systems.

---

[1] Observe that any assignment of 0/1 values to the middle level of the Boolean hypercube $\{0,1\}$ corresponds to at least one monotone function, and hence there are at least $2^{\Omega(2^n/\sqrt{n})}$ many distinct monotone functions over $\{0,1\}^n$.

[2] This follows straightforwardly from the fact that $O(\log N)$ samples suffice to properly PAC learn any concept class of N Boolean functions [7] and the well-known reduction from proper PAC learning to property testing given in [31].

Intersectingness. A set system $\mathcal{S} \subseteq 2^{[n]}$ is said to be *intersecting* if any two sets $S_1, S_2 \in \mathcal{S}$ have a nonempty intersection, i.e. $S_1 \cap S_2 \neq \emptyset$. Intersecting families are intensively studied in extremal combinatorics, where they are the subject of many touchstone results, beginning with the seminal Erdös-Ko-Rado theorem [24] and continuing to the present day. Recent years have witnessed exciting progress on many problems dealing with intersecting families and their generalizations via analytic techniques that are highly relevant to the study of Boolean functions in theoretical computer science; see e.g. [28, 20, 22] and more generally [23] for a recent and extensive survey.

Translating the above definition to the setting of Boolean functions, a function $f : \{0,1\}^n \to \{0,1\}$ is intersecting if the following "pair condition" holds: whenever $f(x) = f(y) = 1$, there is (at least one) coordinate $i \in [n]$ such that $x_i = y_i = 1$. This is equivalent to

$$x \leq \overline{y} \implies f(x) \leq \overline{f(y)}, \tag{2}$$

i.e. if $x \leq \overline{y}$, then having $f(y) = 1$ implies that $f(x)$ must be 0, where $\overline{y} = (\overline{y}_1, \overline{y}_2, \ldots, \overline{y}_n)$ is the point in $\{0,1\}^n$ that is antipodal to y. Finally, we observe that any n-variable Boolean function whose satisfying assignments all have first bit 1 is an intersecting function, so indeed the set of all n-variable intersecting Boolean functions is of doubly exponential size (at least $2^{2^{n-1}}$).

Union-closedness. A set system $\mathcal{S} \subseteq 2^{[n]}$ is said to be *union-closed* if whenever S_1 and S_2 belong to $\mathcal{S}$ then $S_1 \cup S_2$ also belongs to $\mathcal{S}$. In the Boolean function setting, this corresponds to the "triple condition" that $f : \{0,1\}^n \to \{0,1\}$ satisfy

$$z = x \cup y \implies f(x)f(y) \leq f(z), \tag{3}$$

i.e. if $f(x) = f(y) = 1$ then $f(x \cup y)$ must also be 1. Union-closed families have long been of interest in combinatorics, in part due to the well-known "union-closed conjecture" of Frankl [27, 10], which states that in any union-closed family some element $i \in [n]$ must appear in at least half the sets in the family. Dramatic progress was recently made on the union-closed conjecture by Gilmer [29], who proved a weaker form of the conjecture with 1/2 replaced by 0.01 (this constant was subsequently improved to $\frac{3-\sqrt{5}}{2} \approx 0.38$ by [1, 15, 39, 40]). Since every monotone function is easily seen to be union-closed, union-closedness is a "large" property, with at least $2^{\Omega(2^n/\sqrt{n})}$ n-variable functions having the property.

In this paper we initiate the study of intersectingness and union-closedness from a property testing perspective. Given that (like monotonicity) these are "large" properties that are defined by a simple "pair" or "triple" property, it is natural to wonder: Is the query complexity of testing these properties similar to the query complexity of testing monotonicity, or are these properties harder – or easier – to test than monotonicity?

1.1 Main Results

As our main results, we show that both intersectingness and union-closedness are significantly more difficult to test than monotonicity: We give information-theoretic lower bounds which establish that neither of these properties admits a $\mathrm{poly}(n, 1/\varepsilon)$-query non-adaptive testing algorithm. We also give sub-exponential non-adaptive testing algorithms for each of these properties; our algorithms have one-sided error (they never reject functions which have the property), while most of our lower bounds are for testing algorithms that are allowed two-sided error. We turn now to a detailed description of our main results.

Positive Results: Algorithms for Testing Intersectingness and Union-Closedness. As a warm-up, and to develop intuition for these properties, we give simple testing algorithms for intersectingness and for union-closedness which have sub-exponential query complexity:

▶ **Theorem 1** (Testers for intersectingness and union-closedness). *There is a*

$$\mathrm{poly}(n^{\sqrt{n\log(1/\varepsilon)}}, 1/\varepsilon)\text{-}query$$

non-adaptive, one-sided[3] *algorithm for ε-testing whether an unknown $f : \{0,1\}^n \to \{0,1\}$ is intersecting versus ε-far from every intersecting function. The same is true for union-closedness.*

We defer the algorithms as well as their analyses to the full version of this paper. Theorem 1 is proved by analyzing a "pair tester" for intersectingness and a "triple tester" for union-closedness. The distribution of pairs (respectively, triples) used by our algorithm is extremely simple, so it is natural to wonder whether a more sophisticated algorithm, perhaps using a cleverer distribution over pairs or triples, could result in a tester with an improved query complexity (indeed, this would be analogous to how the cleverer distribution over pairs used in [11, 34] resulted in a better query complexity for testing monotonicity than the simple distribution that was used in [30]). However, our main results – lower bounds for testing intersectingness and union-closedness – indicate that there are strong information-theoretic limitations on the possible performance of any non-adaptive testing algorithm for these properties.

Negative Results: Lower Bounds for Testing. Our lower bounds show that both intersectingness and union-closedness are significantly harder to test than monotonicity: Neither of these properties has a $\mathrm{poly}(n, 1/\varepsilon)$-query non-adaptive testing algorithm, even if we allow two-sided error. (Recall that in contrast, the algorithms of [30, 11, 34] are all $\mathrm{poly}(n, 1/\varepsilon)$-query non-adaptive one-sided testing algorithms for monotonicity.) In more detail, our main lower bound for intersectingness is the following (in all of our lower bound theorem statements, $c > 0$ represents some sufficiently small absolute positive constant):

▶ **Theorem 2** (Two-sided lower bound for intersectingness). *For $c > \varepsilon \geq 1/\sqrt{n}$, any non-adaptive ε-testing algorithm for whether an unknown $f : \{0,1\}^n \to \{0,1\}$ is intersecting versus ε-far from intersecting must make $2^{\Omega(n^{1/4}/\sqrt{\varepsilon})}$ queries to f.*

When $\varepsilon = 1/\sqrt{n}$, the lower bound of Theorem 2 essentially matches the performance of our algorithm from Theorem 1, and even when ε is a constant, Theorem 2 gives a $2^{\Omega(n^{1/4})}$ lower bound. In view of the similarity between the defining conditions for monotonicity and intersectingness (Equation (1) and Equation (2)), we view Theorem 2 as a potentially surprising result.

By imposing a stricter one-sided error condition, we can establish a stronger lower bound which almost matches the one-sided algorithm from Theorem 1 even for constant ε:

▶ **Theorem 3** (One-sided lower bound for intersectingness). *For $c > \varepsilon \geq 2^{-n}$, any non-adaptive one-sided ε-testing algorithm for whether an unknown $f : \{0,1\}^n \to \{0,1\}$ is intersecting versus ε-far from intersecting must make $2^{\Omega(\sqrt{n\log(1/\varepsilon)})}$ queries to f.*

[3] A tester is *non-adaptive* if the choice of its i-th query point does not depend on the responses received to queries $1, \ldots, i-1$. A *one-sided* tester for a class of functions is one which must accept every function in the class with probability 1.

Turning to union-closedness, the lower bound we give is not as strong as for intersectingness, but it is strong enough to rule out a $\mathrm{poly}(n, 1/\varepsilon)$-query non-adaptive algorithm, again even allowing two-sided error:

▶ **Theorem 4** (Two-sided lower bound for union-closedness). *For $c > \varepsilon \geq 2^{-n^{0.49}}$, any non-adaptive ε-testing algorithm for whether an unknown $f : \{0,1\}^n \to \{0,1\}$ is union-closed versus ε-far from union-closed must make $n^{\Omega(\log(1/\varepsilon))}$ queries to f.*

As we discuss in Section 6 of the full version, an interesting goal for future work is to narrow the gap between our algorithm and our lower bound for testing union-closed families.

1.2 Techniques

In this section, we give a technical overview of our main results, starting with the lower bounds.

Lower Bounds. Our two-sided lower bound for intersectingness, Theorem 2, builds on a lower bound approach for *tolerant* monotonicity testing which was introduced in [37] and was recently quantitatively strengthened in [16]. As is standard for non-adaptive property testing lower bounds, [37] and [16] use Yao's minimax lemma and define a "yes"-distribution $\mathcal{D}_{\mathrm{yes}}$ and a "no"-distribution $\mathcal{D}_{\mathrm{no}}$ over Boolean functions; in the rest of this discussion we focus chiefly on [16]. A function $\boldsymbol{f}$ drawn from either of the [16] distributions $\mathcal{D}_{\mathrm{yes}}$ or $\mathcal{D}_{\mathrm{no}}$ is defined based on a random partition of the n variables into a (large) set of "control" variables and a (small) set of "action" variables. In both cases $\boldsymbol{f} \sim \mathcal{D}_{\mathrm{yes}}$ or $\boldsymbol{f} \sim \mathcal{D}_{\mathrm{no}}$, the definition of $\boldsymbol{f}$ involves a "Talagrand DNF," $\boldsymbol{T} = \boldsymbol{T}_1 \vee \cdots \vee \boldsymbol{T}_m$, which is essentially a random monotone DNF formula over the control variables.[4] The crucial assignments to $\boldsymbol{f}$ are the ones for which the control variables satisfy exactly one term $\boldsymbol{T}_i$ of the Talagrand DNF; for such an input string x, the value of $\boldsymbol{f}$ then depends on the setting of the action variables, and the difference between $\boldsymbol{f} \sim \mathcal{D}_{\mathrm{yes}}$ and $\boldsymbol{f} \sim \mathcal{D}_{\mathrm{no}}$ comes from how the function is defined over the action variables in each case. The values of the function on the action subcubes are carefully defined in such a way as to make it impossible for a testing algorithm to distinguish a "yes"-function from a "no"-function unless it manages to query two inputs x, x' which (i) both have their control variables set in such a way as to uniquely satisfy the same term $\boldsymbol{T}_i$, but (ii) differ on "many coordinates" among the action variables: essentially, one of x, x' must have its vector of action bits landing in the top portion of the action subcube while the other one must have its vector of action bits landing in the bottom portion. The crux of the non-adaptive lower bound of [16] is the tension between requirements (i) and (ii): if x and x' differ in too many coordinates then it is difficult to satisfy (i), but if they differ in too few coordinates then it is difficult to satisfy (ii).

In the setting of monotonicity testing, the [16] construction's yes-functions are only close to, but not actually, monotone; their non-monotonicity essentially comes from assignments for which the vector of action bits lands in the middle portion of the action subcube. This is why the mildly exponential lower bound proved in that paper only holds for *tolerant* monotonicity testing (indeed, the existence of highly efficient monotonicity testers [30, 19, 34] implies that quantitatively strong lower bounds such as those of [16] are impossible for "standard" non-tolerant monotonicity testing). The main component of our lower bound for intersectingness in this paper is a careful modification of the [16] construction; we show that,

[4] The earlier work [37] used a different function over the control variables instead of a Talagrand DNF.

perhaps surprisingly, for the modification that we introduce, the yes-functions have satisfying assignments which form a *perfectly* intersecting family, while the no-functions are far from intersecting. We thus obtain a quantitatively strong lower bound, similar to [16], already for the "standard" testing problem of intersectingness rather than the more challenging tolerant version.

Our $2^{\Omega(\sqrt{n \log(1/\varepsilon)})}$-query one-sided lower bound for intersectingness, Theorem 3, takes a related but somewhat simpler approach. In a nutshell, since for one-sided lower bounds it is not necessary to give a yes-distribution and establish indistinguishability of yes-functions and no-functions, it turns out that we can dispense with the Talagrand DNF part of the construction. Instead, our construction "hides" a randomly chosen "small" set of action bits in a simpler way (see Section 3.2 for details); since we do not need to use the Talagrand DNF, it turns out that we can have the "small" set of action bits be larger than in our intersectingness lower bound, and this lets us obtain a quantitatively stronger lower bound.

Finally, our $n^{\Omega(\log(1/\varepsilon))}$-query two-sided lower bound for union-closedness, like our two-sided intersectingness lower bound, uses the framework of control bits and action bits with a Talagrand DNF over the control bits. This construction uses a somewhat different definition of the yes- and no- functions over the action bits, which now ensures that a testing algorithm can distinguish yes-functions from no-functions only if it manages to query two inputs whose control variables satisfy the same term $\boldsymbol{T}_i$ but whose action variables are set to two particular *antipodal* assignments in the action cube. For this construction we use many fewer action bits than in the earlier construction (and the quantitative lower bound obtained is correspondingly weaker than the lower bound of the earlier construction); this is because in our no-functions, the distance to union-closedness is inverse exponential in the dimension of the action cubes.

Algorithms. Our algorithms for testing intersectingness and for testing union-closedness are similar at a high level; for conciseness we only describe the algorithm for testing union-closedness.

As is standard for testing algorithms, we consider the two possible scenarios. In the "yes" case, the given function f is union-closed. In the "no" case, the function f is ε-far in Hamming distance from any union-closed function.

At a conceptual level, the first simplification is as follows: given f, we can define a truncated version of f, call it f_{trunc} as follows: for any x such that $|x| \in [n/2 - T, n/2 + T]$ where $T = \sqrt{n \log(4/\varepsilon)}$, $f_{\mathsf{trunc}}(x) = f(x)$. If $|x| > n/2 + T$, we set $f_{\mathsf{trunc}}(x) = 1$ and if $|x| < n/2 - T$, we set $f_{\mathsf{trunc}}(x) = 0$. In other words, f_{trunc} is obtained by keeping it the same as f in the middle $2T$ layers; otherwise, it is set to 1 in the layers above the middle layers and 0 below it. Since all but $\varepsilon/2$ fraction of the mass of the discrete cube lies in the layers $[n/2 - T, n/2 + T]$, the following is immediate: (i) if f is union-closed, so is f_{trunc}; (ii) if f is ε-far from union-closed, f_{trunc} is also $\varepsilon/2$-far from union-closed. The above property of f_{trunc} ensures that instead of working with f, the algorithm can instead work with f_{trunc}.

Now, the main idea behind the algorithm is to search for *violations of union-closedness*. In this sense, our algorithm is similar in spirit to algorithms for monotonicity testing [30, 11, 34] which search for violations of monotonicity. In particular, we call a sequence $(x_1, \ldots, x_k, x_1 \cup \ldots \cup x_k)$ a *union-closed violating tuple* if $f(x_1) = \ldots = f(x_k) = 1$ and $f(x_1 \cup \ldots \cup x_k) = 0$ – we will abbreviate this as a UC-violating tuple. Note that if the algorithm finds a union-closed violating tuple in f, then it is a certificate for f not being union-closed.

The main technical lemma we prove is that if f is ε-far from union closed, then it has at least $\varepsilon \cdot 2^n$ UC-violating tuples which are end-disjoint. This means that for any two such tuples $(x_1, \ldots, x_k, x_1 \cup \ldots \cup x_k)$ and $(y_1, \ldots, y_k, y_1 \cup \ldots \cup y_k)$, the last coordinate

$(x_1 \cup \ldots \cup x_k) \neq (y_1 \cup \ldots \cup y_k)$. The proof of this lemma is quite simple – essentially, we show that the function f can be changed to a union closed function by only modifying it at points which are the last coordinate of a UC-violating tuple. Given this lemma, it follows that f must have at least $\varepsilon \cdot 2^n$ end-disjoint UC-violating tuples. Since f and f_{trunc} are $\varepsilon/2$-close to each other, it follows that f_{trunc} also has at least $\varepsilon/2 \cdot 2^n$ end-disjoint UC-violating tuples.

We next observe that a UC-violating tuple $(x_1, \ldots, x_k, x_1 \cup \ldots \cup x_k)$ for f_{trunc} is such that (i) for each $1 \leq i \leq k$, $||x_i| - n/2| \leq T$; (ii) $||x_1 \cup \ldots \cup x_k| - n/2| \leq T$. Let us call a point $x = x_1 \cup \ldots \cup x_k$ a witness if there is a UC-violating tuple $(x_1, \ldots, x_k, x_1 \cup \ldots \cup x_k)$ satisfying the above conditions. From the fact that f_{trunc} also has at least $\varepsilon/2 \cdot 2^n$ end-disjoint UC-violating tuples, it follows that there are at least $\varepsilon/2 \cdot 2^n$ points which are a witness.

Our algorithm now proceeds as follows: We sample a random point $\boldsymbol{x} \in \{0,1\}^n$ conditioned on $||\boldsymbol{x}| - n/2| \leq T$. Next, we query f on $\boldsymbol{x}$ as well as all the points in the set $\boldsymbol{x}_\downarrow := \{y \leq \boldsymbol{x} : ||y| - n/2| \leq T\}$. We then check if there are any points $y_1, \ldots, y_k \in \boldsymbol{x}_\downarrow$ such that $(y_1, \ldots, y_k, \boldsymbol{x})$ is a UC-violating tuple. Note that if f is union-closed, then the algorithm is certainly not going to find a UC-violating tuple, i.e., it has perfect completeness. On the other hand, if f is at least ε-far from union closed, then the point $\boldsymbol{x}$ sampled above is a witness with probability $\varepsilon/2$. If $\boldsymbol{x}$ is a witness then since we are querying every point in $\boldsymbol{x}_\downarrow$, the algorithm is going to find a UC-violating tuple.

Thus, repeating the above procedure say $100/\varepsilon$ times, the algorithm will still have perfect completeness. On the other hand, if f is ε-far from union-closed, it is going to find a UC-violating tuple with probability at least 0.9. The query complexity of the algorithm is given by $O(1/\varepsilon) \cdot |\boldsymbol{x}_\downarrow|$. As $|\boldsymbol{x}_\downarrow|$ is uniformly bounded by $n^{O(\sqrt{n \log(1/\varepsilon)})}$, this establishes the upper bound on the query complexity of our algorithm. (While the algorithm described above is not a "triple tester," an easy modification of the algorithm and its analysis yields a triple tester with similar query complexity.)

1.3 Related Work

As mentioned earlier, some of the technical specifics of our lower bound constructions build off of the tolerant testing lower bounds of [37] and [16]; in particular, the idea, first introduced by [37], of "hiding" a set of action variables among the entire set of input variables was a significant influence on the lower bound constructions of the current paper. More generally, the entire broad literature on monotonicity testing of Boolean functions (i.e. testing upward-closed set systems) provided the conceptual backdrop for a study of the testability of other types of combinatorial finite set systems.

We note that the recent work of Filmus et al. [25] (see also [14]) studies the problem of "AND-testing," which at first glance may seem to be related to the problems we consider. The "AND-property" is that of satisfying the implication

$$z = x \cap y \implies f(z) = f(x) \wedge f(y) \tag{4}$$

for *every* $x, y \in \{0,1\}^n$; the main result of [25], roughly speaking, is that the only functions which have a high probability of satisfying Equation (4) for uniform random x, y are functions which are close to being either a constant-function or an AND of some subset of the n input variables.

Despite the superficial resemblance between Equation (3) and Equation (4), it turns out that the AND-property and the properties we consider are of quite different character from each other. To see this, observe that the only functions $f : \{0,1\}^n \to \{0,1\}$ which perfectly satisfy the AND-property are constant functions and AND-functions; hence there are only

$O(2^n)$ many possible yes-functions, and every yes-function must have a very precise and rigid structure (and a very simple description). This is quite different from the intersectingness and union-closedness properties we study; each of these properties has $2^{2^{\Theta(n)}}$ many yes-functions, and hence yes-functions do not need to be so highly structured (and by standard counting arguments almost all yes-functions require highly complex descriptions). As another point of difference, the [25] result mentioned above implies that there is an $O_\varepsilon(1)$-query non-adaptive one-sided tester for the AND-property. In contrast, our Theorem 4 shows that even two-sided non-adaptive testers for the property of union-closedness must have a query complexity which not only depends on n, but in fact is at least $n^{\Omega(\log(1/\varepsilon))}$.

2 Preliminaries

We will write

$$\binom{[n]}{k} := \{S \subseteq [n] : |S| = k\}$$

to denote the collection of all k-element subsets of $[n]$, and for a subset $I \subseteq [n]$ we will write $\binom{[n]}{I}$ to denote $\cup_{j \in I}\binom{[n]}{j}$. We will denote the 0/1-indicator of an event A by $\mathbf{1}\{A\}$. All probabilities and expectations will be with respect to the uniform distribution over the relevant domain unless stated otherwise. We use boldfaced letters such as $\boldsymbol{x}, \boldsymbol{f}$, and $\boldsymbol{A}$ to denote random variables (which may be real-valued, vector-valued, function-valued, or set-valued; the intended type will be clear from the context). We write $\boldsymbol{x} \sim \mathcal{D}$ to indicate that the random variable $\boldsymbol{x}$ is distributed according to probability distribution $\mathcal{D}$.

▶ **Notation 5.** *Given a string $x \in \{0,1\}^n$ and a set $A \subseteq [n]$, we write $x_A \in \{0,1\}^A$ to denote the $|A|$-bit string obtained by restricting x to coordinates in A, i.e. $x_A := (x_i)_{i \in A}$, and we write $|x|$ to denote the number of 1's in x.*

We will frequently view strings in $\{0,1\}^n$ as subsets of $[n]$ and vice versa; i.e. for $x, y \in \{0,1\}^n$ we refer to "$x \cap y$" to mean the string in $\{0,1\}^n$ which has a 1 in coordinate i iff $x_i = y_i = 1$.

Given two Boolean functions $f, g : \{0,1\}^n \to \{0,1\}$, we define the *distance* between f and g (denoted by $\mathrm{dist}(f,g)$) to be the normalized Hamming distance between f and g, i.e. $\mathrm{dist}(f,g) := \mathbf{Pr}_{\boldsymbol{x} \sim \{0,1\}^n}\left[f(\boldsymbol{x}) \neq g(\boldsymbol{x})\right]$. A *property* $\mathcal{P}$ is a collection of Boolean functions; we say that a function $f : \{0,1\}^n \to \{0,1\}$ is *ε-far from the property $\mathcal{P}$* if $\mathrm{dist}(f,\mathcal{P}) := \min_{g \in \mathcal{P}} \mathrm{dist}(f,g) \geq \varepsilon$.

2.1 Lower Bounds for Testing Algorithms

Our query-complexity lower bounds for testing algorithms are obtained via Yao's minimax principle [42], which we recall below. (We remind the reader that an algorithm for the problem of ε-property testing is correct on an input function f provided that it outputs "yes" if f *perfectly* satisfies the property and outputs "no" if f is ε-far from the property; if the distance to the property is strictly between 0 and ε then the algorithm is correct regardless of what it outputs.)

▶ **Theorem 6** (Yao's principle)**.** *To prove a q-query lower bound on the worst-case query complexity of any non-adaptive randomized testing algorithm, it suffices to give a distribution $\mathcal{D}$ on instances such that for any q-query non-adaptive* deterministic *algorithm $\mathcal{A}$, we have*

$$\mathbf{Pr}_{\boldsymbol{f} \sim \mathcal{D}}\left[\mathcal{A} \text{ is correct on } \boldsymbol{f}\right] \leq 99.9\%.$$

Here 99.9% can be replaced by any universal constant in $[0,1)$.

2.2 Talagrand's Random DNF

We define a useful distribution over Boolean functions that will play a central role in the proofs of our lower bounds. The construction is a slight generalization of a distribution over DNF (disjunctive normal form) formulas that was constructed by Talagrand [41]. The generalization we consider, which was also studied in [16], is that we allow a parameter ε to control the size of each term and the number of terms; the original construction corresponds to $\varepsilon = 1$.

▶ **Definition 7** (Talagrand's random DNF). *Let $\varepsilon \in (0,1]$ and let $L := 0.1 \cdot 2^{\sqrt{n}/\varepsilon}$. Let $\mathsf{Talagrand}(n,\varepsilon)$ be the following distribution on ordered tuples of L monotone terms: for each $i = 1, \ldots, L$, the i-th term is obtained by independently drawing a set $\boldsymbol{T}_i \subseteq [n]$ where each set $\boldsymbol{T}_i$ is obtained by drawing $\sqrt{n}/\varepsilon$ elements of $[n]$ independently and with replacement. We use $\boldsymbol{T}$ to denote the ordered tuple $\boldsymbol{T} = (\boldsymbol{T}_1, \cdots, \boldsymbol{T}_L)$ which is a draw from $\mathsf{Talagrand}(n,\varepsilon)$. Then a "Talagrand DNF" is given by*

$$\boldsymbol{f}(x) = \bigvee_{\ell=1}^{L} \left(\bigwedge_{j \in \boldsymbol{T}_\ell} x_j \right).$$

It is clear that any Talagrand DNF obtained by a draw from $\mathsf{Talagrand}(n,\varepsilon)$ is a monotone function.

We will frequently view $T_i \subseteq [n]$ as the term $\bigwedge_{j \in T_i} x_j$, where we say $T_i(x) = 1$ if and only if $x_j = 1$ for all $j \in T_i$. We may also write $T = (T_1, \cdots, T_k)$ to represent a DNF, which is defined by the disjunction of the terms T_i. We will often be interested in the probability of a random input $\boldsymbol{x} \sim \{0,1\}^n$ satisfying a unique term $\boldsymbol{T}_i$ in a Talagrand DNF; towards this, we introduce the following notation:

▶ **Notation 8.** *Given a DNF $T = (T_1, \cdots, T_k)$ where each T_i is a term, we define the collection of* terms of T satisfied by x, *written $S_T(x)$, as $S_T(x) := \{\ell \in [k] : T_\ell(x) = 1\}$.*

The following claim shows that on average over the draw of $\boldsymbol{T} \sim \mathsf{Talagrand}(n,\varepsilon)$, an $\Omega(\varepsilon)$ fraction of strings from $\{0,1\}^n$ satisfy a *unique* term in the Talagrand DNF (i.e. $|S_{\boldsymbol{T}}(x)| = 1$ for $\Omega(\varepsilon)$-fraction of $x \in \{0,1\}^n$). We note that an elegant argument of Kane [33] gives this for $\varepsilon = \Theta(1)$, but this argument does not extend to the setting of small ε which we require. The proof of the following appears in [16] and is repeated in the full version of this paper.

▶ **Proposition 9.** *For $\varepsilon \in (0,1]$, let $\boldsymbol{T} \sim \mathsf{Talagrand}(n,\varepsilon)$ be as in Definition 7. Then*

$$\Pr_{\boldsymbol{T},\boldsymbol{x}} \left[|S_{\boldsymbol{T}}(\boldsymbol{x})| = 1 \right] = \Omega\left(\max\{\varepsilon, 1/\sqrt{n}\} \right).$$

3 Lower Bounds for Testing Intersecting Families

We now present our lower bound for two-sided non-adaptive testers for intersecting families. As mentioned earlier, the construction builds closely on the earlier constructions of [37, 16] which were used in those papers for tolerant testing lower bounds.

Let $\varepsilon \in (0, c]$ be a parameter with $c > \varepsilon \geq c_0/\sqrt{n}$ for some sufficiently large constant c_0 and sufficiently small constant $c > 0$. We start with some objects that we need in the construction of the two distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$. We partition the variables $x_1, \cdots, x_n$ into *control* variables and *action* variables as follows: Let $a := \sqrt{n}/\varepsilon$ and let $A \subseteq [n]$ be a fixed subset of $[n]$ of size a. Let $C := [n] \setminus A$. We refer to the variables x_i for $i \in C$ as

control variables and the variables x_i for $i \in A$ as *action variables*. We first define two pairs of functions over $\{0,1\}^A$ on the action variables as follows (we will use these functions later in the definition of $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$):

$$g^{(+,0)}(x_A) = \begin{cases} 0 & |x_A| > \frac{a}{2} + \sqrt{a}; \\ 0 & |x_A| \in [\frac{a}{2} - \sqrt{a}, \frac{a}{2} + \sqrt{a}]; \\ 0 & |x_A| < \frac{a}{2} - \sqrt{a}. \end{cases} \quad g^{(+,1)}(x_A) = \begin{cases} 1 & |x_A| > \frac{a}{2} + \sqrt{a}; \\ 0 & |x_A| \in [\frac{a}{2} - \sqrt{a}, \frac{a}{2} + \sqrt{a}]; \\ 1 & |x_A| < \frac{a}{2} - \sqrt{a}. \end{cases}$$

and

$$g^{(-,0)}(x_A) = \begin{cases} 1 & |x_A| > \frac{a}{2} + \sqrt{a}; \\ 0 & |x_A| \in [\frac{a}{2} - \sqrt{a}, \frac{a}{2} + \sqrt{a}]; \\ 0 & |x_A| < \frac{a}{2} - \sqrt{a}. \end{cases} \quad g^{(-,1)}(x_A) = \begin{cases} 0 & |x_A| > \frac{a}{2} + \sqrt{a}; \\ 0 & |x_A| \in [\frac{a}{2} - \sqrt{a}, \frac{a}{2} + \sqrt{a}]; \\ 1 & |x_A| < \frac{a}{2} - \sqrt{a}. \end{cases}$$

Now we are ready to define the distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ over $f : \{0,1\}^{n+2} \to \{0,1\}$. We follow the convention that random variables are in boldface and fixed quantities are in the standard typeface.

A function $\boldsymbol{f}_{\text{yes}} \sim \mathcal{D}_{\text{yes}}$ is drawn as follows. We start by sampling a subset $\boldsymbol{A} \subseteq [n]$ of size a uniformly at random and let $\boldsymbol{C} \coloneqq [n] \setminus \boldsymbol{A}$. Note that there are in total $n - a$ control variables. We let $L \coloneqq 0.1 \cdot 2^{\sqrt{n-a}/\varepsilon}$ and draw an L-term monotone Talagrand DNF $\boldsymbol{T} \sim \mathsf{Talagrand}(n - a, \varepsilon)$ on $\boldsymbol{C}$ as described in Definition 7. Finally, we sample L random bits $\boldsymbol{b} \in \{0,1\}^L$ uniformly at random. Given $\boldsymbol{A}, \boldsymbol{T}$ and $\boldsymbol{b}$, $\boldsymbol{f}_{\text{yes}}$ is defined by letting $\boldsymbol{f}_{\text{yes}}(x,0,0) = \boldsymbol{f}_{\text{yes}}(x,1,1) = 0$ for all $x \in \{0,1\}^n$, and letting

$$\boldsymbol{f}_{\text{yes}}(x,0,1) = \begin{cases} 0 & |S_{\boldsymbol{T}}(x_{\boldsymbol{C}})| \neq 1; \\ g^{(+,0)}(x_{\boldsymbol{A}}) & S_{\boldsymbol{T}}(x_{\boldsymbol{C}}) = \{\ell\} \text{ and } \boldsymbol{b}_\ell = 0; \\ g^{(+,1)}(x_{\boldsymbol{A}}) & S_{\boldsymbol{T}}(x_{\boldsymbol{C}}) = \{\ell\} \text{ and } \boldsymbol{b}_\ell = 1. \end{cases}$$

$$\boldsymbol{f}_{\text{yes}}(x,1,0) = \begin{cases} 0 & |S_{\boldsymbol{T}}(\overline{x}_{\boldsymbol{C}})| \neq 1; \\ g^{(+,1)}(x_{\boldsymbol{A}}) & S_{\boldsymbol{T}}(\overline{x}_{\boldsymbol{C}}) = \{\ell\} \text{ and } \boldsymbol{b}_\ell = 0; \\ g^{(+,0)}(x_{\boldsymbol{A}}) & S_{\boldsymbol{T}}(\overline{x}_{\boldsymbol{C}}) = \{\ell\} \text{ and } \boldsymbol{b}_\ell = 1. \end{cases}$$

(Recall that $\overline{x}$ is the bitwise complement of string x).

To draw a function $\boldsymbol{f}_{\text{no}} \sim \mathcal{D}_{\text{no}}$, we sample $\boldsymbol{A}, \boldsymbol{T}$ and $\boldsymbol{b}$ exactly as in the definition of $\mathcal{D}_{\text{yes}}$ above, but we use $g^{(+,b)}$ and $g^{(-,b)}$ functions in a different way than in the $\mathcal{D}_{\text{yes}}$ functions described above. In more detail, $\boldsymbol{f}_{\text{no}}$ is defined by $\boldsymbol{f}_{\text{no}}(x,0,0) = \boldsymbol{f}_{\text{no}}(x,1,1) = 0$ for all $x \in \{0,1\}^n$, and

$$\boldsymbol{f}_{\text{no}}(x,0,1) = \begin{cases} 0 & |S_{\boldsymbol{T}}(x_{\boldsymbol{C}})| \neq 1; \\ g^{(-,0)}(x_{\boldsymbol{A}}) & S_{\boldsymbol{T}}(x_{\boldsymbol{C}}) = \{\ell\} \text{ and } \boldsymbol{b}_\ell = 0; \\ g^{(-,1)}(x_{\boldsymbol{A}}) & S_{\boldsymbol{T}}(x_{\boldsymbol{C}}) = \{\ell\} \text{ and } \boldsymbol{b}_\ell = 1. \end{cases}$$

$$\boldsymbol{f}_{\text{no}}(x,1,0) = \begin{cases} 0 & |S_{\boldsymbol{T}}(\overline{x}_{\boldsymbol{C}})| \neq 1; \\ g^{(-,0)}(x_{\boldsymbol{A}}) & S_{\boldsymbol{T}}(\overline{x}_{\boldsymbol{C}}) = \{\ell\} \text{ and } \boldsymbol{b}_\ell = 0; \\ g^{(-,1)}(x_{\boldsymbol{A}}) & S_{\boldsymbol{T}}(\overline{x}_{\boldsymbol{C}}) = \{\ell\} \text{ and } \boldsymbol{b}_\ell = 1. \end{cases}$$

See Figures 1 and 2 for illustrations of the yes- and no- functions.

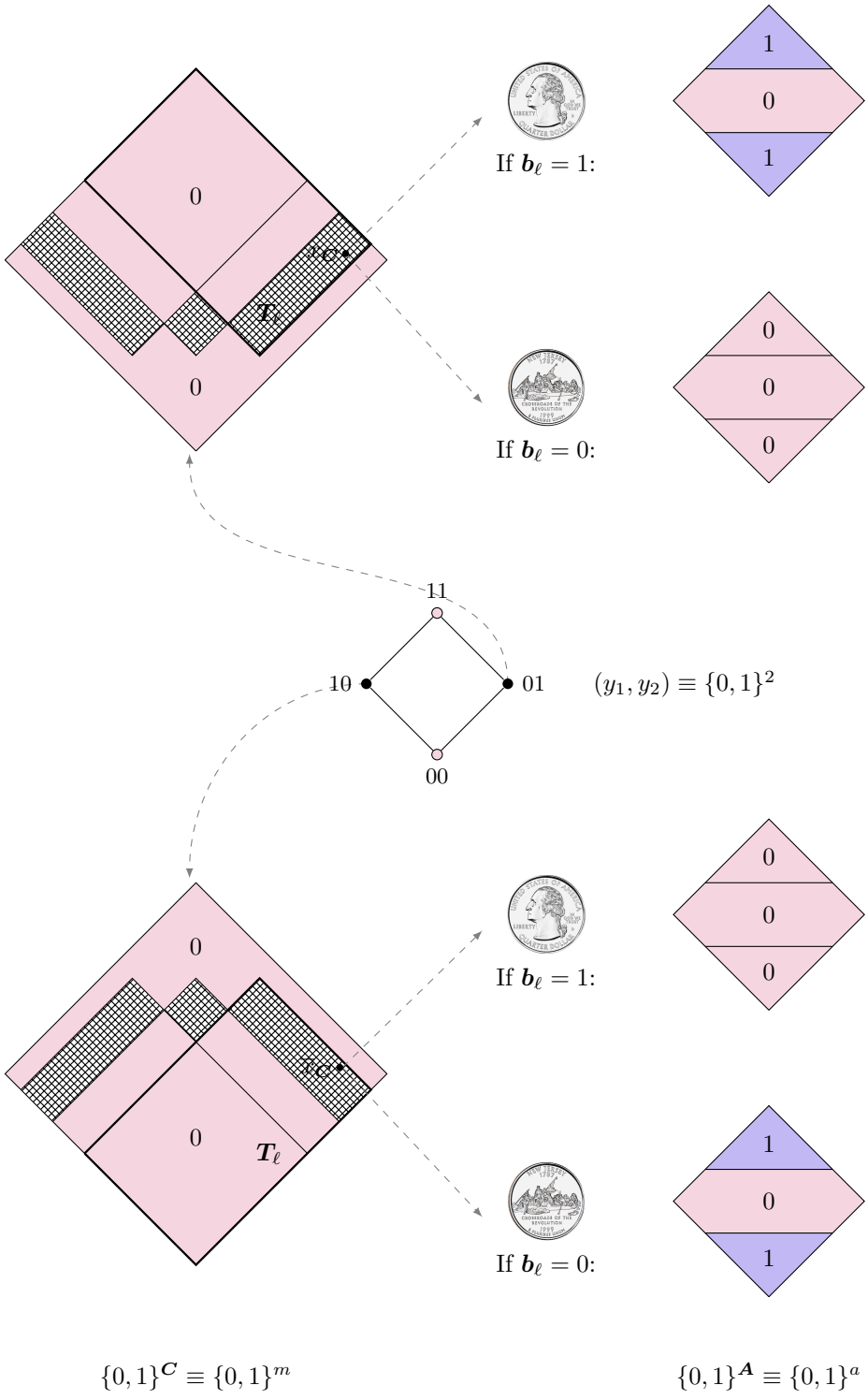


Figure 1 A draw of $f_{\text{yes}} \sim \mathcal{D}_{\text{yes}}$. All our hypercubes adopt the convention that the bottom-most point is $(0, \ldots, 0)$ and the topmost point is $(1, \ldots, 1)$, and horizontal lines denote Hamming levels. Given an input $(x, y_1, y_2) \in \{0,1\}^n \times \{0,1\}^2$ we follow the arrows starting with $\{0,1\}^2$ in the center. The cross-hatched region in the control cube $\{0,1\}^C$ corresponds to inputs satisfying a unique Talagrand DNF term T_ℓ. The pink regions correspond to 0 assignments and blue regions to 1 assignments.

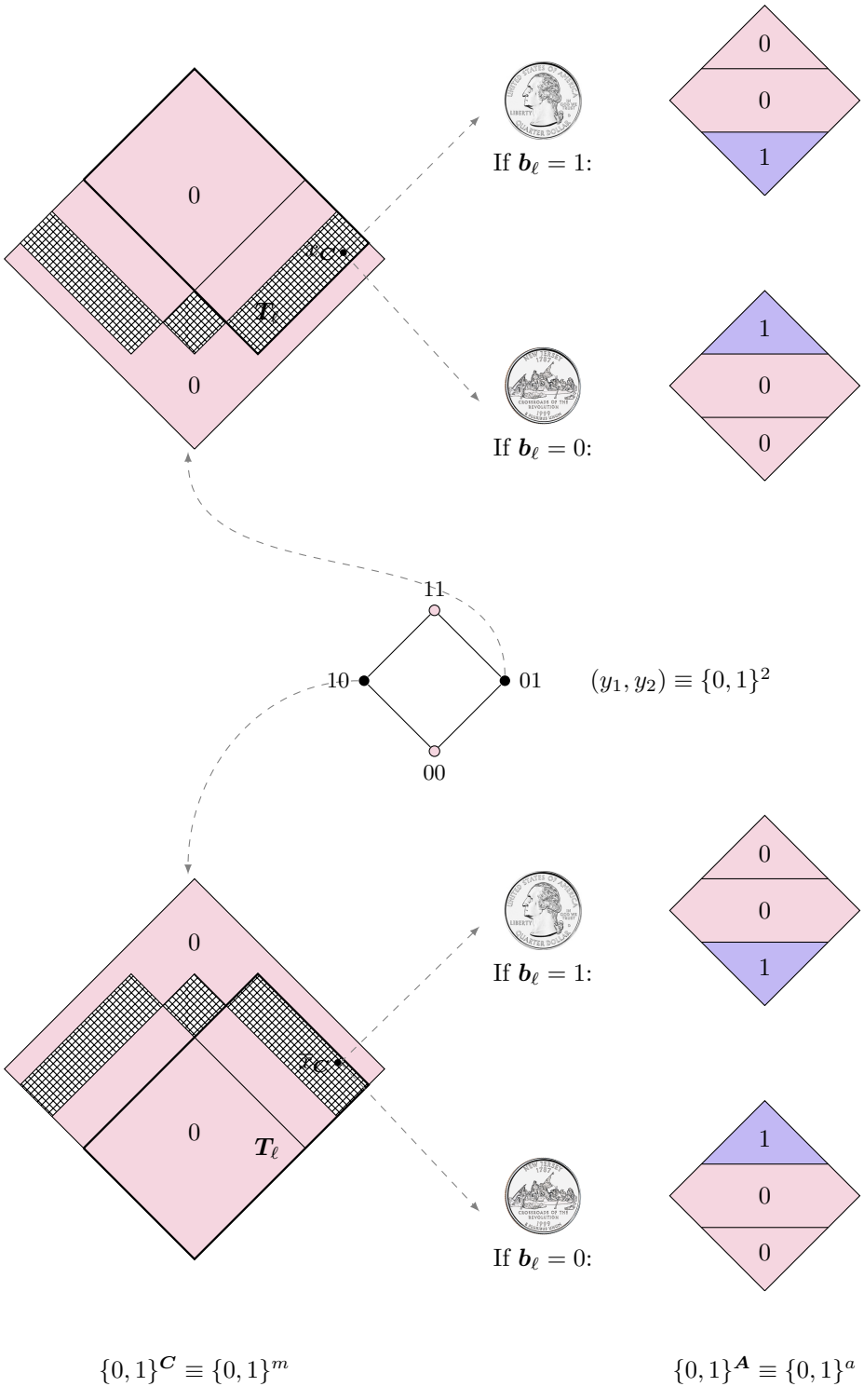


Figure 2 A draw of $\boldsymbol{f}_{\text{no}} \sim \mathcal{D}_{\text{no}}$. Our conventions are as in Figure 1.

The proofs of the following lemmas are deferred to the full version:

▶ **Lemma 10.** *Every function f_{yes} in the support of $\mathcal{D}_{\text{yes}}$ is intersecting.*

▶ **Lemma 11.** *With probability at least 0.01, $\boldsymbol{f}_{\text{no}} \sim \mathcal{D}_{\text{no}}$ is $\Omega(\varepsilon)$-far from intersecting.*

3.1 Indistinguishability of $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$

In this section we establish the indistinguishability of the distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$. Specifically, for any nonadaptive deterministic algorithm $\mathcal{A}$ with query complexity $q = 2^{0.1n^{1/4}/\sqrt{\varepsilon}}$, we show that

$$\Pr_{\boldsymbol{f}_{\text{yes}} \sim \mathcal{D}_{\text{yes}}}[\mathcal{A} \text{ accepts } \boldsymbol{f}_{\text{yes}}] \leq \Pr_{\boldsymbol{f}_{\text{no}} \sim \mathcal{D}_{\text{no}}}[\mathcal{A} \text{ accepts } \boldsymbol{f}_{\text{no}}] + o_n(1). \tag{5}$$

Our arguments closely follow the approach for proving indistinguishability that was used in [16].

We begin with some simplifying assumptions: for any point $u \in \{0,1\}^{n+2}$ that is queried by the algorithm $\mathcal{A}$ we assume that $u_{n+1} \neq u_{n+2}$ (since otherwise the answer to the query must be 0), and we assume that for each point $u \in \{0,1\}^{n+2}$ that is queried by $\mathcal{A}$ the point $\overline{u}$ is also queried as well (since this only affects the query complexity by at most a factor of two). So the set of q query points of $\mathcal{A}$ can be characterized by a set $Q_{\mathcal{A}} \coloneqq \{x^1, \cdots, x^q\} \subseteq \{0,1\}^n$, where both $(x^i, 0, 1)$ and $(\overline{x}^i, 1, 0)$ are queried for each $i \in [q]$.

A crucial step of the argument is that the only way for $\mathcal{A}$ to distinguish $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ is to query two points x^i, x^j with $S_T(x^i_C) = S_T(x^j_C) = \{\ell\}$ for some $\ell \in [L]$ such that one is in the top region and the other is in the bottom region of the action cube, namely $|x^i_A| > \frac{a}{2} + \sqrt{a}$ and $|x^j_A| < \frac{a}{2} - \sqrt{a}$. We let Bad denote this event (that $Q_{\mathcal{A}}$ contains two points x^i, x^j satisfying the above conditions).

Formally, let us write $\mathcal{A}(f)$ to denote the sequence of q answers to the queries made by $\mathcal{A}$ to f. We write $\text{view}_{\mathcal{A}}(\mathcal{D}_{\text{yes}})$ (respectively $\text{view}_{\mathcal{A}}(\mathcal{D}_{\text{no}})$) to be the distribution of $\mathcal{A}(f)$ for $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ (respectively $\boldsymbol{f} \sim \mathcal{D}_{\text{no}}$). The following claim asserts that conditioned on Bad not happening, the distributions $\text{view}_{\mathcal{A}}(\mathcal{D}_{\text{yes}}|_{\overline{\mathsf{Bad}}})$ and $\text{view}_{\mathcal{A}}(\mathcal{D}_{\text{no}}|_{\overline{\mathsf{Bad}}})$ are identical.

▶ **Lemma 12.** $\text{view}_{\mathcal{A}}(\mathcal{D}_{\text{yes}}|_{\overline{\mathsf{Bad}}}) = \text{view}_{\mathcal{A}}(\mathcal{D}_{\text{no}}|_{\overline{\mathsf{Bad}}})$.

Proof. The distributions of the partition of $[n]$ into control variables $\boldsymbol{C}$ and action variables $\boldsymbol{A}$ are identical for $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$. So fix an arbitrary partition C and A. As the distribution of the Talagrand DNF $\boldsymbol{T} \sim \mathsf{Talagrand}(m, \varepsilon)$ is also identical, we fix an arbitrary T.

We divide the points $Q_{\mathcal{A}}$ into disjoint groups according to x_C. More precisely, for every $\ell \in [L]$, let $Q_{\mathcal{A}}(\ell) = \{x^i \mid S_T(x^i_C) = \{\ell\}\}$. The points outside $\bigcup_{\ell \in [L]} Q_{\mathcal{A}}(\ell)$ are not important as $\boldsymbol{f}$ will be identically 0 for both $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$.

Let $\boldsymbol{f}_\ell(x)$ denote the function $\boldsymbol{f}(x, 0, 1)$ restricted to points in $Q_{\mathcal{A}}(\ell)$, and let $\boldsymbol{f}'_\ell(x)$ similarly denote the function $\boldsymbol{f}(\overline{x}, 1, 0)$ restricted to inputs $x \in Q_{\mathcal{A}}(\ell)$. Note that for a fixed $\ell \in [L]$, the functions $\boldsymbol{f}_\ell(x)$ and $\boldsymbol{f}'_\ell(x)$ only depend on the random bit $\boldsymbol{b}_\ell$. As a result, the distributions of functions $\boldsymbol{f}_\ell(x)$ and $\boldsymbol{f}'_\ell(x)$ for different ℓ are independent.

So fix an arbitrary $\ell \in [L]$. The condition that Bad does not happen implies that *either* $|x_A| > a/2 + \sqrt{a}$ for all $x \in Q_{\mathcal{A}}(\ell)$ *or* $|x_A| < a/2 - \sqrt{a}$ for all $x \in Q_{\mathcal{A}}(\ell)$, which holds for both $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$. So we have $\boldsymbol{f}'_\ell(x) = 1 - \boldsymbol{f}_\ell(x)$ for all $x \in Q_{\mathcal{A}}(\ell)$, which also holds for both $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$.

Finally, noticing that the distribution of $\boldsymbol{f}_\ell(x)$ is simply a uniform random bit $\boldsymbol{b}_\ell$ for both $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$, this finishes the proof. ◀

Next, we show that the probability that Bad happens is small (recall that $q = 2^{0.1n^{1/4}/\sqrt{\varepsilon}}$):

▶ **Lemma 13.** *For any set of points $Q_{\mathcal{A}} = \{x^1, \cdots, x^q\} \subseteq \{0,1\}^n$, $Pr[\mathsf{Bad}] = o_n(1)$.*

Proof. Fix any two points $x, y \in \{0,1\}^n$. We will upper bound the probability that $S_{\boldsymbol{T}}(x_{\boldsymbol{C}}) = S_{\boldsymbol{T}}(y_{\boldsymbol{C}}) = \{\ell\}$ for some $\ell \in [L]$ and $|x_{\boldsymbol{A}}| < \frac{a}{2} - \sqrt{a}$ and $|y_{\boldsymbol{A}}| > \frac{a}{2} + \sqrt{a}$. Call this specific event Bad_{xy}.

Let I_{01} be the set of indices i such that $x_i = 0$ and $y_i = 1$. On the one hand, to have Bad_{xy} happen, we must have that

$$|I_{01} \cap \boldsymbol{A}| \geq 2\sqrt{a}. \tag{$\diamond$}$$

On the other hand, to have $S_{\boldsymbol{T}}(x_{\boldsymbol{C}}) = S_{\boldsymbol{T}}(y_{\boldsymbol{C}}) = \{\ell\}$, we must have that

$$\text{There exists an } \ell \in [L] \text{ such that } S_{\boldsymbol{T}}(x) = S_{\boldsymbol{T}}(y) = \{\ell\}. \tag{$\star$}$$

So we have $\Pr[\mathsf{Bad}_{xy}] \leq \min(\Pr[\diamond], \Pr[\star])$; we will show that $\min(\Pr[\diamond], \Pr[\star]) \leq 2^{-0.05n^{1/4}/\sqrt{\varepsilon}}$. Let $t = |I_{01}|$. Then by the random choice of the coordinates defining the action cube $\boldsymbol{A}$, we have

$$\begin{aligned}\Pr[\diamond] &\leq \Pr\left[\mathrm{Bin}\left(a, \frac{t}{n-a}\right) \geq 2\sqrt{a}\right] \leq \binom{a}{2\sqrt{a}} \cdot \left(\frac{t}{n-a}\right)^{2\sqrt{a}} \\ &\leq \left(\frac{ea}{2\sqrt{a}}\right)^{2\sqrt{a}} \cdot \left(\frac{t}{n-a}\right)^{2\sqrt{a}} \leq \left(\frac{et\sqrt{a}}{2(n-a)}\right)^{2\sqrt{a}} \leq \left(\frac{et\sqrt{a}}{2(1-\frac{1}{c_0})n}\right)^{2\sqrt{a}}.\end{aligned}$$

To bound $\Pr[\star]$, we use

$$\begin{aligned}\Pr[\star] &= \Pr[S_{\boldsymbol{T}}(x) = S_{\boldsymbol{T}}(y) \ \&\ \exists \ell \in [L] \text{ such that } S_{\boldsymbol{T}}(y) = \{\ell\}] \\ &\leq \Pr[S_{\boldsymbol{T}}(x) = S_{\boldsymbol{T}}(y) \ \mid\ \exists \ell \in [L] \text{ such that } S_{\boldsymbol{T}}(y) = \{\ell\}] \\ &\leq \max_{\ell \in [L]} \Pr[S_{\boldsymbol{T}}(x) = S_{\boldsymbol{T}}(y) \ \mid\ S_{\boldsymbol{T}}(y) = \{\ell\}] \\ &\leq \left(1 - \frac{t}{n-a}\right)^{\sqrt{n-a}/\varepsilon} \leq e^{-t/(\varepsilon\sqrt{n-a})} \leq e^{-t/(\varepsilon\sqrt{n})},\end{aligned}$$

where the last line above is by the definition of the random process $\boldsymbol{T} \sim \mathsf{Talagrand}(n-a, \varepsilon)$.

When $t \leq \frac{1}{4}n^{3/4}/\sqrt{\varepsilon}$, we have $\Pr[\diamond] \leq 2^{-n^{1/4}/\sqrt{\varepsilon}}$. When $t \geq \frac{1}{4}n^{3/4}/\sqrt{\varepsilon}$, we have $\Pr[\star] \leq 2^{-0.25n^{1/4}/\sqrt{\varepsilon}}$.

So overall we have

$$\Pr[\mathsf{Bad}_{xy}] \leq \min(\Pr[\diamond], \Pr[\star]) \leq 2^{-0.25n^{1/4}/\sqrt{\varepsilon}}.$$

By a union bound for all pairs of points of $Q_{\mathcal{A}}$, we know that

$$\Pr[\mathsf{Bad}] \leq 2^{-0.25n^{1/4}/\sqrt{\varepsilon}} \cdot \left(2^{0.1n^{1/4}/\sqrt{\varepsilon}}\right)^2 = o_n(1),$$

and the lemma is proved. ◀

Now we are ready to prove Theorem 2.

Proof of Theorem 2. Let $\mathcal{D} = \frac{1}{2}\{\mathcal{D}_{\text{yes}} + \mathcal{D}_{\text{no}}\}$. Then we have

$$\begin{aligned}
\mathop{\mathbf{Pr}}_{\boldsymbol{f}\sim\mathcal{D}}[\mathcal{A} \text{ is correct on } \boldsymbol{f}] &= \frac{1}{2}\left(\mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{yes}}\sim\mathcal{D}_{\text{yes}}}[\mathcal{A} \text{ is correct on } \boldsymbol{f}_{\text{yes}}] + \mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{no}}\sim\mathcal{D}_{\text{no}}}[\mathcal{A} \text{ is correct on } \boldsymbol{f}_{\text{no}}]\right) \\
&= \frac{1}{2}\left(\mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{yes}}\sim\mathcal{D}_{\text{yes}}}[\mathcal{A} \text{ accepts } \boldsymbol{f}_{\text{yes}}] + \mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{no}}\sim\mathcal{D}_{\text{no}}}[\mathcal{A} \text{ is correct on } \boldsymbol{f}_{\text{no}}]\right) \qquad (6) \\
&\le \frac{1}{2}\left(\mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{yes}}\sim\mathcal{D}_{\text{yes}}}[\mathcal{A} \text{ accepts } \boldsymbol{f}_{\text{yes}}] + 0.99 + 0.01\mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{no}}\sim\mathcal{D}_{\text{no}}}[\mathcal{A} \text{ rejects } \boldsymbol{f}_{\text{no}}]\right) \qquad (7) \\
&= \frac{1}{2}\left(\mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{yes}}\sim\mathcal{D}_{\text{yes}}}[\mathcal{A} \text{ accepts } \boldsymbol{f}_{\text{yes}}] + 1 - 0.01\mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{no}}\sim\mathcal{D}_{\text{no}}}[\mathcal{A} \text{ accepts } \boldsymbol{f}_{\text{no}}]\right) \\
&\le \frac{199}{200} + \frac{1}{200}\left(\mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{yes}}\sim\mathcal{D}_{\text{yes}}}[\mathcal{A} \text{ accepts } \boldsymbol{f}_{\text{yes}}] - \mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{no}}\sim\mathcal{D}_{\text{no}}}[\mathcal{A} \text{ accepts } \boldsymbol{f}_{\text{no}}]\right) \\
&= \frac{199}{200} + \frac{\mathbf{Pr}[\mathsf{Bad}]}{200}\left(\mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{yes}}\sim\mathcal{D}_{\text{yes}}|_{\mathsf{Bad}}}[\mathcal{A} \text{ accepts } \boldsymbol{f}_{\text{yes}}] - \mathop{\mathbf{Pr}}_{\boldsymbol{f}_{\text{no}}\sim\mathcal{D}_{\text{no}}|_{\mathsf{Bad}}}[\mathcal{A} \text{ accepts } \boldsymbol{f}_{\text{no}}]\right) \qquad (8) \\
&\le \frac{199}{200} + \frac{\mathbf{Pr}[\mathsf{Bad}]}{200} \\
&\le \frac{199}{200} + o_n(1), \qquad (9)
\end{aligned}$$

where Equation (6) is because of Lemma 10, Equation (7) is because $\boldsymbol{f}_{\text{no}}$ is not ε-far from intersecting with probability at most 0.99 thanks to Lemma 11, Equation (8) is from Lemma 12, and Equation (9) follows from Lemma 13. Theorem 2 now follows from Yao's minimax principle (Theorem 6). ◀

3.2 A $2^{\Omega(\sqrt{n\log(1/\varepsilon)})}$ Lower Bound for One-Sided Non-adaptive Testers of Intersectingness

In this section we prove Theorem 3, by giving a $2^{\Omega(\sqrt{n\log(1/\varepsilon)})}$-query complexity lower bound against any *non-adaptive* and *one-sided* algorithm testing ε-intersectingness. This almost matches the query complexity of our $n^{O(\sqrt{n\log(1/\varepsilon)})}/\varepsilon$-query one-sided non-adaptive algorithm even for constant ε.

Since we are working against one-sided algorithms, it suffices for us to describe a distribution $\mathcal{D}_{\text{no}}$ over $f : \{0,1\}^{n+2} \to \{0,1\}$ of "no"-functions (functions that are far from intersecting). Let $K = \sqrt{n\ln(1/\varepsilon)}$. A draw from our $\mathcal{D}_{\text{no}}$ distribution is obtained as follows: first, we sample a subset $\boldsymbol{A} \subseteq [n]$ of size $a = n/100$ uniformly at random (looking ahead, 100 will be an important constant later in the proof). Then $\boldsymbol{f}_{\text{no}} \sim \mathcal{D}_{\text{no}}$ is defined by letting $\boldsymbol{f}_{\text{no}}(x,0,0) = \boldsymbol{f}_{\text{no}}(x,1,1) = 0$ for all $x \in \{0,1\}^n$, and

$$\boldsymbol{f}_{\text{no}}(x,0,1) = \boldsymbol{f}_{\text{no}}(x,1,0) = \begin{cases} 0 & |x| \notin [n/2 - 10K, n/2 + 10K]; \\ 0 & |x_{\boldsymbol{A}}| > n/200 + K; \\ 0 & |x_{\boldsymbol{A}}| \in [n/200 - K, n/200 + K]; \\ 1 & |x_{\boldsymbol{A}}| < n/200 - K. \end{cases}$$

The constant "10" above will also be important vis-a-vis the "100" in the definition of the size of $\boldsymbol{A}$.

We first show that every $\boldsymbol{f}_{\text{no}} \sim \mathcal{D}_{\text{no}}$ is $\varepsilon^{O(1)}$-far from intersecting (observe that this suffices for our claimed lower bound, since the difference between ε and $\varepsilon^{O(1)}$ is swallowed up by the log and the big-Omega):

▶ **Lemma 14.** *Every $\boldsymbol{f}_{\mathrm{no}}$ in the support of $\mathcal{D}_{\mathrm{no}}$ is $\varepsilon^{O(1)}$-far from intersecting.*

Proof. Fix an arbitrary $A \subseteq [n]$ with size $a = n/100$, which determines a function f_{no} in the support of $\mathcal{D}_{\mathrm{no}}$. For the convenience of notations, we use $C \coloneqq [n] \setminus A$.

By the same argument as Claim 12 from the full version, we know for any $0 \le w < n/200$, the bipartite graph $(P_w, P_{n/100-w})$ with poset relations as edges has a perfect matching. Next, we use the Chernoff bound (which upper bounds the lower tail of the Binomial distribution) and a "reverse Chernoff bound" (which *lower* bounds the lower tail of the Binomial distribution) to show that

$$|\{x \in \{0,1\}^A \mid |x| \in [n/200 - 5K, n/200 - K]\}| \ge (\varepsilon^{1800} - \varepsilon^{5000})2^a = \Omega(\varepsilon^{1800}) \cdot 2^a.$$

To this end, for $w \in [0, n/200]$, let $\mathcal{P}_{\le w}$ to denote $\{x \in \{0,1\}^A \mid |x| \le w\}$. Then it suffices to show that

$$|\mathcal{P}_{\le n/200-5K}| \le \varepsilon^{5000} \cdot 2^a,$$

which follows from the standard Chernoff bound, and

$$|\mathcal{P}_{\le n/200-K}| \ge \varepsilon^{1800} \cdot 2^a,$$

which follows from the following "reverse Chernoff bound:"

▶ **Lemma 15** ([35], Lemma 4)**.** *Let X be the sum of k independent 0/1 random variables. For any $K \in (0, pk/2]$ and $p \in [0, 1/2]$ such that $K^2/(pk) \ge 3$, if each random variable is 1 with probability at most p, then*

$$\mathbf{Pr}[X \le pk - K] \ge \exp(-9K^2/(pk)).$$

Next, consider any $x \in \{0,1\}^n$ such that $|x| \in [n/2 - 10K, n/2]$ and $|x_A| \in [n/200 - 5K, n/200 - K]$. Let $y \in \{0,1\}^n$ be such that $y_C = x_C$ and y_A is the matched point of x_C in the perfect matching. Then we have $|y| \in [n/2 - 10K, n/2 + 10K]$ and $|y_A| \in [n/200 + K, n/200 + 5K]$.

Note that for any such pair (x, y) we have $x \le y$, $f(x, 0, 1) = 1$ and $f(\overline{y}, 1, 0) = 1$, which serves as an I-violating pair. Since the edges in a perfect matching are vertex-disjoint, we have the number of I-violating pairs is at least the number of $x \in \{0,1\}^n$ such that $|x| \in [n/2 - 10K, n/2]$ and $|x_A| \in [n/200 - 5K, n/200 - K]$.

We have shown that

$$|\{x \in \{0,1\}^A \mid |x| \in [n/200 - 5K, n/200 - K]\}| = \Omega(\varepsilon^{1800}) \cdot 2^a.$$

Note also that

$$|\{x \in \{0,1\}^C \mid |x| \in [99n/200 - 5K, 99n/200]\}| = \Omega(1) \cdot 2^{n-a}.$$

This finishes the proof. ◀

Below we show that for any nonadaptive deterministic query algorithm $\mathcal{A}$ with query complexity $q = 2^{0.9\sqrt{n \log(1/\varepsilon)}}$ the probability that $\mathcal{A}$ succeeds in finding a violation of intersectingness is $o_n(1)$; this proves Theorem 3.

Proof of Theorem 3. We establish the following lemma, from which the theorem follows by a straightforward union bound:

▶ **Lemma 16.** *For any two points $x, y \in \{0,1\}^n$ such that $|x|, |y| \in [n/2 - 10K, n/2 + 10K]$ and $x \leq y$,*

$$\mathop{\mathbf{Pr}}_{\boldsymbol{A}}[|x \cap \boldsymbol{A}| < n/200 - K \text{ and } |y \cap \boldsymbol{A}| > n/200 + K] \leq 2^{-2K}.$$

Proof. Let I be the indices i such that $x_i = 0$ and $y_i = 1$ and let $t = |I|$. Then we know $0 \leq t \leq 20K$. On the other hand, in order for the event $|x \cap \boldsymbol{A}| < n/200 - K$ and $|y \cap \boldsymbol{A}| > n/200 + K$ to happen, the set $\boldsymbol{A}$ has to hit at least $2K$ many indices in I. So

$$\begin{aligned}
&\mathop{\mathbf{Pr}}_{\boldsymbol{A}}[|x \cap \boldsymbol{A}| < n/200 - K \text{ and } |y \cap \boldsymbol{A}| > n/200 + K] \\
&\leq \mathbf{Pr}\left[\mathrm{Bin}\left(n/100, \frac{20K}{0.99n}\right) \geq 2K\right] \leq \binom{n/100}{2K} \cdot \left(\frac{20K}{0.99n}\right)^{2K} \\
&\leq \left(\frac{en/100}{2K}\right)^{2K} \cdot \left(\frac{20K}{0.99n}\right)^{2K} \\
&\leq \left(\frac{10e}{99}\right)^{2K} \leq 2^{-2K},
\end{aligned}$$

completing the proof. ◀

By a union bound over all pairs of query strings where $q = 2^{0.9K} = 2^{0.9\sqrt{n \log(1/\varepsilon)}}$, it follows that the probability that $\mathcal{A}$ succeeds in finding a violation of intersectingness is $o_n(1)$. Since a one-sided tester must find such a violation in order to reject, this finishes the proof. ◀

4 Lower bounds for Testing Union-Closed Families

In this section, we prove a $n^{\Omega(\log(1/\varepsilon))}$-query lower bound against non-adaptive algorithms for testing union-closedness (with either one-sided or two-sided error). We describe the hard distributions in Section 4.1 and then prove Theorem 4 in Section 4.2.

4.1 The $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ Distributions

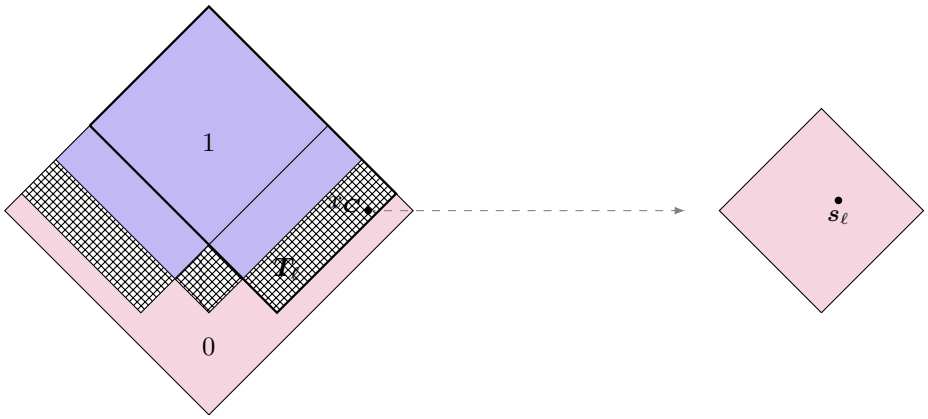
Our construction of the hard distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ are inspired by the constructions for the lower bound against intersectingness testing in Section 3; in particular, our hard functions will also comprise of a truncated Talagrand random DNF on a set of "control bits" $\boldsymbol{C}$, and then a function tailored to the union-closedness property on a set of "action bits" $\boldsymbol{A}$. We illustrate both $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ in Figure 3, and start by describing the $\mathcal{D}_{\text{yes}}$ distribution:

▶ **Definition 17.** *Given $\varepsilon > 0$, a draw of a Boolean function $\boldsymbol{f}_{\text{yes}} : \{0,1\}^n \to \{0,1\}$ from the distribution $\mathcal{D}_{\text{yes}} := \mathcal{D}_{\text{yes}}(n, \varepsilon)$ is obtained as follows:*

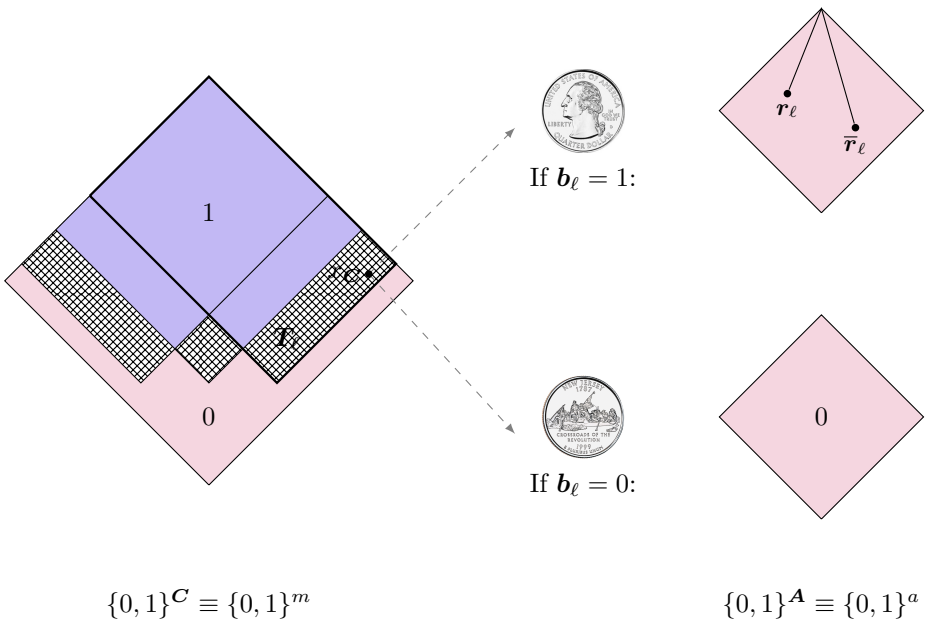
1. *Draw a random set of $a := \log(1/\varepsilon)$ coordinates $\boldsymbol{A} \subseteq [n]$, i.e.*

$$\boldsymbol{A} \sim \binom{[n]}{a}, \qquad \text{and set} \qquad \boldsymbol{C} := [n] \setminus \boldsymbol{A}.$$

   Let $c := |\boldsymbol{C}| = n - a$.
2. *Let $L := 0.1 \cdot 2^{\sqrt{c}}$ and draw an L-term monotone Talagrand DNF $\boldsymbol{T} \sim \mathsf{Talagrand}(c, 1)$ as defined in Definition 7 on $\{0,1\}^{\boldsymbol{C}}$.*
3. *For each $\ell \in [L]$, independently draw a uniformly random a-bit string $\boldsymbol{s}_\ell \in \{0,1\}^{\boldsymbol{A}}$.*



(a) A draw of $\boldsymbol{f}_{\text{yes}} \sim \mathcal{D}_{\text{yes}}$



(b) A draw of $\boldsymbol{f}_{\text{no}} \sim \mathcal{D}_{\text{no}}$

Figure 3 An illustration of the yes- and no-distributions for the union-closedness lower bound. Our conventions are as in Figure 1. In (b), if $\boldsymbol{b}_\ell = 1$ then as long as $\boldsymbol{r}_\ell \notin \{0^a, 1^a\}$ the action cube $\{0,1\}^{\boldsymbol{A}}$ will contain a single violation of union-closedness.

4. *Output the function*

$$\boldsymbol{f}_{\text{yes}}(x_{\boldsymbol{C}}, x_{\boldsymbol{A}}) := \begin{cases} 1 & |S_{\boldsymbol{T}}(x_{\boldsymbol{C}})| \geq 2 \\ \mathbf{1}\{x_{\boldsymbol{A}} = \boldsymbol{s}_\ell\} & S_{\boldsymbol{T}}(x_{\boldsymbol{C}}) = \{\ell\} \\ 0 & |S_{\boldsymbol{T}}(x_{\boldsymbol{C}})| = 0 \end{cases}$$

where $S_{\boldsymbol{T}}$ is as defined in Notation 8.

It is straightforward to verify that functions drawn from $\mathcal{D}_{\text{yes}}$ are indeed union-closed:

▷ Claim 18. Every function f_{yes} in the support of $\mathcal{D}_{\text{yes}}$ is union-closed.

We now turn to a description of the $\mathcal{D}_{\text{no}}$ distribution.

▶ **Definition 19.** *Given $\varepsilon > 0$, a draw of a Boolean function $\boldsymbol{f}_{\text{no}} : \{0,1\}^n \to \{0,1\}$ from the distribution $\mathcal{D}_{\text{no}} := \mathcal{D}_{\text{no}}(n, \varepsilon)$ is obtained as follows:*

1. *Draw a random set of $a := \log(1/\varepsilon)$ coordinates $\boldsymbol{A} \subseteq [n]$, i.e.*

$$\boldsymbol{A} \sim \binom{[n]}{a}, \qquad \text{and set} \qquad \boldsymbol{C} := [n] \setminus \boldsymbol{A}.$$

   Let $c := |\boldsymbol{C}| = n - a$.
2. *Let $L := 0.1 \cdot 2^{\sqrt{c}}$ and draw an L-term monotone Talagrand DNF $\boldsymbol{T} \sim \mathsf{Talagrand}(c, 1)$ as defined in Definition 7 on $\{0,1\}^{\boldsymbol{C}}$.*
3. *For each $\ell \in [L]$, independently draw a uniformly random a-bit string $\boldsymbol{r}_\ell \in \{0,1\}^{\boldsymbol{A}}$ as well as a uniformly random bit $\boldsymbol{b}_\ell \in \{0,1\}$.*
4. *Output the function*

$$\boldsymbol{f}_{\text{yes}}(x_{\boldsymbol{C}}, x_{\boldsymbol{A}}) := \begin{cases} 1 & |S_{\boldsymbol{T}}(x_{\boldsymbol{C}})| \geq 2 \\ \boldsymbol{b}_\ell \cdot \mathbf{1}\{x_{\boldsymbol{A}} \in \{\boldsymbol{r}_\ell, \overline{\boldsymbol{r}}_\ell\}\} & S_{\boldsymbol{T}}(x_{\boldsymbol{C}}) = \{\ell\} \\ 0 & |S_{\boldsymbol{T}}(x_{\boldsymbol{C}})| = 0 \end{cases}$$

   where $\overline{\boldsymbol{r}}_\ell := 1^a - \boldsymbol{r}_\ell$ is the antipode of $\boldsymbol{r}_\ell$.

As illustrated by Figure 3, we associated each Talagrand term $\boldsymbol{T}_i$ with a uniformly random bit $\boldsymbol{b}_\ell$. If $\boldsymbol{b}_\ell = 1$ then the action cube comprises a single union-closedness violation,[5] and if $\boldsymbol{b}_\ell = 0$ then the action cube has zero satisfying assignments. This ensures that in expectation, the measure of a function drawn from $\mathcal{D}_{\text{no}}$ is indistinguishable from that of a function drawn from $\mathcal{D}_{\text{yes}}$. The proof of the following is deferred to the full version:

▷ Claim 20. With probability at least 0.001, a function $\boldsymbol{f}_{\text{no}} \sim \mathcal{D}_{\text{no}} := \mathcal{D}_{\text{no}}(n, \varepsilon)$ satisfies $\text{dist}(\boldsymbol{f}_{\text{no}}, g) \geq \Omega(\varepsilon)$ for every union-closed function $g : \{0,1\}^n \to \{0,1\}$.

4.2 Indistinguishability of the Hard Distributions

In this section, we establish the indistinguishability of the distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ and prove Theorem 4. Our proof will closely follow the approach used in Section 3.1 to prove a lower bound against intersectingness testers.

[5] This is with the exception of $\boldsymbol{r}_\ell = 0^a$ or 1^a; in this case $\boldsymbol{r}_\ell \cup \overline{\boldsymbol{r}}_\ell = 1^a$ and so the function on the action bits will indeed be union-closed. Note, however, that this only happens with probability $1/2^a$.

As before, we will write $Q_{\mathcal{A}} := \{x^1, \ldots, x^q\} \subseteq \{0,1\}^n$ for the set of points queried by the algorithm. The argument will crucially rely on the fact that the only way for $\mathcal{A}$ to distinguish $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ is to draw two antipodal points from the same action cube, i.e. if there exist x^i and x^j such that $S_T(x^i_C) = S_T(x^j_C) = \{\ell\}$ for some $\ell \in [L]$ and x^i_A and x^j_A are antipodes; as before, we write Bad to denote this event. With $\text{view}_{\mathcal{A}}$ defined as in Section 3.1, we have the following:

▶ **Lemma 21.** *We have* $\text{view}_{\mathcal{A}}(\mathcal{D}_{\text{yes}}|_{\overline{\mathsf{Bad}}}) = \text{view}_{\mathcal{A}}(\mathcal{D}_{\text{no}}|_{\overline{\mathsf{Bad}}})$.

Proof. As before the distributions of the partition of $[n]$ into $\boldsymbol{C} \sqcup \boldsymbol{A}$ are identical for both $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$, so we may fix an arbitrary partition. As the distribution of the Talagrand DNF $\boldsymbol{T} \sim \mathsf{Talagrand}(c, 1)$ is also identical, we can fix an arbitrary T. We define

$$Q_{\mathcal{A}}(\ell) := \Big\{x^i : S_T(x^i_C) = \{\ell\}\Big\}.$$

Note that the points outside $\bigcup_{\ell \in [L]} Q_{\mathcal{A}}(\ell)$ do not matter as the the function is identically 0 or 1 for both $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$. We will abuse notation and view $Q_{\mathcal{A}}(\ell)$ as a subset of the action cube $\{0,1\}^a$ corresponding to the Talagrand term T_ℓ.

We will write $\boldsymbol{f}_\ell$ for the function restricted to inputs in $Q_{\mathcal{A}}(\ell)$, and will write $\mathcal{A}(\boldsymbol{f}_\ell)$ for the sequence of answers to the queries made by $\mathcal{A}$ to $\boldsymbol{f}_\ell$ (i.e. the sequence of answers to queries by $\mathcal{A}$ on inputs in $Q_{\mathcal{A}}(\ell)$). We will write $\text{view}_{\mathcal{A},\ell}(\mathcal{D}_{\text{yes}})$ (respectively $\text{view}_{\mathcal{A},\ell}(\mathcal{D}_{\text{no}})$) to be the distribution of $\mathcal{A}(\boldsymbol{f}_\ell)$ for $\boldsymbol{f}_\ell \sim \mathcal{D}_{\text{yes}}$ (respectively $\boldsymbol{f}_\ell \sim \mathcal{D}_{\text{no}}$). Since $Q_{\mathcal{A}}$ is partitioned as $Q_{\mathcal{A}} = \bigsqcup_{\ell \in [L]} Q_{\mathcal{A}}(\ell)$, note that in order to show that $\text{view}_{\mathcal{A}}(\mathcal{D}_{\text{yes}}|_{\overline{\mathsf{Bad}}}) = \text{view}_{\mathcal{A}}(\mathcal{D}_{\text{no}}|_{\overline{\mathsf{Bad}}})$, it suffices to show that $\text{view}_{\mathcal{A},\ell}(\mathcal{D}_{\text{yes}}|_{\overline{\mathsf{Bad}}}) = \text{view}_{\mathcal{A},\ell}(\mathcal{D}_{\text{no}}|_{\overline{\mathsf{Bad}}})$; this is what we will establish below.

Fixing an action cube $\{0,1\}^a$ (which is indexed by $\ell \in [L]$), note that the actions cubes in the yes- and no-distributions can be equivalently described as follows:

1. Draw a uniformly random pair of points $(\boldsymbol{y}, \overline{\boldsymbol{y}})$ from the 2^{a-1} pairs $(x, \overline{x})$ for $x \in \{0,1\}^a$, and draw a uniformly random bit $\boldsymbol{b}_\ell$.
2. We consider the "yes" and "no" cases separately:
   a. In the "yes" case, if $\boldsymbol{b}_\ell = 1$, then set $\boldsymbol{s}_\ell = \boldsymbol{y}$; otherwise set $\boldsymbol{s}_\ell = \overline{\boldsymbol{y}}$.
   b. In the "no" case, set $(\boldsymbol{r}_\ell, \overline{\boldsymbol{r}}_\ell) = (\boldsymbol{y}, \overline{\boldsymbol{y}})$; and if $\boldsymbol{b}_\ell = 0$, then the function $\boldsymbol{f}|_\ell$ is defined to be identically zero on the action cube (cf. Definition 19 and Figure 3).

Note that conditioned on Bad not happening, we have that none of the query points in $Q_{\mathcal{A}}(\ell)$ are antipodes of each other. We now split into two cases depending on whether either $\boldsymbol{y}$ or $\overline{\boldsymbol{y}}$ is in the query set $Q_{\mathcal{A}}(\ell)$:

1. If $\boldsymbol{y}, \overline{\boldsymbol{y}} \notin Q_{\mathcal{A}}(\ell)$, then note that $\text{view}_{\mathcal{A},\ell}(\mathcal{D}_{\text{yes}}|_{\overline{\mathsf{Bad}}}) = \text{view}_{\mathcal{A},\ell}(\mathcal{D}_{\text{no}}|_{\overline{\mathsf{Bad}}})$ since $\boldsymbol{f}_\ell$ is identically 0 on $Q_{\mathcal{A}}(\ell)$ in both the "yes" and the "no" cases.
2. Otherwise, since we conditioned on $\overline{\mathsf{Bad}}$, only one of $\boldsymbol{y}, \overline{\boldsymbol{y}}$ can be in $Q_{\mathcal{A}}(\ell)$; without loss of generality, suppose that it is $\boldsymbol{y}$. In both the "yes" and the "no" cases, $\boldsymbol{y}$ is a 1-input if and only if $\boldsymbol{b}_\ell = 1$, and the function is identically 0 on all other points. (Recall that we view points of $Q_{\mathcal{A}}(\ell)$ as a subset of the action cube $\{0,1\}^a$ corresponding to the Talagrand DNF term T_ℓ.)

It follows that $\text{view}_{\mathcal{A},\ell}(\mathcal{D}_{\text{yes}}|_{\overline{\mathsf{Bad}}}) = \text{view}_{\mathcal{A},\ell}(\mathcal{D}_{\text{no}}|_{\overline{\mathsf{Bad}}})$, and since $Q_{\mathcal{A}}$ is partitioned by the indices $\ell \in [L]$, we have $\text{view}_{\mathcal{A}}(\mathcal{D}_{\text{yes}}|_{\overline{\mathsf{Bad}}}) = \text{view}_{\mathcal{A}}(\mathcal{D}_{\text{no}}|_{\overline{\mathsf{Bad}}})$, completing the proof. ◀

Next, we will show that Bad happens with $o_n(1)$ probability:

▶ **Lemma 22.** *For any set of points* $Q_{\mathcal{A}} = \{x^1, \ldots, x^q\} \subseteq \{0,1\}^n$ *where* $q := n^{0.001 \log(1/\varepsilon)}$, *we have* $\mathbf{Pr}[\mathsf{Bad}] = o_n(1)$.

Proof. For $x, y \in \{0,1\}^n$, let Bad_{xy} be the event that $S_{\boldsymbol{T}}(x_{\boldsymbol{C}}) = S_{\boldsymbol{T}}(y_{\boldsymbol{C}}) = \{\ell\}$ for some $\ell \in [L]$ and $x_{\boldsymbol{A}} = \overline{y}_{\boldsymbol{A}}$. We will upper bound the probability of Bad_{xy} in what follows.

Let $J \subseteq [n]$ be the coordinates in which x and y differ, i.e. $J := \{i \in [n] : x_i \neq y_i\}$. Define the event $\diamond$ as:

$$\boldsymbol{A} \subseteq J. \tag{$\diamond$}$$

We also define the event $\star$ as before as

$$\text{There exists an } \ell \in [L] \text{ such that } S_{\boldsymbol{T}}(x) = S_{\boldsymbol{T}}(y) = \{\ell\}. \tag{$\star$}$$

By definition of Bad_{xy}, we have that $\mathbf{Pr}[\mathsf{Bad}_{xy}] \leq \min\left\{\mathbf{Pr}[\star], \mathbf{Pr}[\diamond]\right\}$. In the rest of the proof, we will establish that

$$\min\left\{\mathbf{Pr}[\star], \mathbf{Pr}[\diamond]\right\} \leq \Theta\left(\frac{1}{n}\right)^{0.01a}, \tag{10}$$

from which the lemma follows immediately by taking a union bound over all q^2 pairs $(x, y) \in Q_{\mathcal{A}} \times Q_{\mathcal{A}}$. Note that $\mathbf{Pr}[\diamond] = \mathbf{Pr}\left[\boldsymbol{A} \subseteq J\right] \leq \left(\frac{e|J|}{n}\right)^a$ via standard bounds on binomial coefficients. On the other hand, proceeding as in the proof of Lemma 13, we have

$$\mathbf{Pr}[\star] \leq \max_{\ell \in [L]} \mathbf{Pr}\left[S_{\boldsymbol{T}}(x) = S_{\boldsymbol{T}}(y) \mid S_{\boldsymbol{T}}(y) = \{\ell\}\right] \leq \left(1 - \frac{1}{\sqrt{c}}\right)^{|J|} \leq \exp\left(\frac{-|J|}{\sqrt{c}}\right)$$

where the final line follows from the definition of $\mathsf{Talagrand}(c, 1)$. In particular, note that if $|J| \leq n^{0.5}$, then

$$\mathbf{Pr}[\star] \leq \left(\frac{e}{n^{0.5}}\right)^a,$$

and if $|J| > n^{0.5}$ then we have

$$\mathbf{Pr}[\diamond] \leq \exp\left(\frac{-n^{0.5}}{\sqrt{n - \log(1/\varepsilon)}}\right) \ll \left(\frac{1}{n}\right)^{\Theta(a)}$$

where the final inequality uses the fact that $\varepsilon \geq \Theta\left(\frac{1}{2^{n^{0.49}}}\right)$. Putting everything together establishes Equation (10) which in turn completes the proof. ◀

Theorem 4 follows from Lemmas 21 and 22 *mutatis mutandis* as Theorem 2 follows from Lemmas 12 and 13.

References

1. Ryan Alweiss, Brice Huang, and Mark Sellke. Improved Lower Bound for Frankl's Union-Closed Sets Conjecture. Available at `arXiv:2211.11731`, 2022.
2. A. Belovs and E. Blais. A polynomial lower bound for testing monotonicity. In *Proceedings of the 48th ACM Symposium on Theory of Computing*, 2016.
3. Hadley Black, Deeparnab Chakrabarty, and C. Seshadhri. A $o(d) \cdot$ polylog n Monotonicity Tester for Boolean Functions over the Hypergrid $[n]^d$. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 2133–2151. SIAM, 2018.

4 Hadley Black, Deeparnab Chakrabarty, and C. Seshadhri. Domain Reduction for Monotonicity Testing: A $o(d)$ Tester for Boolean Functions in d-Dimensions. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1975–1994. SIAM, 2020.

5 Hadley Black, Deeparnab Chakrabarty, and C. Seshadhri. Directed Isoperimetric Theorems for Boolean Functions on the Hypergrid and an $\tilde{O}(n\sqrt{d})$ Monotonicity Tester. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 233–241. ACM, 2023.

6 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47:549–595, 1993. Earlier version in STOC'90.

7 Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(84):929–965, October 1989.

8 Mark Braverman, Subhash Khot, Guy Kindler, and Dor Minzer. Improved monotonicity testers via hypercube embeddings. In *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPIcs*, pages 25:1–25:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023.

9 Jop Briët, Sourav Chakraborty, David García-Soriano, and Arie Matsliah. Monotonicity testing and shortest-path routing on the cube. *Comb.*, 32(1):35–53, 2012.

10 Henning Bruhn and Oliver Schaudt. The journey of the union-closed conjecture. *Graphs and Combinatorics*, 31:2043–2074, 2015. doi:10.1007/s00373-014-1515-0.

11 Deeparnab Chakrabarty and C. Seshadhri. A $o(n)$ monotonicity tester for boolean functions over the hypercube. In *Proceedings of the 45th ACM Symposium on Theory of Computing*, pages 411–418, 2013.

12 Deeparnab Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and lipschitz testing over hypercubes and hypergrids. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 419–428. ACM, 2013.

13 Deeparnab Chakrabarty and C. Seshadhri. Adaptive boolean monotonicity testing in total influence time. In *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*, volume 124 of *LIPIcs*, pages 20:1–20:7. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

14 Gilad Chase, Yuval Filmus, Dor Minzer, Elchanan Mossel, and Nitin Saurabh. Approximate polymorphisms. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20–24, 2022*, pages 195–202. ACM, 2022.

15 Zachary Chase and Shachar Lovett. Approximate union closed conjecture. Available at arXiv:2211.11689, 2022.

16 Xi Chen, Anindya De, Yuhao Li, Shivam Nadimpalli, and Rocco A. Servedio. Mildly exponential lower bounds on tolerant testers for monotonicity, unateness, and juntas. *SODA 2024*, To appear, 2024. doi:10.48550/arXiv.2309.12513.

17 Xi Chen, Anindya De, Rocco A. Servedio, and Li-Yang Tan. Boolean Function Monotonicity Testing Requires (Almost) $n^{1/2}$ Non-adaptive Queries. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015*, pages 519–528, 2015.

18 Xi Chen, Rocco A. Servedio, and Li-Yang Tan. New algorithms and lower bounds for testing monotonicity. In *Proceedings of the 55th IEEE Symposium on Foundations of Computer Science*, pages 286–295, 2014.

19 Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond Talagrand functions: new lower bounds for testing monotonicity and unateness. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 523–536, 2017.

20 Irit Dinur and Ehud Friedgut. Intersecting families are essentially contained in juntas. *Combinatorics, Probability and Computing*, 18(1-2):107–122, 2009. doi:10.1017/S0963548308009309.

21 Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved testing algorithms for monotonocity. In *Proceedings of RANDOM*, pages 97–108, 1999.

22 D. Ellis, N. Keller, and N. Lifshitz. Stability versions of Erdös-Ko-Rado type theorems, via isoperimetry. *J. Eur. Math. Soc*, 21:3857–3902, 2019.

23 David Ellis. *Intersection Problems in Extremal Combinatorics: Theorems, Techniques and Questions Old and New*, pages 115–173. Cambridge University Press, 2022.

24 P. Erdös, C. Ko, and R. Rado. Intersection theorems for systems of finite sets. *Quart. J. Math. Oxford (Series 2)*, 12:313–320, 1961.

25 Yuval Filmus, Noam Lifshitz, Dor Minzer, and Elchanan Mossel. AND testing and robust judgement aggregation. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 222–233. ACM, 2020.

26 E. Fischer, E. Lehman, I. Newman, S. Raskhodnikova, R. Rubinfeld, and A. Samorodnitsky. Monotonicity testing over general poset domains. In *Proc. 34th Annual ACM Symposium on the Theory of Computing*, pages 474–483, 2002.

27 Pŕer Frankl. Extremal set systems. In *Handbook of combinatorics*, pages 2:1293–1329, 1995.

28 E. Friedgut. On the measure of intersecting families, uniqueness and stability. *Combinatorica*, 28:503–528, 2008.

29 Justin Gilmer. A constant lower bound for the union-closed sets conjecture. Available at `arXiv:2211.09055`, 2022.

30 Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samordinsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

31 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45:653–750, 1998.

32 S. Halevy and E. Kushilevitz. Distribution-Free Property Testing. *SIAM J. Comput.*, 37(4):1107–1138, 2007.

33 Daniel M. Kane. A monotone function given by a low-depth decision tree that is not an approximate junta. *Theory Comput.*, 9:587–592, 2013.

34 Subhash Khot, Dor Minzer, and Muli Safra. On monotonicity testing and boolean isoperimetric-type theorems. *SIAM J. Comput.*, 47(6):2238–2276, 2018.

35 Philip N. Klein and Neal E. Young. On the number of iterations for dantzig-wolfe optimization and packing-covering approximation algorithms. *SIAM J. Comput.*, 44(4):1154–1172, 2015. `doi:10.1137/12087222X`.

36 Kevin Matulef, Ryan O'Donnell, Ronitt Rubinfeld, and Rocco A. Servedio. Testing halfspaces. *SIAM Journal on Computing*, 39(5):2004–2047, 2010.

37 Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Erik Waingarten. Approximating the distance to monotonicity of boolean functions. *Random Struct. Algorithms*, 60(2):233–260, 2022.

38 M. Parnas, D. Ron, and A. Samorodnitsky. Testing Basic Boolean Formulae. *SIAM J. Disc. Math.*, 16:20–46, 2002. URL: `https://citeseer.ifi.unizh.ch/parnas02testing.html`.

39 Luke Pebody. Extension of a Method of Gilmer. Available at `arXiv:2211.13139`, 2022.

40 Will Sawin. An improved lower bound for the union-closed set conjecture. Available at `arXiv:2211.11504`, 2022.

41 M. Talagrand. How much are increasing sets positively correlated? *Combinatorica*, 16(2):243–258, 1996.

42 A. Yao. Probabilistic computations: Towards a unified measure of complexity. In *Proc. Seventeenth Annual Symposium on Foundations of Computer Science (STOC)*, pages 222–227, 1977.