Testing for Concise Representations

Ilias Diakonikolas[*] Columbia University ilias@cs.columbia.edu

Homin K. Lee[†] Columbia University homin@cs.columbia.edu

Kevin Matulef[‡] MIT matulef@mit.edu

Krzysztof Onak[§] MIT konak@mit.edu

Ronitt Rubinfeld[¶] MIT ronitt@theory.csail.mit.edu

Rocco A. Servedio[‖] Columbia University rocco@cs.columbia.edu

Andrew Wan[**] Columbia University atw12@columbia.edu

Abstract

We describe a general method for testing whether a function on n input variables has a concise representation. The approach combines ideas from the junta test of Fischer et al. [6] with ideas from learning theory, and yields property testers that make poly(s/ϵ) queries (independent of n) for Boolean function classes such as s-term DNF formulas (answering a question posed by Parnas et al. [12]), size-s decision trees, size-s Boolean formulas, and size-s Boolean circuits.

The method can be applied to non-Boolean valued function classes as well. This is achieved via a generalization of the notion of variation *from Fischer et al. to non-Boolean functions. Using this generalization we extend the original junta test of Fischer et al. to work for non-Boolean functions, and give poly(s/ϵ)-query testing algorithms for non-Boolean valued function classes such as size-s algebraic circuits and s-sparse polynomials over finite fields.*

We also prove an $\tilde{\Omega}(\sqrt{s})$ query lower bound for nonadaptively testing s-sparse polynomials over finite fields of constant size. This shows that in some instances, our general method yields a property tester with query complexity that is optimal (for nonadaptive algorithms) up to a polynomial factor.

1. Introduction

Suppose you are given black-box access to a program computing an unknown function. You would like to gain some understanding of the program by querying it as few times as possible. A natural first question is whether the program has some sort of concise representation: is it representable by a small decision tree? a small DNF formula, Boolean circuit, or algebraic circuit? a sparse polynomial?

In this paper we study the problem of testing whether a function has a concise representation for various different types of representations, including those mentioned above. We work in the standard model of *property testing*. Namely, we assume that we have black-box query access to an unknown function f :

[*]Supported in part by NSF grant CCF-04-30946 and an Alexander S. Onassis Foundation Fellowship.

[†]Supported in part by NSF award CCF-0347282 and by NSF award CCF-0523664.

[‡]Supported in part by an NSF graduate fellowship.

[§]Supported in part by NSF grant 0514771.

[¶]Supported in part by NSF grant 0514771.

[‖]Supported in part by NSF award CCF-0347282, by NSF award CCF-0523664, and by a Sloan Foundation Fellowship.

[**]Supported in part by NSF award CCF-0347282 and by NSF award CCF-0523664.

$\Omega^n \to X$, and we are interested in algorithms that accept any function which has a concise representation of the desired type and reject any function which is ϵ-far from having such a representation (i.e. for every function f' which has such a representation, f and f' disagree on at least an ϵ fraction of inputs). As is standard in property testing, we assume that queries to the function are the limiting resource (rather than computation time), and we would like to obtain algorithms whose query complexity is independent of n, the number of inputs to the function.

Previous work on testing function classes. There has been considerable research on testing functions for various types of representations. Our work is most directly motivated by the paper of Parnas *et al.* [12], who gave algorithms for testing whether Boolean functions $f : \{0,1\}^n \to \{0,1\}$ have certain very simple representations as Boolean formulae. They gave an $O(1/\epsilon)$-query algorithm for testing whether f is a single Boolean literal or a Boolean conjunction, and an $\tilde{O}(s^2/\epsilon)$-query algorithm for testing whether f is an s-term monotone DNF. Parnas *et al.* posed as an open question whether a similar testing result can be obtained for the broader class of general (non-monotone) s-term DNF formulas.

Other closely related results include the following: An $O(1/\epsilon)$-query algorithm for testing whether a function can be represented as a linear form over a finite field is given in Blum *et al.* [2]. This algorithm was subsequently generalized in several works to test whether f can be represented as a low-degree polynomial. In particular, [1, 8, 9] consider the case when f is defined over a small finite field. Fischer *et al.* [6] gave an algorithm to test whether a Boolean function $f : \Omega^n \to \{0,1\}$ is a J-junta (i.e. depends only on at most J of its n arguments) with query complexity polynomial in J and $1/\epsilon$.

Other research in the area includes the work of Kearns and Ron [10], who gave testing algorithms for the classes of interval functions over the continuous interval $[0,1]$ and for neural networks and decision trees over the continuous cube $[0,1]^n$. Their results are not comparable to ours because they differ from the "standard" property testing results in several ways; for one thing, they view the dimension n as a constant and their algorithms have query complexity that depends (exponentially) on n.

Our Results. Our main result is a general algorithm that can be used to test whether an unknown function $f : \Omega^n \to X$ belongs to one of many different representation classes, as long as the representation class satisfies certain conditions. We show that this algorithm yields property testers for many classes that were not previously known to be testable. These include decision lists, size-s decision trees, size-s branching programs, s-term DNF (resolving the aforementioned open question of Parnas *et al.*), size-s Boolean formulas, size-s Boolean circuits, and s-sparse polynomials over $\mathbb{F}_2$.[1] For each of these classes the testing algorithm uses poly$(s, 1/\epsilon)$ many queries, independent of the number n of inputs to the function (the running time is exponential in s, though linear in n). These testing results are summarized in the top part of Table 1. We note that our general algorithm can also be easily shown to yield property testers for all of the classes tested in [12]; the query complexities would be slightly larger than in [12], but would not require a specialized algorithm for each problem.

Our second contribution is a generalization of the notion of *variation* given in [6] to functions with non-Boolean ranges. This generalization, and the properties we establish for the generalized variation, lets us extend the junta test of [6] to functions with non-Boolean ranges. It also allows us to use our general algorithm to achieve testers for non-Boolean valued function classes such as size-s algebraic circuits, size-s algebraic computation trees, and s-sparse polynomials over finite fields (see the bottom of Table 1).

Our third main contribution is a lower bound; we show that any non-adaptive algorithm to test s-sparse polynomials over finite fields of constant size must make $\tilde{\Omega}(\sqrt{s})$ queries. Since this is within a polynomial factor of our upper bound, this result shows that in at least one instance our general algorithm yields a tester that is nearly optimal. (For testing other representation classes, there is a larger gap between our upper

[1]We remind the reader that if $\mathcal{C}$ is a subclass of $\mathcal{C}'$, the existence of a testing algorithm for $\mathcal{C}'$ does *not* imply the existence of a testing algorithm for $\mathcal{C}$; thus, for example, our testing result for Boolean circuits does not imply the results for weaker representations such as Boolean formulas or DNF formulas.

Class of functions	Number of Queries	Reference
Boolean functions $f : \{0,1\}^n \to \{0,1\}$		
Boolean literals (dictators), conjunctions	$O(1/\epsilon)$	[12]
s-term monotone DNFs	$\tilde{O}(s^2/\epsilon)$	[12]
J-juntas	$\tilde{O}(J^2/\epsilon)$, $\Omega(J)$ (adaptive)	[6], [3]
decision lists	$\tilde{O}(1/\epsilon^2)$	this paper
size-s decision trees, size-s branching programs, s-term DNFs, size-s Boolean formulas	$\tilde{O}(s^4/\epsilon^2)$, $\Omega(\log s/\log\log s)$ (adaptive)	this paper
s-sparse polynomials over $\mathbb{F}_2$	$\tilde{O}(s^4/\epsilon^2)$, $\tilde{\Omega}(\sqrt{s})$	this paper
size-s Boolean circuits	$\tilde{O}(s^6/\epsilon^2)$	this paper
functions with Fourier degree $\leq d$	$\tilde{O}(2^{6d}/\epsilon^2)$, $\tilde{\Omega}(\sqrt{d})$	this paper
General functions $f : \Omega^n \to X$		
J-juntas	$\tilde{O}(J^2/\epsilon)$	this paper
s-sparse polynomials over field of size $\lvert\mathbb{F}\rvert$	$\tilde{O}((s\lvert\mathbb{F}\rvert)^4/\epsilon^2)$, $\tilde{\Omega}(\sqrt{s})$ for $\lvert\mathbb{F}\rvert = O(1)$	this paper
size-s algebraic circuits, size-s algebraic computation trees over $\mathbb{F}$	$\tilde{O}(s^4 \log^3 \lvert\mathbb{F}\rvert/\epsilon^2)$	this paper

Table 1. Selected previous results on testing function classes. Our upper bounds are for adaptive algorithms, though in all cases very similar bounds for non-adaptive algorithms can be achieved (see Appendix C). The lower bounds are for non-adaptive algorithms unless otherwise indicated by (adaptive).

and lower bounds. We give some simple but fairly weak lower bounds for other representation classes in Appendix E.)

Our techniques. Our approach combines ideas from the junta test of Fischer *et al.* [6] with ideas from learning theory. The basic idea of using a learning algorithm to do property testing goes back to Goldreich *et al.* [7]. They observed that any proper learning algorithm for a class $\mathcal{C}$ can immediately be used as a testing algorithm for $\mathcal{C}$. (If f belongs to $\mathcal{C}$, then a proper learning algorithm can be used to find a function $f' \in \mathcal{C}$ that f is $\epsilon/2$-close to, while if f is ϵ-far from $\mathcal{C}$ then clearly the proper learning algorithm cannot find any function $f' \in \mathcal{C}$ that f is even ϵ-close to.) However, it is well known that proper learning algorithms for virtually every interesting class of n-variable functions (such as all the classes listed in Table 1, including such simple classes as Boolean literals) must make at least $\Omega(\log n)$ queries. Thus this testing-by-learning approach did not previously yield any strong results for testing interesting function classes.

We get around this impediment by making the key observation that many interesting classes $\mathcal{C}$ of functions are "well-approximated" by juntas in the following sense: every function in $\mathcal{C}$ is close to some function in $\mathcal{C}_J$, where $\mathcal{C}_J \subseteq \mathcal{C}$ and every function in $\mathcal{C}_J$ is a J-junta. For example, every s-term DNF over $\{0,1\}^n$ is τ-close to an s-term DNF that depends on only $s \log s/\tau$ variables, since each term with more than $\log s/\tau$ variables can be removed from the DNF at the cost of at most τ/s error. Roughly speaking, our algorithm for testing whether f belongs to $\mathcal{C}$ works by attempting to learn the "structure" of the junta in $\mathcal{C}_J$ that f is close to *without actually identifying the relevant variables on which the junta depends*. If the algorithm finds such a junta function, it accepts, and if it does not, it rejects. Our approach can be characterized as *testing by implicit learning* (as opposed to the explicit proper learning in the approach of Goldreich *et al.* [7]), since we are "learning" the structure of the junta to which f is close without explicitly identifying its relevant variables. Indeed, avoiding identifying the relevant variables is what makes it possible to have query complexity independent of n.

We find the structure of the junta f' in $\mathcal{C}_J$ that f is close to by using the techniques of [6]. As in [6], we begin by randomly partitioning the variables of f into subsets and identifying which subsets contain an influential variable (the random partitioning ensures that with high probability, each subset contains at most one such variable if f is indeed in $\mathcal{C}$). Next, we create a sample of random labeled examples $(x^1, y^1), (x^2, y^2), ..., (x^m, y^m)$, where each x^i is a string of length J (not length n; this is crucial to the query complexity of the algorithm) whose bits correspond to the influential variables of f, and where y^i corresponds with high probability to the value of junta f' on x^i. Finally, we exhaustively check whether any function in $\mathcal{C}_J$ – over J input variables – is consistent with this labeled sample. This step takes at least $|\mathcal{C}_J|$ time steps, which is exponential in s for the classes in Table 1; but since $|\mathcal{C}_J|$ is independent of n we are able to get away with an overall query complexity that is independent of n. (The overall time complexity is linear as a function of n; note that such a runtime dependence on n is inevitable since it takes n time steps simply to prepare a length-n query string to the black-box function.) We explain our testing algorithm in more detail in Section 3.

In order to extend our testing results and the junta testing results in [6] to functions with non-Boolean ranges, we extend the technical definition of *variation* given in [6] to more general functions (intuitively, the variation is a measure of the ability of a set of variables to sway a function's output). We show that this extended definition has the necessary properties to carry the analysis of the junta tests and our test over to this more general setting. We present and analyze our extended definition of variation in Section 3.3.

Finally, we prove our lower bound for testing s-sparse polynomials over finite fields in two stages. We first show that any non-adaptive algorithm that can successfully distinguish a linear form $x_{i_1} + \cdots + x_{i_s}$ (over s randomly selected variables from $x_1, \ldots, x_n$) from a linear form $x_{i_1} + \cdots + x_{i_{s+p}}$ (over $s + p$ randomly selected variables, where p is the characteristic of the finite field) must make $\tilde{\Omega}(\sqrt{s})$ queries. This is a technical generalization of a similar result for $\mathbb{F}_2$ in [6]; the heart of our proof is an extension of a convergence type result about random walks over $\mathbb{Z}_2^q$ with arbitrary step distribution to random walks over $\mathbb{Z}_p^q$. (As an interesting side product, the latter also partially answers a question posed in [6] as to what groups possess a similar convergence type property.) We then prove that every s-sparse polynomial g over finite field $\mathbb{F}$ is "far" from every affine function with at least $s + 1$ non-zero coefficients. This result does not have an analogue in [6] (that paper establishes a lower bound on distinguishing size-s parities from size-$(s + 2)$ parities, and it is trivially true that every size-s parity is far from every size-$(s + 2)$ parity) and its proof requires several ideas; our argument uses random restrictions chosen according to a distribution that depends on the structure of the polynomial g. We present these results in Section 4.

2. Preliminaries

For $i \in \mathbb{N}$, we denote $[i] \stackrel{\text{def}}{=} \{1, 2, \ldots, i\}$. Throughout the paper, Ω denotes an arbitrary finite set and X denotes an arbitrary finite range set. We will be interested in functions f that map from Ω^n to X. In keeping with the notation of Fischer *et al.* [6] we sometimes write $\mathcal{P}([n])$ to denote the domain Ω^n, and we write $x = (x_1, \ldots, x_n)$ to denote an element of the domain $\mathcal{P}([n])$. An important special case for many of the applications of our main result, discussed in Appendix D.1, is when f is a Boolean function over the Boolean hypercube, i.e. $\Omega = \{0, 1\}^n$ and $X = \{-1, 1\}$.

We view the domain $\mathcal{P}([n])$ as endowed with the uniform probability measure. Two functions $f_1, f_2 : \mathcal{P}([n]) \to X$ are said to be *ϵ-close* if $\Pr[f_1(x) \neq f_2(x)] \leq \epsilon$, and are *$\epsilon$-far* if $\Pr[f_1(x) \neq f_2(x)] > \epsilon$. We write $\mathbb{E}$ to denote expectation and $\mathbb{V}$ to denote variance.

Let $f : \mathcal{P}([n]) \to X$ be a function and let $I \subseteq [n]$ be a subset of the input coordinates. We define $\mathcal{P}(I)$ to be the set of all partial assignments to the input coordinates x_i for $i \in I$. Thus $\mathcal{P}([n]) = \Omega^n$ is the entire domain of all input vectors of length n. For $w \in \mathcal{P}([n] \setminus I)$ and $z \in \mathcal{P}(I)$, we write $w \sqcup z$ to denote the assignment whose i-th coordinate is w_i if $i \in [n] \setminus I$ and is z_i if $i \in I$.

A function $f : \mathcal{P}([n]) \to X$ is said to be a J-*junta* if there exists a set $\mathcal{J} \subseteq [n]$ of size at most J such that $f(x) = f(y)$ for every two assignments $x, y \in \mathcal{P}([n])$ that agree on $\mathcal{J}$.

Let S be a finite set and $\mathbb{P}, \mathbb{Q}$ be probability measures on it. The *statistical distance* between $\mathbb{P}$ and $\mathbb{Q}$ is defined by $\|\mathbb{P} - \mathbb{Q}\| \stackrel{\text{def}}{=} \max_{A \subseteq S} |\mathbb{P}(A) - \mathbb{Q}(A)|$.

3. The test and an overview of its analysis

In this section we present our testing algorithm and give an intuitive explanation of how it works. We close this section with a detailed statement of our main theorem, Theorem 4, describing the correctness and query complexity of the algorithm.

3.1. Subclass approximators.

Let $\mathcal{C}$ denote a class of functions from $\mathcal{P}([n])$ to X. We will be interested in classes of functions that can be closely approximated by juntas in the class. We have the following:

Definition 1. *For $\tau > 0$, we say that a subclass $\mathcal{C}(\tau) \subseteq \mathcal{C}$ is a $(\tau, J(\tau))$*-approximator *for $\mathcal{C}$ if*

- *$\mathcal{C}(\tau)$ is closed under permutation of variables, i.e. if $f(x_1, \ldots, x_n) \in \mathcal{C}(\tau)$ then $f(x_{\sigma_1}, \ldots, x_{\sigma_n})$ is also in $\mathcal{C}(\tau)$ for every permutation σ of $[n]$; and*
- *for every function $f \in \mathcal{C}$, there is a function $f' \in \mathcal{C}(\tau)$ such that f' is τ-close to f and f' is a $J(\tau)$-junta.*

Typically for us $\mathcal{C}$ will be a class of functions with size bound s in some particular representation, and $J(\tau)$ will depend on s and τ. (A good running example to keep in mind is $\Omega = \{0, 1\}$, $X = \{-1, 1\}$, and $\mathcal{C}$ is the class of all functions that have s-term DNF representations. In this case we may take $\mathcal{C}(\tau)$ to be the class of all s-term $\log(s/\tau)$-DNFs, and we have $J(\tau) = s \log(s/\tau)$.) Our techniques will work on function classes $\mathcal{C}$ for which $J(\tau)$ is a slowly growing function of $1/\tau$ such as $\log(1/\tau)$. In Section 3.7 we will consider many different specific instantiations of $\mathcal{C}$ and corresponding choices of $\mathcal{C}(\tau)$.

We write $\mathcal{C}(\tau)_k$ to denote the subclass of $\mathcal{C}(\tau)$ consisting of those functions that depend only on variables in $\{x_1, \ldots, x_k\}$. We may (and will) view functions in $\mathcal{C}(\tau)_k$ as taking k arguments from Ω rather than n.

3.2. The independence test.

An important sub-test that will be used throughout the main test is the independence test from [6].

Independence test: Given a function f, and a set of variables I, choose $w \in_{\mathrm{R}} \mathcal{P}([n] \setminus I)$ and $z_1, z_2 \in_{\mathrm{R}} \mathcal{P}(I)$. Accept if $f(w \sqcup z_1) = f(w \sqcup z_2)$ and reject if $f(w \sqcup z_1) \neq f(w \sqcup z_2)$.

If f is independent of the coordinates in I, the independence test always accepts. On the other hand, intuitively if I contains highly relevant variables that are likely to sway the output of f, the independence test is likely to reject.

3.3. Extended variation and testing juntas with non-Boolean ranges.

Fischer *et al.* [6] defined the notion of the *variation* of a function on a subset of input variables. The variation is a measure of the extent to which the function is sensitive to the values of the variables in the set. Let us recall their definition of variation.

Definition 2. *Let f be a function from $\mathcal{P}([n])$ to $\{-1,1\}$, and let $I \subseteq [n]$ be a subset of coordinates. We define the* variation *of f on I as*

$$\mathrm{Vr}_f(I) \stackrel{\text{def}}{=} \mathbb{E}_{w \in \mathcal{P}([n]\setminus I)} \left[\mathbb{V}_{z \in \mathcal{P}(I)} \left[f(w \sqcup z) \right] \right]. \tag{1}$$

Fischer *et al.* showed that the variation is monotone and sub-additive; that for a subset I of the variables, the probability that the independence test rejects is exactly $\frac{1}{2}\mathrm{Vr}_f(I)$; and that if $\mathrm{Vr}_f(I) \leq 2\epsilon$ then f is ϵ-close to a function which does not depend on the variables in I. The analysis of their junta tests depends crucially on these properties of variation.

Unfortunately, the variation properties stated above do not always hold for functions with non-Boolean range, and the original analysis of the junta test does not carry over to the non-Boolean setting. Intuitively, however, the fact that a function may take on more than two values should not make the junta test incorrect. The independence test, which is the main component of the junta test, only checks if values of the function are equal or different. Can one modify the definition of variation and the analysis of the junta test so that the non-Boolean case is captured too?

An approach that we manage to successfully apply is mapping the function range to the Boolean range. The general idea is to pick a mapping from the function range X to the set $\{-1,1\}$ that preserves as much of the sensitivity of the function as possible. If we look at Equation 1 defining variation, we could choose the best mapping to $\{-1,1\}$ either before or after the expectation operator. It turns out that depending on the context, one or the other is more suitable, so we define and use both. Denote by $\mathcal{F}(X)$ the set of all functions from X to $\{-1,1\}$.

Definition 3. *Let f be a function from $\mathcal{P}([n])$ to X, and let $I \subseteq [n]$ be a subset of coordinates. We define the* binary variation *of f on I as*

$$\mathrm{BinVr}_f(I) \stackrel{\text{def}}{=} \max_{g \in \mathcal{F}(X)} \mathrm{Vr}_{g \circ f}(I) = \max_{g \in \mathcal{F}(X)} \mathbb{E}_{w \in \mathcal{P}([n]\setminus I)} \left[\mathbb{V}_{z \in \mathcal{P}(I)} \left[g(f(w \sqcup z)) \right] \right],$$

and the extreme variation *of f on I as*

$$\mathrm{ExtVr}_f(I) \stackrel{\text{def}}{=} \mathbb{E}_{w \in \mathcal{P}([n]\setminus I)} \left[\max_{g \in \mathcal{F}(X)} \mathbb{V}_{z \in \mathcal{P}(I)} \left[g(f(w \sqcup z)) \right] \right].$$

To be able to use both new notions of variation, we need to show that they are related. Probabilistic analysis shows that these two quantities are always within a factor of 4 of each other:

$$\frac{1}{4}\mathrm{ExtVr}_f(I) \leq \mathrm{BinVr}_f(I) \leq \mathrm{ExtVr}_f(I).$$

In Appendix A, we prove that the *binary variation* has almost identical properties to the original variation. Namely, we show that the binary variation is also monotone and sub-additive; that the independence test rejects with probability at least $\frac{1}{2}\mathrm{BinVr}_f(I)$; and that if $\mathrm{BinVr}_f(I) \leq \epsilon/4$ for some subset I of the variables of f then f is ϵ-close to a function that does not depend on I. Furthermore, in Appendix A.6 we explain how these properties imply that the three junta tests given by Fischer *et al.* essentially work for functions with non-Boolean ranges as well (with minor modifications). Indeed, the first step of our general testing algorithm $\mathcal{A}$ is essentially the junta test of Fischer *et al.* modified to apply to non-Boolean valued functions. We carefully analyze this first step in Appendix B.1; the results there are easily seen to imply that this first step gives an $\tilde{O}(J^2/\epsilon)$-query junta test for non-Boolean functions as claimed in Table 1.

Identify-Critical-Subsets (input is black-box access to $f : \Omega^n \to X$ and $\epsilon > 0$)

1. Partition the variables $x_1, \dots, x_n$ into r random subsets by assigning each of $x_1, \dots, x_n$ equiprobably to one of $I_1, \dots, I_r$.
2. Choose s random subsets $\Lambda_1, \dots, \Lambda_s \subseteq [r]$ of size $J(\tau^\star)$ by uniformly choosing without repetitions $J(\tau^\star)$ members of $[r]$. Each set Λ_i determines a block $B_i \stackrel{\text{def}}{=} \bigcup_{j \in \Lambda_i} I_j$. (Note that we do not guarantee that the blocks are disjoint.)
3. Apply h iterations of the *independence test* (see Section 3.2) to each block B_i. If all of the independence test iterations applied to block B_i accept, then B_i is declared to be *variation-free*, and all the subsets I_j with $j \in \Lambda_i$ are declared to be variation-free on its behalf.
4. If:
   (a) at least half of the blocks $B_1, \dots, B_s$ are variation-free; and
   (b) except for at most $J(\tau^\star)$ subsets, every subset in the partition $I_1, \dots, I_r$ is declared variation-free on behalf of some block,

   then output the list $I_{i_1}, \dots, I_{i_j}$ of those subsets that are *not* declared to be variation-free. (We call these the *critical* subsets.) Otherwise, halt and output "Not in $\mathcal{C}$."

Figure 1. The subroutine Identify-Critical-Subsets.

3.4. Explanation of our testing algorithm.

Our algorithm for testing whether a function $f : \mathcal{P}([n]) \to X$ belongs to $\mathcal{C}$ or is ϵ-far from $\mathcal{C}$ is given in Figures 1 through 3. Given $\epsilon > 0$ and black-box access to f, the algorithm performs three main steps:

1. **Identify critical subsets.** In Step 1, we first randomly partition the variables $x_1, \dots, x_n$ into r disjoint subsets $I_1, \dots, I_r$. We then attempt to identify a set of $j \le J(\tau^\star)$ of these r subsets, which we refer to as *critical* subsets because they each contain a "highly relevant" variable. (For now the value $\tau^\star$ should be thought of as a small quantity; we discuss how this value is selected below.) This step is essentially the same as the 2-sided test for J-juntas from Section 4.2 of Fischer *et al.* [6]. We will show that if f is close to a $J(\tau^\star)$-junta then this step will succeed w.h.p., and if f is far from every $J(\tau^\star)$-junta then this step will fail w.h.p.
2. **Construct a sample.** Let $I_{i_1}, \dots, I_{i_j}$ be the critical subsets identified in the previous step. In Step 2 we construct a set S of m labeled examples $\{(x^1, y^1), \dots, (x^m, y^m)\}$, where each x^i is independent and uniformly distributed over $\Omega^{J(\tau^\star)}$. We will show that if f belongs to $\mathcal{C}$, then with high probability there is a fixed $f'' \in \mathcal{C}(\tau^\star)_{J(\tau^\star)}$ such that each y^i is equal to $f''(x^i)$. On the other hand, if f is far from $\mathcal{C}$, then we will show that w.h.p. no such $f'' \in \mathcal{C}(\tau^\star)_{J(\tau^\star)}$ exists.

   To construct each labeled example, we again borrow a technique outlined in [6]. We start with a uniformly random $z \in \Omega^n$. We then attempt to determine how the j highly relevant coordinates of z are set. Although we don't know which of the coordinates of z are highly relevant, we do know that, assuming the previous step was successful, there should be one highly relevant coordinate in each of the critical subsets. We use the independence test repeatedly to determine the setting of the highly relevant coordinate in each critical subset.

   For example, suppose that $\Omega = \{0, 1\}$ and I_1 is a critical subset. To determine the setting of the highly relevant coordinate of z in critical subset I_1, we subdivide I_1 into two sets: the subset $\Omega_0 \subseteq I_1$ of

Construct-Sample (input is the list $I_{i_1}, \ldots, I_{i_j}$ output by **Identify-Critical-Subsets** and black-box access to f)

1. Repeat the following m times to construct a set S of m labeled examples $(x, y) \in \Omega^{J(\tau^\star)} \times X$, where $\Omega = \{\omega_0, \omega_1, \ldots, \omega_{|\Omega|-1}\}$:

   (a) Draw z uniformly from Ω^n. Let $X_q \stackrel{\text{def}}{=} \{i : z_i = \omega_q\}$, for each $0 \leq q \leq |\Omega| - 1$.

   (b) For $\ell = 1, \ldots, j$

      i. $w \stackrel{\text{def}}{=} 0$

      ii. For $k = 1, \ldots, \lceil \lg |\Omega| \rceil$

         A. $\Omega_0 \stackrel{\text{def}}{=}$ union of $(X_q \cap I_{i_\ell})$ taken over all $0 \leq q \leq |\Omega| - 1$ such that the k-th bit of q is zero

         B. $\Omega_1 \stackrel{\text{def}}{=}$ union of $(X_q \cap I_{i_\ell})$ taken over all $0 \leq q \leq |\Omega| - 1$ such that the k-th bit of q is one

         C. Apply g iterations of the *independence test* to Ω_0. If any of the g iterations reject, mark Ω_0. Similarly, apply g iterations of the *independence test* to Ω_1; if any of the g iterations reject, mark Ω_1.

         D. If exactly one of Ω_0, Ω_1 (say Ω_b) is marked, set the k-th bit of w to b.

         E. If neither of Ω_0, Ω_1 is marked, set the k-th bit of w to unspecified.

         F. If both Ω_0, Ω_1 are marked, halt and output "no".

      iii. If any bit of w is unspecified, choose w at random from $\{0, 1, \ldots, |\Omega| - 1\}$.

      iv. If $w \notin [0, |\Omega| - 1]$, halt and output "no."

      v. Set $x_\ell = \omega_w$.

   (c) Evaluate f on z, assign the remaining $J(\tau^\star) - j$ coordinates of x randomly, and add the pair $(x, f(z))$ to the sample of labeled examples being constructed.

Figure 2. The subroutine Construct-Sample.

Check-Consistency (input is the sample S output by **Identify-Critical-Subsets**)

1. Check every function in $\mathcal{C}(\tau^\star)_{J(\tau^\star)}$ to see if any of them are consistent with sample S. If so output "yes" and otherwise output "no."

Figure 3. The subroutine Check-Consistency.

indices where z is set to 0, and the subset $\Omega_1 = I_1 \backslash \Omega_0$ of indices where z is set to 1. We can then use the independence test on both Ω_0 and Ω_1 to find out which one contains the highly relevant variable. This tells us whether the highly relevant coordinate of z in subset I_1 is set to 0 or 1. We repeat this process for each critical subset in order to find the settings of the j highly relevant coordinates of z; these form the string x. (The other $J(\tau^\star) - j$ coordinates of x are set to random values; intuitively, this is okay since they are essentially irrelevant.) We then output $(x, f(z))$ as the labeled example.

3. **Check consistency.** Finally, in Step 3 we search through $\mathcal{C}(\tau^\star)_{J(\tau^\star)}$ looking for a function f'' over $\Omega^{J(\tau^\star)}$ that is consistent with all m examples in S. (Note that this step takes $\Omega(|\mathcal{C}(\tau^\star)_{J(\tau^\star)}|)$ time but uses no queries.) If we find such a function then we accept f, otherwise we reject.

3.5. Sketch of the analysis.

We now give an intuitive explanation of the analysis of the test.

Completeness. Suppose f is in $\mathcal{C}$. Then there is some $f' \in \mathcal{C}(\tau^\star)$ that is $\tau^\star$-close to f. Intuitively, $\tau^\star$-close is so close that for the entire execution of the testing algorithm, the black-box function f might as well actually be f' (the algorithm only performs $\ll 1/\tau^\star$ many queries in total, each on a uniform random string, so w.h.p. the view of the algorithm will be the same whether the target is f or f'). Thus, for the rest of this intuitive explanation of completeness, we pretend that the black-box function is f'.

Recall that the function f' is a $J(\tau^\star)$-junta. Let us refer to the variables, x_i, that have $\mathrm{BinVr}_f(x_i) > \theta$ (recall that $\mathrm{BinVr}_f(x_i)$ is a measure of the influence of variable x_i, and θ is some threshold to be defined later) as the *highly relevant* variables of f'. Since f' is a junta, in Step 1 we will be able to identify a collection of $j \leq J(\tau^\star)$ "critical subsets" with high probability. Intuitively, these subsets have the property that:

- each highly relevant variable occurs in one of the critical subsets, and each critical subset contains at most one highly relevant variable (in fact at most one relevant variable for f');
- the variables outside the critical subsets are so "irrelevant" that w.h.p. in all the queries the algorithm makes, it doesn't matter how those variables are set (randomly flipping the values of these variables would not change the value of f' w.h.p.).

Given critical subsets from Step 1 that satisfy the above properties, in Step 2 we construct a sample of labeled examples $S = \{(x^1, y^1), \ldots, (x^m, y^m)\}$ where each x^i is independent and uniform over $\Omega^{J(\tau^\star)}$. We show that w.h.p. there is a $J(\tau^\star)$-junta $f'' \in C(\tau^\star)_{J(\tau^\star)}$ with the following properties:

- there is a permutation $\sigma : [n] \to [n]$ for which $f''(x_{\sigma(1)}, \ldots, x_{\sigma(J(\tau))})$ is close to $f'(x_1, \ldots, x_n)$;
- The sample S is labeled according to f''.

Finally, in Step 3 we do a brute-force search over all of $\mathcal{C}(\tau^\star)_{J(\tau^\star)}$ to see if there is a function consistent with S. Since f'' is such a function, the search will succeed and we output "yes" with high probability overall.

Soundness. Suppose now that f is ϵ-far from $\mathcal{C}$.

One possibility is that f is ϵ-far from every $J(\tau^\star)$-junta; if this is the case then w.h.p. the test will output "no" in Step 1.

The other possibility is that f is ϵ-close to a $J(\tau^\star)$-junta f' (or is itself such a junta). Suppose that this is the case and that the testing algorithm reaches Step 2. In Step 2, the algorithm tries to construct a set of labeled examples that is consistent with f'. The algorithm may fail to construct a sample at all; if this happens then it outputs "no." If the algorithm succeeds in constructing a sample S, then w.h.p. this sample is indeed consistent with f'; but in this case, w.h.p. in Step 3 the algorithm will not find any function $g \in \mathcal{C}(\tau^\star)_{J(\tau^\star)}$ that is consistent with all the examples. (If there were such a function g, then standard arguments in learning theory show that w.h.p. any such function $g \in \mathcal{C}(\tau^\star)_{J(\tau^\star)}$ that is consistent with S is in fact close to f'. Since f' is in turn close to f, this would mean that g is close to f. But g belongs to $\mathcal{C}(\tau^\star)_{J(\tau^\star)}$ and hence to $\mathcal{C}$, so this violates the assumption that f is ϵ-far from $\mathcal{C}$.)

3.6. The main theorem.

We now state our main theorem, which is proved in detail in Appendix B. The algorithm $\mathcal{A}$ is adaptive, but in Appendix C we discuss how to make it non-adaptive with only a slight increase in query complexity.

Theorem 4. *There is an algorithm $\mathcal{A}$ with the following properties:*

Let $\mathcal{C}$ be a class of functions from Ω^n to X. Suppose that for every $\tau > 0$, $\mathcal{C}(\tau) \subseteq \mathcal{C}$ is a $(\tau, J(\tau))$-approximator for $\mathcal{C}$. Suppose moreover that for every $\epsilon > 0$, there is a τ satisfying

$$\tau \leq \kappa\epsilon^2 \cdot [\ln(|\Omega|) \cdot J(\tau)^2 \cdot \ln^2(J(\tau)) \cdot \ln^2(|\mathcal{C}(\tau)_{J(\tau)}|) \cdot \ln\ln(J(\tau)) \cdot \ln(\tfrac{\ln(|\Omega|)}{\epsilon} \ln|\mathcal{C}(\tau)_{J(\tau)}|)]^{-1}, \quad (2)$$

where $\kappa > 0$ is a fixed absolute constant. Let $\tau^\star$ be the largest value τ satisfying (2) above. Then algorithm $\mathcal{A}$ makes

$$\tilde{O}\left(\frac{\ln|\Omega|}{\epsilon^2} J(\tau^\star)^2 \ln^2(|\mathcal{C}(\tau^\star)_{J(\tau^\star)}|)\right)$$

many black-box queries to f, and satisfies the following:

- *If $f \in \mathcal{C}$ then $\mathcal{A}$ outputs "yes" with probability at least $2/3$;*
- *If f is ϵ-far from $\mathcal{C}$ then $\mathcal{A}$ outputs "no" with probability at least $2/3$.*

Here are some observations to help interpret the bound (2). Note that if $J(\tau)$ grows too rapidly as a function of $1/\tau$, e.g. $J(\tau) = \Omega(1/\sqrt{\tau})$, then there will be no $\tau > 0$ satisfying inequality (2). On the other hand, if $J(\tau)$ grows slowly as a function of $1/\tau$, e.g. $\log(1/\tau)$, then it is may be possible to satisfy (2).

In all of our applications $J(\tau)$ will grow as $O(\log(1/\tau))$, and $\ln|C(\tau)_{J(\tau)}|$ will always be at most $\mathrm{poly}(J(\tau))$, so (2) will always be satisfiable. The most typical case for us will be that $J(\tau) \leq \mathrm{poly}(s)\log(1/\tau)$ (where s is a size parameter for the class of functions in question) and $\ln|C(\tau)_{J(\tau)}| \leq \mathrm{poly}(s) \cdot \mathrm{poly}\log(1/\tau)$, which yields $\tau^\star = \tilde{O}(\epsilon^2)/\mathrm{poly}(s)$ and an overall query bound of $\mathrm{poly}(s)/\tilde{O}(\epsilon^2)$.

3.7. Applications to Boolean and Non-Boolean Functions

Theorem 4 can be used to achieve testing algorithms, in most cases polynomial-query ones, for a wide range of natural and well-studied classes of Boolean functions over the n-dimensional Boolean hypercube (i.e. $\Omega = \{0, 1\}$ and $X = \{-1, 1\}$), such as s-term DNF. We use Theorem 4 to achieve testing algorithms for several interesting classes of non-Boolean functions as well. These testing results are noted in Table 1; we give detailed statements and proofs of these results in Appendix D.

4. Lower bounds for testing sparse polynomials.

One consequence of Theorem 4 is a poly(s/ϵ)-query algorithm for testing s-sparse polynomials over finite fields of fixed size (independent of n). In this section we present a polynomial lower bound for non-adaptive algorithms for this testing problem. (Detailed proofs for all results in this section are given in Appendix E.)

Theorem 5. *Let $\mathbb{F}$ be any fixed finite field, i.e. $|\mathbb{F}| = O(1)$ independent of n. There exists a fixed constant $\epsilon > 0$ (depending on $|\mathbb{F}|$) such that any* non-adaptive *ϵ-testing algorithm for the class of s-sparse polynomials over $\mathbb{F}^n$ must make $\tilde{\Omega}(\sqrt{s})$ queries.*

To prove Theorem 5 we use Yao's principle [16] in (what has become) a standard way for proving lower bounds in property testing (e.g. see [5]). We present two distributions D_{YES} and D_{NO}, the former on inputs satisfying the property (i.e. s-sparse polynomials from $\mathbb{F}^n$ to $\mathbb{F}$), the latter on inputs that are ϵ-far from satisfying it, and show that any deterministic (non-adaptive) algorithm making "few" queries cannot distinguish between a random draw from D_{YES} versus a random draw from D_{NO}. By standard arguments (see for example Lemma 8.1 in [5]), it suffices to argue that for any query set $\mathcal{Q} \subset \mathbb{F}^n$ of cardinality $q = \tilde{O}(\sqrt{s})$ the induced distributions on $\mathbb{F}^q$ (obtained by restricting the randomly chosen functions to these q points) have statistical distance less than $1/3$.

We define both D_{YES} and D_{NO} to be distributions over linear forms from $\mathbb{F}^n$ to $\mathbb{F}$. A random function from D_{YES} is obtained by independently and uniformly (with repetitions) picking s variables from $x_1, \ldots, x_n$ and taking their sum. D_{NO} is defined in the same way, but instead we pick $s+p$ variables, where p is the characteristic of the field $\mathbb{F}$. Clearly, every draw from D_{YES} is an s-sparse polynomial over $\mathbb{F}$, and for $n = \omega((s+p)^2)$, the birthday paradox implies that almost all the probability mass of D_{NO} is on functions with $s+p$ distinct nonzero coefficients. We claim that, for any set of $q = \tilde{O}(\sqrt{s})$ points in $\mathbb{F}^n$, the corresponding induced distributions have statistical distance less than $1/3$.

Let $(G, +)$ be a finite group. A probability measure $\mathbb{P}$ on G induces a random walk on G as follows: Denoting by X_n its position at time n, the walk starts at the identity element and at each step selects an element $\xi_n \in G$ according to $\mathbb{P}$ and goes to $X_{n+1} = \xi_n + X_n$. By arguments parallel to those in Section 6 of [6], the aforementioned claim can be reduced to the following theorem about random walks over $\mathbb{Z}_r^q$, which we prove in Section E.1.2:

Theorem 6. *Let r be a prime, $q \in \mathbb{N}^*$ and $\mathbb{P}$ be a probability measure on $\mathbb{Z}_r^q$. Consider the random walk X on $\mathbb{Z}_r^q$ with step distribution $\mathbb{P}$. Let $\mathbb{P}_t$ be the distribution of X at step t. There exists an absolute constant $C > 0$ such that for every $0 < \delta \leq 1/2$, if $t \geq C\frac{\log 1/\delta}{\delta} \cdot r^4 \log r \cdot q^2 \log^2(q+1)$ then $\|\mathbb{P}_t - \mathbb{P}_{t+r}\| \leq \delta$.*

Theorem 6 is a non-trivial generalization of a similar result proved in [6] for the special case $r = 2$. We now give a high-level overview of the overall strategy. Any given $x \in (\mathbb{Z}_r^q)^*$ partitions the space into r non-empty subspaces $V_i^x = \{y \in \mathbb{Z}_r^q : \langle y, x \rangle = i\}$ for $i = 0, 1, \ldots, r-1$. We say that an $x \in (\mathbb{Z}_r^q)^*$ is *degenerate* if there exists some i whose probability measure $\mathbb{P}(V_i^x)$ is "large". We consider two cases: If all the Fourier coefficients of $\mathbb{P}$ are not "very large", then we can show by standard arguments that the walk is close to its stationary (uniform) distribution after the desired number of steps. If, on the other hand, there exists a "very large" Fourier coefficient, then we argue that there must also exist a degenerate direction and we use induction on q.

So far we have shown that any algorithm that can successfully distinguish a random linear form $x_{i_1} + \cdots + x_{i_s}$ from a random linear form $x_{i_1} + \cdots + x_{i_{s+p}}$ must make $\tilde{\Omega}(\sqrt{s})$ queries. To complete the proof of Theorem 5, we must show that every s-sparse polynomial over $\mathbb{F}^n$ is "far" from every linear function of the form $x_{i_1} + \cdots + x_{i_{s+p}}$. We do this via the following new theorem (stated and proved in more detail as Theorem 36 in Appendix E), which may be of independent interest:

Theorem 7. *There exists a function $\epsilon = \epsilon(|\mathbb{F}|)$ such that for any $g : \mathbb{F}^n \to \mathbb{F}$ that is an s-sparse polynomial with $s \leq n-1$, g is ϵ-far from every affine function with at least $s+1$ non-zero coefficients.*

The high-level idea of the proof of Theorem 7 is as follows. Let M be a particular monomial in g, and consider what happens when g is hit with a restriction that fixes all variables that do not occur in M. M itself is not affected by the restriction, but it is possible for a longer monomial to "collapse" onto M and obliterate it (i.e. if M is $x_1x_2^2$ and g contains another monomial $M' = -x_1x_2^2x_3^3$, then a restriction that fixes $x_3 \leftarrow 1$ would cause M' to collapse onto M and in fact obliterate M). We show that g must have a short monomial M (which, however, has degree at least 2) with the following property: for a constant fraction of all possible restrictions of variables not in M, no longer monomial collapses onto M. This implies that for a constant fraction of all such restrictions ρ, the induced polynomial g_ρ is "substantially" different from any affine function (since g_ρ – a polynomial of degree at least two – is not identical to any affine function, it must be "substantially" different since there are only length(M) surviving variables), and hence g itself must be "far" from any affine function.

Lower bounds for other function classes. By adapting techniques of Chockler and Gutfreund [3], we can also obtain $\tilde{\Omega}(\log s)$ lower bounds for many of the other testing problems listed in Table 1. We state and prove these lower bounds at the end of Appendix E.

5. Conclusion

Our positive results are all achieved via a single generic algorithm that is not geared toward any particular class of functions. For many classes of interest, the query complexity of this algorithm is $\mathrm{poly}(s, 1/\epsilon)$, but the running time is exponential in s. It would be interesting to study algorithms that are more specifically tailored for classes such as s-term DNF, size-s Boolean formulas, etc. with the aim of obtaining poly(s) runtimes.

One approach to achieving better runtimes is to replace our "implicit learning" step with a more efficient proper learning algorithm (the current learning algorithm simply gathers random examples and exhaustively checks for a consistent hypothesis in the concept class $\mathcal{C}(\tau^\star)_{J(\tau^\star)}$). For some specific concept classes, proper learning is known to be NP-hard, but for other classes, the complexity of proper learning is unknown. The existence of a time-efficient proper learning algorithm for some specific class $\mathcal{C}(\tau^\star)_{J(\tau^\star)}$ would likely yield a time-efficient test in our framework.

Another goal for future work is to strengthen our lower bounds; can poly(s) query lower bounds be obtained for classes such as size-s decision trees, s-term DNF, etc?

References

[1] N. Alon, T. Kaufman, M. Krivelevich, S. Litsyn, and D. Ron. Testing low-degree polynomials over GF(2). In *Proceedings of RANDOM-APPROX*, pages 188–199, 2003.

[2] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *J. Comp. Sys. Sci.*, 47:549–595, 1993. Earlier version in STOC'90.

[3] H. Chockler and D. Gutfreund. A lower bound for testing juntas. *Information Processing Letters*, 90(6):301–305, 2004.

[4] P. Diaconis. *Group Representations in Probability and Statistics*. Institute of Mathematical Statistics, Hayward, CA, 1988.

[5] E. Fischer. The art of uninformed decisions: A primer to property testing. *Computational Complexity Column of The Bulletin of the European Association for Theoretical Computer Science*, 75:97–126, 2001.

[6] E. Fischer, G. Kindler, D. Ron, S. Safra, and A. Samorodnitsky. Testing juntas. *Journal of Computer & System Sciences*, 68:753–787, 2004.

[7] O. Goldreich, S. Goldwaser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45:653–750, 1998.

[8] C. Jutla, A. Patthak, A. Rudra, and D. Zuckerman. Testing low-degree polynomials over prime fields. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS '04)*, pages 423–432, 2004.

[9] T. Kaufman and D. Ron. Testing polynomials over general fields. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS '04)*, pages 413–422, 2004.

[10] M. Kearns and D. Ron. Testing problems with sub-learning sample complexity. *J. Comp. Sys. Sci.*, 61:428–456, 2000.

[11] N. Nisan and M. Szegedy. On the degree of Boolean functions as real polynomials. In *Proceedings of the Twenty-Fourth Annual Symposium on Theory of Computing*, pages 462–467, 1992.

[12] M. Parnas, D. Ron, and A. Samorodnitsky. Testing basic boolean formulae. *SIAM J. Disc. Math.*, 16:20–46, 2002.

[13] D. Štefankovič. Fourier transform in computer science. Master's thesis, University of Chicago, 2000.

[14] A. Terras. *Fourier Analysis on Finite Groups and Applications*. Cambridge University Press, Cambridge, UK, 1999.

[15] K. Verbeurgt. Learning DNF under the uniform distribution in quasi-polynomial time. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 314–326, 1990.

[16] A. Yao. Probabilistic computations: Towards a unified measure of complexity. In *Proceedings of the Seventeenth Annual Symposium on Foundations of Computer Science*, pages 222–227, 1977.

A. Variation and testing juntas for non-Boolean ranges

For a random variable X, we write $\mathbb{E}[X]$ to denote its expectation and $\mathbb{V}[X]$ to denote its variance. We write $\mathcal{F}(X)$ to denote the set of all functions from X to $\{-1, 1\}$.

A.1. The original variation notion.

In the paper of Fischer *et al.* [6] on testing juntas, the notion of variation played a central role in the proof of correctness of their algorithms. Unfortunately, their definition of variation only works for functions with Boolean range. We will redefine the notion of variation so that it works for non-Boolean ranges, and we will argue that the tests by Fischer *et al.* indeed work for non-Boolean ranges, with the only difference being multiplicative constants.

Let us recall the original definition of variation.

Definition 8. *Let f be a function from $\mathcal{P}([n])$ to $\{-1, 1\}$, and let $I \subseteq [n]$ be a subset of coordinates. We define the* variation *of f on I as*

$$\mathrm{Vr}_f(I) \stackrel{\text{def}}{=} \mathbb{E}_{w \in \mathcal{P}([n]\setminus I)} \left[\mathbb{V}_{z \in \mathcal{P}(I)} \left[f(w \sqcup z) \right] \right].$$

Fischer *et al.* showed the following two facts on the variation, which were the heart of the proofs of the soundness of their algorithms.

Lemma 9 (probability of detection [6])**.** *Let f be a function from $\mathcal{P}([n])$ to $\{-1, 1\}$, and let $I \subseteq [n]$ be a subset of coordinates. If $w \in \mathcal{P}([n]\setminus I)$ and $z_1, z_2 \in \mathcal{P}(I)$ are chosen independently, then*

$$\Pr[f(w \sqcup z_1) \neq f(w \sqcup z_2)] = \frac{1}{2}\mathrm{Vr}_f(I).$$

Lemma 10 (monotonicity and sub-additivity [6])**.**

$$\mathrm{Vr}_f(A) \leq \mathrm{Vr}_f(A \cup B) \leq \mathrm{Vr}_f(A) + \mathrm{Vr}_f(B).$$

A.2. The binary and extreme variation.

Now we will define the notion of the *binary variation* and the *extreme variation* which work also for functions of non-Boolean ranges. Even though now we may have more than 2 different values, we will map the range to only two different values, which results in not distinguishing some values of a function. We will try to minimize the negative effects of such a mapping by taking a mapping that maximizes what we can distinguish. Let us start with the notion of the binary variation.

Definition 11. *Let f be a function from $\mathcal{P}([n])$ to X, and let $I \subseteq [n]$ be a subset of coordinates. We define the* binary variation *of f on I as*

$$\mathrm{BinVr}_f(I) \stackrel{\text{def}}{=} \max_{g \in \mathcal{F}(X)} \mathrm{Vr}_{g \circ f}(I) = \max_{g \in \mathcal{F}(X)} \mathbb{E}_{w \in \mathcal{P}([n]\setminus I)} \left[\mathbb{V}_{z \in \mathcal{P}(I)} \left[g(f(w \sqcup z)) \right] \right].$$

By Lemma 9 and by the definition of the binary variation, the following simple fact follows.

Lemma 12 (probability of detection)**.** *Let f be a function from $\mathcal{P}([n])$ to X, and let $I \subseteq [n]$ be a subset of coordinates. If $w \in \mathcal{P}([n]\backslash I)$ and $z_1, z_2 \in \mathcal{P}(I)$ are chosen independently, then*

$$\Pr[f(w \sqcup z_1) \neq f(w \sqcup z_2)] \geq \frac{1}{2}\mathrm{BinVr}_f(I).$$

The binary variation also is monotone and sub-additive, which directly follows from the sub-additivity and monotonicity of the original variation (Lemma 10).

Lemma 13 (monotonicity and sub-additivity)**.**

$$\mathrm{BinVr}_f(A) \leq \mathrm{BinVr}_f(A \cup B) \leq \mathrm{BinVr}_f(A) + \mathrm{BinVr}_f(B).$$

Proof.

$$\begin{aligned}
\mathrm{BinVr}_f(A) &= \max_g \mathrm{Vr}_{g\circ f}(A) \leq \max_g \mathrm{Vr}_{g\circ f}(A \cup B) \\
&\leq \max_g \left(\mathrm{Vr}_{g\circ f}(A) + \mathrm{Vr}_{g\circ f}(B)\right) \\
&\leq \max_g \mathrm{Vr}_{g\circ f}(A) + \max_g \mathrm{Vr}_{g\circ f}(B) \\
&\leq \mathrm{BinVr}_f(A) + \mathrm{BinVr}_f(B).
\end{aligned}$$

■

Now we will define the extreme variation which differs from the binary variation by switching the order of the expectation and maximization in the definition.

Definition 14. *Let f be a function from $\mathcal{P}([n])$ to X, and let $I \subseteq [n]$ be a subset of coordinates. We define the* extreme variation *of f on I as*

$$\mathrm{ExtVr}_f(I) \stackrel{\text{def}}{=} \mathbb{E}_{w\in\mathcal{P}([n]\backslash I)}\left[\max_{g\in\mathcal{F}(X)} \mathbb{V}_{z\in\mathcal{P}(I)}\left[g(f(w \sqcup z))\right]\right].$$

It turns out that the two new notions of variation are closely related. Namely, they stay within a constant factor.

Lemma 15.

$$\mathrm{BinVr}_f(I) \leq \mathrm{ExtVr}_f(I) \leq 4 \cdot \mathrm{BinVr}_f(I).$$

Proof. The first inequality is trivial, and directly follows from the definitions of the binary and extreme variations.

Focus now on the second inequality. Fix $w \in \mathcal{P}([n]\backslash I)$. To maximize $\mathbb{V}_{z\in\mathcal{P}(I)}[(g \circ f)(w \sqcup z)]$, we need to take g such that splits X into two sets such that the probability that the function value belongs to each of them is as close to $1/2$ as possible. If p is the probability that $(g \circ f)(w \sqcup z) = -1$, then

$$V(p) \stackrel{\text{def}}{=} \mathbb{V}_{z\in\mathcal{P}(I)}[(g \circ f)(w \sqcup z)] = 4p(1-p).$$

Because V is concave in p, we have

$$2V(p/2) \geq V(p)$$

for $p \in [0, 1]$. Let $p_\star$ be the greatest p in the range $[0, 1/2]$ that we can achieve. This means that the corresponding function $g_\star$ splits X into two sets X_1 and X_2 of probability $p_\star$ and $1 - p_\star$, respectively, where the first one is mapped to -1, and the other to 1.

Now consider a function $g \in \mathcal{F}(X)$ that is uniformly chosen at random. Such a g maps at least half (measured by probability) of X_1 to either -1 or 1; assume w.l.o.g. that it maps at least half of X_1 to -1. Independently, with probability at least $1/2$ we have that g maps at least half of X_2 to 1. This means that for a randomly chosen g, with probability $1/2$ we have that p is in the range $[p_\star/2, 1 - p_\star/2]$, which implies in turn that $V(p) \geq V(p_\star)/2$. Therefore,

$$\begin{aligned}
\mathrm{BinVr}_f(I) &= \max_g \mathbb{E}_{w \in \mathcal{P}([n]\setminus I)} \left[\mathbb{V}_{z \in \mathcal{P}(I)} \left[(g \circ f)(w \sqcup z)\right]\right] \\
&\geq \mathbb{E}_g \left[\mathbb{E}_{w \in \mathcal{P}([n]\setminus I)} \left[\mathbb{V}_{z \in \mathcal{P}(I)} \left[(g \circ f)(w \sqcup z)\right]\right]\right] \\
&= \mathbb{E}_{w \in \mathcal{P}([n]\setminus I)} \left[\mathbb{E}_g \left[\mathbb{V}_{z \in \mathcal{P}(I)} \left[(g \circ f)(w \sqcup z)\right]\right]\right] \\
&\geq \mathbb{E}_{w \in \mathcal{P}([n]\setminus I)} \left[\frac{1}{2} \cdot \frac{1}{2} \max_g \mathbb{V}_{z \in \mathcal{P}(I)} \left[(g \circ f)(w \sqcup z)\right]\right] \\
&= \frac{1}{4}\mathrm{ExtVr}_f(I).
\end{aligned}$$

■

A.3. The independence test.

An important sub-test that will be used throughout the main test is the independence test.

Independence test: Given a function f, and a set of variables I, choose $w \in_{\mathrm{R}} \mathcal{P}([n]\setminus I)$ and $z_1, z_2 \in_{\mathrm{R}} \mathcal{P}(I)$. Accept if $f(w \sqcup z_1) = f(w \sqcup z_2)$ and reject if $f(w \sqcup z_1) \neq f(w \sqcup z_2)$.

The independence test always accepts if f is independent of the coordinates in I, and Lemma 12 states that it rejects with probability at least $\frac{1}{2}\mathrm{BinVr}_f(I)$ in the non-Boolean setting, and with probability exactly $\frac{1}{2}\mathrm{Vr}_f(I)$ in the Boolean setting.

A.4. Small variation and closeness to juntas.

Denote by $\mathrm{Plur}_x f(x)$ the most commonly occurring output of f for arguments x with ties broken arbitrarily (often referred to as the plurality).

Fischer *et al.* [6] showed that if the variation of some subset of variables is small, then the function is close to a function that does not depend on these variables. We will show that an almost identical claim holds for the binary variation.

Lemma 16. *Let $\mathcal{J}$ be a set of coordinates such that $\mathrm{BinVr}_f(\overline{\mathcal{J}}) < \frac{1}{4}\epsilon$. Let*

$$h(x) \stackrel{\text{def}}{=} \mathrm{Plur}_{z \in \mathcal{P}(\overline{\mathcal{J}})}[f((x \cap \mathcal{J}) \sqcup z)].$$

The function h is a $|\mathcal{J}|$-junta, depends only on variables in $\mathcal{J}$, and agrees with f on a set of assignments of measure more than $1 - \epsilon$.

The original lemma stated that it suffices to have $\mathrm{Vr}_f(\overline{\mathcal{J}}) < 2\epsilon$ to be ϵ-close to a junta on $\mathcal{J}$ for a Boolean-valued function f. Because of the difference in the required bound on variation of $\overline{\mathcal{J}}$ in the non-Boolean setting ($\epsilon/4$ vs. 2ϵ) we need to run the independence test, which is a subroutine in the junta test, more times to get the required result. Fortunately, it is enough to replace each single run of the independence test by c independent runs for some constant c. (It is also possible that actually the original algorithms with the original constants work for non-Boolean ranges, but to show this, a more careful analysis would be necessary.)

We start with the following lemma that helps us connect simple probabilities for multi-valued functions with probabilities for two-valued functions.

Lemma 17. *Let f be a function from a set D (with some probability measure on it) to X. It holds that*

$$\Pr_x[f(x) = \mathrm{Plur}_y f(y)] \geq 2 \min_{g\in\mathcal{F}(X)} \Pr_x[(g\circ f)(x) = \mathrm{Plur}_y (g\circ f)(y)] - 1.$$

Proof. Let $p = \Pr_x[f(x) = \mathrm{Plur}_y f(y)]$. This means that for any r in X it holds that $p \geq \Pr_x[f(x) = r]$. Enumerate elements of X. They are r_1, r_2, r_3, and so forth. Denote by p_i the probability that $f(x)$ equals r_j for $j \geq i$. Obviously, $p_i = p_{i+1} + \Pr_x[f(x) = r_i] \leq p_{i+1} + p$, that is $p_{i+1} \geq p_i - p$. Since $p_1 = 1$ and the sequence p_i converges to 0 and does not drop too quickly, there is an index $i_\star$ such that $p_{i_\star} \in [(1-p)/2, (1+p)/2]$. Let $X_1 = \{r_1, \ldots, r_{i_\star - 1}\}$, and $X_2 = \{r_{i_\star}, r_{i_\star+1}, \ldots\}$. Define $g_\star : X \to \{-1, 1\}$ as

$$g_\star(r) \stackrel{\mathrm{def}}{=} \begin{cases} -1 & \text{for } r \in X_1, \\ 1 & \text{for } r \in X_2. \end{cases}$$

It holds that

$$\max_{g\in\mathcal{F}(X)} \Pr_x[(g\circ f)(x) \neq \mathrm{Plur}_y (g\circ f)(y)] \geq \Pr_x[(g\circ f_\star)(x) \neq \mathrm{Plur}_y (g\circ f_\star)(y)] \geq \frac{1-p}{2},$$

which can be rearranged to the required form:

$$\begin{aligned} 1 - \min_{g\in\mathcal{F}(X)} \Pr_x[(g\circ f)(x) = \mathrm{Plur}_y (g\circ f)(y)] &\geq \frac{1}{2}\left(1 - \Pr_x[f(x) = \mathrm{Plur}_y f(y)]\right), \\ \Pr_x[f(x) = \mathrm{Plur}_y f(y)] &\geq 2 \min_{g\in\mathcal{F}(X)} \Pr_x[(g\circ f)(x) = \mathrm{Plur}_y (g\circ f)(y)] - 1. \end{aligned}$$

■

Now we can prove our main lemma on binary variation and closeness to juntas:

Proof of Lemma 16. Let $y \in \mathcal{P}(\mathcal{J})$ and $z \in \mathcal{P}(\overline{\mathcal{J}})$. We have

$$h(y \sqcup z) = \mathrm{Plur}_{t\in\mathcal{P}(\overline{\mathcal{J}})} f(y \sqcup t).$$

Assume now that $x \in \mathcal{P}([n])$, y, z and t are random over their respective domains and independent. We have

$$\begin{aligned} \Pr_x[f(x) = h(x)] &= \mathbb{E}_y\left[\Pr_z[f(y\sqcup z) = h(y\sqcup z)]\right] \\ &= \mathbb{E}_y\left[\Pr_z[f(y\sqcup z) = \mathrm{Plur}_t f(y\sqcup t)]\right] \\ &\geq \mathbb{E}_y\left[2\min_{g\in F(X)} \Pr_z[(g\circ f)(y\sqcup z) = \mathrm{Plur}_t (g\circ f)(y\sqcup t)] - 1\right] && (3) \\ &= \mathbb{E}_y\left[\min_{g\in F(X)} \mathbb{E}_z\left[(g\circ f)(y\sqcup z)\cdot \mathrm{Plur}_t (g\circ f)(y\sqcup t)\right]\right] && (4) \\ &= \mathbb{E}_y\left[\min_{g\in F(X)} \mathbb{E}_z\left[(g\circ f)(y\sqcup z)\right]\cdot \mathrm{sign}(\mathbb{E}_t[(g\circ f)(y\sqcup t)])\right] \\ &= \mathbb{E}_y\left[\min_{g\in F(X)} \left|\mathbb{E}_z\left[(g\circ f)(y\sqcup z)\right]\right|\right] \\ &\geq \mathbb{E}_y\left[\min_{g\in F(X)} \left(\mathbb{E}_z\left[(g\circ f)(y\sqcup z)\right]\right)^2\right] \\ &= \mathbb{E}_y\left[1 - \max_{g\in F(X)} \mathbb{V}_z\left[(g\circ f)(y\sqcup z)\right]\right] \\ &= 1 - \mathrm{ExtVr}_f(\overline{\mathcal{J}}) \geq 1 - 4\mathrm{BinVr}_f(\overline{\mathcal{J}}) > 1 - \epsilon, && (5) \end{aligned}$$

where (3) is by Lemma 17 applied to the function $f(y \sqcup \cdot)$, (4) is because $f \circ g$ and Plur are both ± 1-valued, and the first inequality in (5) is by Lemma 15. ■

A.5. Unique variation.

We will make use of the following technical tool which was defined by Fischer *et al.* [6].

Definition 18. Let f be a function that maps $\mathcal{P}([n])$ to $\{-1, 1\}$, and let $\mathcal{J} \subseteq [n]$ be a set of coordinates. For each coordinate $i \in [n]$, we define the *unique variation of* i *with respect to* $\mathcal{J}$ as

$$\mathrm{Ur}_f(i) \stackrel{\text{def}}{=} \mathrm{Vr}_f([i] \backslash \mathcal{J}) - \mathrm{Vr}_f([i-1] \backslash \mathcal{J}),$$

and for $I \subseteq [n]$ we define the unique variation of I as

$$\mathrm{Ur}_f(I) \stackrel{\text{def}}{=} \sum_{i \in I} \mathrm{Ur}_f(i).$$

The most important property of the unique variation that distinguishes it from the other notions of variation is that for any set of coordinates, its variation simply equals the sum of the variations of each of its coordinates. This makes it easy to compute the expected value of the unique variation on a random subset of coordinates. Furthermore, the following properties hold.

Lemma 19 (Fischer *et al.* [6])**.**

- *For any coordinate* $i \in [n]$, $\mathrm{Ur}_f(\{i\}) \leq \mathrm{Vr}_f(\{i\})$.
- *For every set* $I \subseteq [n]$ *of coordinates,* $\mathrm{Ur}_f(I) \leq \mathrm{Vr}_f(I \backslash \mathcal{J})$.
- $\mathrm{Ur}_f([n]) = \mathrm{Ur}_f([n] \backslash \mathcal{J}) = \mathrm{Vr}_f([n] \backslash \mathcal{J})$.

We will also use the following technical claim.

Lemma 20 (Fischer *et al.* [6])**.** *Let* $X = \sum_{i=1}^{l} X_i$ *be a sum of non-negative independent random variables* X_i*, and denote expectation of* X *by* α*. If every* X_i *is bounded above by* t*, then*

$$\Pr[X < \eta\alpha] < \exp\left(\frac{\alpha}{et}(\eta e - 1)\right)$$

for every $\eta > 0$.

A.6. Application to testing juntas.

It turns out that one can use the binary variation in place of the variation of Fischer *et al.* to carry out the proof that their algorithms work in the non-Boolean setting. The only difference is in some constant factors – we want to make sure that the set of variables that we classify as non-relevant has binary variation at most $\epsilon/4$, instead of variation 2ϵ in the original analysis. This results in an increase in the number of runs of the independence test by a constant factor. Other than this small difference, the properties established above for the binary variation let the proofs given by Fischer *et al.* go through directly for non-Boolean functions, so we do not repeat them. Summarizing, we get three tests for J-juntas for functions with non-Boolean ranges from [6]:

- a non-adaptive one-sided test with query complexity $\tilde{O}(J^4/\epsilon)$,
- an adaptive one-sided test with query complexity $\tilde{O}(J^3/\epsilon)$,
- a non-adaptive two-sided test with query complexity $\tilde{O}(J^2/\epsilon)$.

The last of these is simply the **Identify-Critical-Subsets** subroutine from Figure 1, modified to output "yes" in Step 4 instead of the list of critical subsets.

B. Proof of Theorem 4

For convenience we restate Theorem 4 in somewhat more detail below:

Theorem 4. *There is an algorithm $\mathcal{A}$ with the following properties:*

Let $\mathcal{C}$ be a class of functions from Ω^n to X. Suppose that for every $\tau > 0$, $\mathcal{C}(\tau) \subseteq \mathcal{C}$ is a $(\tau, J(\tau))$-approximator for $\mathcal{C}$. Suppose moreover that for every $\epsilon > 0$, there is a τ satisfying

$$\tau \leq \kappa \cdot \frac{\epsilon^2}{\ln(|\Omega|) \cdot J(\tau)^2 \cdot \ln^2(J(\tau)) \cdot \ln\ln(J(\tau)) \cdot \ln^2(|\mathcal{C}(\tau)_{J(\tau)}|) \cdot \ln(\frac{\ln(|\Omega|)}{\epsilon} \ln |\mathcal{C}(\tau)_{J(\tau)}|)},$$

where $\kappa > 0$ is a fixed absolute constant. Let $\tau^\star$ be the largest value τ satisfying (2) above. Then algorithm $\mathcal{A}$ makes:

$$\begin{aligned} & 2sh + (2gJ(\tau^\star)\lceil \lg |\Omega| \rceil + 1)m \\ = \quad & \Theta\left(\frac{1}{\epsilon} J(\tau^\star)^2 \ln^2(J(\tau^\star)) \log\log J(\tau^\star) \ln(|\mathcal{C}(\tau^\star)_{J(\tau^\star)}|)\right) \\ & + \Theta\left(\frac{\lg |\Omega|}{\epsilon^2} J(\tau^\star)^2 \ln^2(|\mathcal{C}(\tau^\star)_{J(\tau^\star)}|) \ln(\frac{1}{\epsilon} \ln(|\mathcal{C}(\tau^\star)_{J(\tau^\star)}|))\right) \\ = \quad & \tilde{O}\left(\frac{\ln |\Omega|}{\epsilon^2} J(\tau^\star)^2 \ln^2(|\mathcal{C}(\tau^\star)_{J(\tau^\star)}|)\right) \end{aligned}$$

many black-box queries to f, and satisfies the following:

- *If $f \in \mathcal{C}$ then $\mathcal{A}$ outputs "yes" with probability at least $2/3$;*
- *If f is ϵ-far from $\mathcal{C}$ then $\mathcal{A}$ outputs "no" with probability at least $2/3$.*

Let us describe how the parameters s, h, g and m mentioned above (and others) are set. (The table below should perhaps be glossed over on a first pass through the paper, but will be useful for subsequent reference.) Given $\epsilon > 0$, let $\tau^\star$ be as described in the theorem statement. We set:

$r \stackrel{\text{def}}{=} 25J(\tau^\star)^2$	$\Theta(J(\tau^\star)^2)$,
$s \stackrel{\text{def}}{=} 25J(\tau^\star)(7 + \ln r)$	$\Theta(J(\tau^\star) \ln J(\tau^\star))$,
$\epsilon_2 \stackrel{\text{def}}{=} \frac{\epsilon}{2}$	$\Theta(\epsilon)$,
$m \stackrel{\text{def}}{=} \frac{1}{\epsilon_2} \ln 6 \lvert\mathcal{C}(\tau^\star)_{J(\tau^\star)}\rvert$	$\Theta(\frac{1}{\epsilon} \ln(\lvert\mathcal{C}(\tau^\star)_{J(\tau^\star)}\rvert))$,
$\epsilon_1 \stackrel{\text{def}}{=} \frac{1}{200m}$	$\Theta(\epsilon / \ln(\lvert\mathcal{C}(\tau^\star)_{J(\tau^\star)}\rvert))$
$\theta \stackrel{\text{def}}{=} \frac{\epsilon_1 J(\tau^\star)}{24er}$	$\Theta(\epsilon / (\ln(\lvert\mathcal{C}(\tau^\star)_{J(\tau^\star)}\rvert) J(\tau^\star)))$,
$g \stackrel{\text{def}}{=} \frac{2}{\theta} \ln\big(100mJ(\tau^\star)\lceil \lg \lvert\Omega\rvert \rceil\big)$	$\Theta\left(\frac{1}{\epsilon} J(\tau^\star) \ln(\lvert\mathcal{C}(\tau^\star)_{J(\tau^\star)}\rvert) \cdot \ln\left(\frac{\ln \lvert\Omega\rvert}{\epsilon} J(\tau^\star) \ln(\lvert\mathcal{C}(\tau^\star)_{J(\tau^\star)}\rvert)\right)\right)$,
$h \stackrel{\text{def}}{=} \frac{2}{\theta}(3 + 2\ln s)$	$\Theta(\frac{1}{\epsilon} \ln(\lvert\mathcal{C}(\tau^\star)_{J(\tau^\star)}\rvert) J(\tau^\star) \ln J(\tau^\star) \ln\ln J(\tau^\star))$,

where e is the base of the natural logarithm. Note that $\epsilon_1 + \epsilon_2 < \epsilon$.

Observe that for some suitable (small) absolute constant $\kappa > 0$, our setting of parameters and choice of $\tau^\star$ yields the following bounds that we will use later:

- $2mgJ(\tau^\star)\lceil \lg |\Omega| \rceil \cdot \tau^\star \leq 1/100$ (used in Lemma 26)
- $2sh \cdot \tau^\star \leq 1/100$ (used in Corollary 25),
- $m(\epsilon_1 + \tau^\star) < 1/100$ (used in Lemma 26).

Identify-Critical-Subsets (input is black-box access to $f : \Omega^n \to X$ and $\epsilon > 0$)

1. Partition the variables $x_1, \dots, x_n$ into r random subsets by assigning each of $x_1, \dots, x_n$ equiprobably to one of $I_1, \dots, I_r$.
2. Choose s random subsets $\Lambda_1, \dots, \Lambda_s \subseteq [r]$ of size $J(\tau^\star)$ by uniformly choosing without repetitions $J(\tau^\star)$ members of $[r]$. Each set Λ_i determines a block $B_i \stackrel{\text{def}}{=} \bigcup_{j \in \Lambda_i} I_j$. (Note that we do not guarantee that the blocks are disjoint.)
3. Apply h iterations of the *independence test* (see Section A.3) to each block B_i. If all of the independence test iterations applied to block B_i accept, then B_i is declared to be *variation-free*, and all the subsets I_j with $j \in \Lambda_i$ are declared to be variation-free on its behalf.
4. If:
   (a) at least half of the blocks $B_1, \dots, B_s$ are variation-free; and
   (b) except for at most $J(\tau^\star)$ subsets, every subset in the partition $I_1, \dots, I_r$ is declared variation-free on behalf of some block,

   then output the list $I_{i_1}, \dots, I_{i_j}$ of those subsets that are *not* declared to be variation-free. (We call these the *critical* subsets.) Otherwise, halt and output "Not in $\mathcal{C}$."

Figure 4. The subroutine Identify-Critical-Subsets.

B.1. Step 1: Identifying critical subsets.

Step 1 of the algorithm consists of running the procedure **Identify-Critical-Subsets**, reproduced for convenience in Figure 4. This procedure performs $2sh$ queries to f. The procedure is nearly identical to the "two-sided" junta test of Section 4.2 of Fischer *et al.* with two small differences. The first is that we have adjusted various constant factors slightly (we need a smaller failure probability because we are using this in the context of a larger test). The second is that **Identify-Critical-Subsets** outputs the list of subsets that are declared to be not variation-free (whereas the Fischer *et al.* test simply accepts or rejects f), since we will need these subsets for the rest of our test.

We now prove two quick lemmata that will be useful in establishing the soundness and completeness of the algorithm.

Lemma 21. *Let f be a function with at most $J(\tau^\star)$ variables x_i that have* $\mathrm{BinVr}_f(\{i\}) \geq \theta$*. Then with probability at least* $1 - 1/400$*, each of the variables x_i that have* $\mathrm{BinVr}_f(\{i\}) \geq \theta$ *occurs in some subset I_ℓ that is not declared variation-free by* **Identify-Critical-Subsets**.

Proof. Fix a variable x_i such that $\mathrm{BinVr}_f(\{i\}) \geq \theta$. Let I_ℓ denote the subset to which x_i belongs. By Lemma 13 we have that

$$\theta \leq \mathrm{BinVr}_f(\{i\}) \leq \mathrm{BinVr}_f(I_\ell) \leq \mathrm{BinVr}_f(B_k)$$

where B_k is any block such that $\ell \in \Lambda_k$. This implies that for any such block B_k, the probability that all h iterations of the independence test accept is at most $(1 - \frac{\theta}{2})^h < \frac{1}{20s^2} < \frac{1}{400sJ(\tau^\star)}$. So the probability that any block that contains x_i is declared variation-free is at most $\frac{1}{400J(\tau^\star)}$. By a union bound over all at most $J(\tau^\star)$ variables x_i that have $\mathrm{BinVr}_f(\{i\}) \geq \theta$, the probability that any block that contains such a variable causes any subset I_ℓ containing the variable to be declared variation-free is at most $1/400$. ■

Lemma 22. *Let V be any set of at most $J(\tau^\star)$ variables from $x_1, \ldots, x_n$. Then with probability at least $1 - 1/25$, every subset I_ℓ, $1 \le \ell \le r$, contains at most one variable from V.*

Proof. Let j' denote the number of variables in V. The probability that no two variables in V end up in the same subset I_i is

$$\frac{r!}{(r-j')!r^{j'}} > \left(1 - \frac{j'-1}{r}\right)^{j'} > 1 - \frac{j'(j'-1)}{r} = 1 - \frac{j'(j'-1)}{25J(\tau^\star)^2} > 1 - \frac{1}{25}.$$

So the probability that any subset $I_1, \ldots, I_r$ ends up with two or more variables from V is at most $1/25$. ■

Let $\mathcal{K} \subseteq [n]$ denote a set of coordinates satisfying $\mathrm{BinVr}_f(\overline{\mathcal{K}}) < \frac{1}{4}\epsilon_1$. Lemma 16 states that the following function:

$$h(x) \stackrel{\text{def}}{=} \mathrm{Plur}_{z \in \mathcal{P}(\overline{\mathcal{K}})}[f((x \cap \mathcal{K}) \sqcup z)] \tag{6}$$

is ϵ_1-close to f.

Let $\mathcal{J}$ denote the set of those coordinates on which f has binary variation at least θ. To prove the soundness of **Identify-Critical-Subsets**, we must prove that if f passes **Identify-Critical-Subsets** with probability greater than 1/3, then it is ϵ_1-close to a $J(\tau^\star)$-junta. This is accomplished by showing that $|\mathcal{J}| \le J(\tau^\star)$, and that $\mathcal{J}$ can be used in place of $\mathcal{K}$ above, *i.e.*, $\mathrm{BinVr}_f(\overline{\mathcal{J}}) < \frac{1}{4}\epsilon_1$. Then we can invoke Lemma 16 to finish the proof. In addition, we will also prove some properties about the subsets $I_{i_1}, \ldots, I_{i_j}$ output by the algorithm.

Lemma 23. *If f passes* **Identify-Critical-Subsets** *with probability higher than 1/3, then:*

(i) $|\mathcal{J}| \le J(\tau^\star)$*;*

(ii) $\mathrm{BinVr}_f(\overline{\mathcal{J}}) < \frac{1}{4}\epsilon_1$,

and f is thus ϵ_1-close to a $J(\tau^\star)$-junta by Lemma 16.

Let h be defined as in Equation (6) using $\mathcal{J}$ as the set $\mathcal{K}$. Suppose that f passes **Identify-Critical-Subsets** *with probability greater than 1/3. Then given that f passes, the sets output by the algorithm, $I_{i_1}, \ldots, I_{i_j}$, have the following properties with probability at least 6/7:*

(iii) Every $x_i \in \mathcal{J}$ occurs in some subset I_{i_ℓ} that is output;

(iv) Every subset I_{i_ℓ}, $1 \le \ell \le j$, contains at most one variable from $\mathcal{J}$.

Proof. **Condition (i):** (paraphrasing Prop. 3.1 and Lemma 4.3 of [6]) Suppose $|\mathcal{J}| > J(\tau^\star)$. Then with probability at least $3/4$ (using the same argument as in the proof of Lemma 22), the number of subsets I_{i_ℓ} containing an element from $\mathcal{J}$ is at least $J(\tau^\star) + 1$. For any fixed subset I_{i_ℓ} that contains an element from $\mathcal{J}$ and any fixed block B containing I_{i_ℓ}, the probability of B being declared variation-free is bounded by:

$$(1 - \theta/2)^h = (1 - \theta/2)^{2(3 + 2\ln s)/\theta} < \frac{1}{20s(J(\tau^\star) + 1)}.$$

Union bounding over the at most s blocks to which the subset I_{i_ℓ} can belong, and union bounding over $J(\tau^\star) + 1$ subsets that contain an element from $\mathcal{J}$, we have that with probability at least $\frac{3}{4} \cdot \frac{19}{20} > \frac{2}{3}$, at least $J(\tau^\star) + 1$ subsets are not declared variation-free and consequently f does not pass **Identify-Critical-Subsets**. Thus, if f passes **Identify-Critical-Subsets** with probability at least $1/3$, it must be the case that $|\mathcal{J}| \le J(\tau^\star)$.

Condition (ii): (paraphrasing Prop. 3.1 and Lemma 4.3 of [6]) Suppose $\mathrm{BinVr}_f(\overline{\mathcal{J}}) \ge \frac{1}{4}\epsilon_1$, and let g be a function such that $\mathrm{BinVr}_f(\overline{\mathcal{J}}) = \mathrm{Vr}_{g \circ f}(\overline{\mathcal{J}})$. We will show that each block B_ℓ has high variation with high probability. This will imply that the number of blocks not declared variation-free is larger than $s/2$ with high probability, so the test will reject with probability at least $2/3$.

Fix any value $\ell \in [s]$. The block B_ℓ is a random set of variables independently containing each variable x_i coordinate with probability $J(\tau^\star)/r$. Let $\mathrm{Ur}_{g\circ f}(I)$ be the unique variation of a set I with respect to $\mathcal{J}$ (see Definition 18). Then the expected value of the unique variation of B_ℓ is

$$\mathbb{E}[\mathrm{Ur}_{g\circ f}(B_\ell)] = \frac{J(\tau^\star)}{r}\mathrm{Ur}_{g\circ f}(\overline{\mathcal{J}}) = \frac{J(\tau^\star)}{r}\mathrm{Vr}_{g\circ f}(\overline{\mathcal{J}}) \geq \frac{\epsilon_1 J(\tau^\star)}{4r}.$$

By Lemma 19 and Lemma 20 (taking $\eta = 1/2e$, $t = \theta$ and $\alpha = \frac{\epsilon_1 J(\tau^\star)}{4r}$ in Lemma 20), we have

$$\Pr\left[\mathrm{Vr}_{g\circ f}(B_\ell) < \frac{\epsilon_1 J(\tau^\star)}{8er}\right] \leq \Pr\left[\mathrm{Ur}_{g\circ f}(B_\ell) < \frac{\epsilon_1 J(\tau^\star)}{8er}\right] < \exp\left(-\frac{\epsilon_1 J(\tau^\star)}{8er\theta}\right) = e^{-3} < \frac{1}{12}.$$

Hence the probability that the variation of B_ℓ is less than $\epsilon_1 J(\tau^\star)/8er = 3\theta$ is less than $1/12$. This implies that the expected number of blocks with variation less than 3θ is smaller than $s/12$. From Markov's inequality we get that with probability at least $1 - \frac{1}{6}$, there are less than $s/2$ blocks with variation smaller than 3θ.

The probability of a block with variation greater than 3θ being declared variation free is at most:

$$\left(1 - \frac{3\theta}{2}\right)^h = \left(1 - \frac{3\theta}{2}\right)^{2(3+2\ln s)/\theta} < e^{-(9+6\ln s)} < \frac{1}{1000s},$$

and therefore with probability at least $1 - \frac{1}{1000}$ none of these blocks are declared variation free. So with overall probability at least $1 - (\frac{1}{6} + \frac{1}{1000}) > \frac{2}{3}$, more than $s/2$ blocks are declared variation-free and the test rejects.

Condition (iii): We may suppose that f passes **Identify-Critical-Subsets** with probability greater than 1/3. Then we know that $|\mathcal{J}| \leq J(\tau^\star)$ by Condition (i). By Lemma 21, given that f passes **Identify-Critical-Subsets**, the probability that some $x_i \in \mathcal{J}$ does not occur in some subset I_{i_ℓ} output by the algorithm is at most $3/400$. (The bound is $3/400$ rather than $1/400$ because we are conditioning on f passing **Identify-Critical-Subsets**, which takes place with probability at least $1/3$.)

Condition (iv): As above we may suppose that f passes **Identify-Critical-Subsets** with probability greater than 1/3. By Condition (i) we know that $|\mathcal{J}| \leq J(\tau^\star)$, so we may apply Lemma 22. Hence conditioned on f passing **Identify-Critical-Subsets** (an event which has probability at least $1/3$), the probability that any subset I_{i_ℓ} output by the algorithm includes more than one relevant variable of h is at most $3/25$.

Summing the probabilities, we get that conditions (iii) and (iv) are true with probability at least $1 - (\frac{3}{400} + \frac{3}{25}) > \frac{6}{7}$. ∎

Fischer *et al.* establish completeness by showing that if f is a junta then with probability at least $2/3$ conditions (a) and (b) are both satisfied in Step 4. However we need more than this, since we are going to use the subsets $I_{i_1}, \ldots, I_{i_j}$ later in the test. We will prove:

Lemma 24. *Suppose that f is a $J(\tau^\star)$-junta. Let $\mathcal{K}$ be the set of variables satisfying $\mathrm{BinVr}_f(\{i\}) \geq \theta$. Then with probability at least $6/7$, algorithm* **Identify-Critical-Subsets** *outputs a list of $j \leq J(\tau^\star)$ subsets $I_{i_1}, \ldots, I_{i_j}$ with the property that:*

(i) each variable $x_i \in \mathcal{K}$ occurs in some subset I_ℓ that is output;

(ii) $\mathrm{BinVr}_f(\overline{\mathcal{K}}) < \epsilon_1/4$;

(iii) Every subset I_{i_ℓ}, $1 \leq \ell \leq j$, contains at most one relevant variable for f.

Proof. **Condition (a):** Fix any partition $I_1, \ldots, I_r$. If f is a $J(\tau^\star)$-junta, then it is independent of all but at most $J(\tau^\star)$ subsets in the partition. Hence for any fixed ℓ, the probability over the selection of the blocks that f is independent of B_ℓ is at least:

$$\binom{r - J(\tau^\star)}{J(\tau^\star)} \Big/ \binom{r}{J(\tau^\star)} > \left(\frac{r - 2J(\tau^\star)}{r - J(\tau^\star)}\right)^{J(\tau^\star)} = \left(1 - \frac{J(\tau^\star)}{r - J(\tau^\star)}\right)^{J(\tau^\star)} > 1 - \frac{J(\tau^\star)^2}{r - J(\tau^\star)} \geq \frac{23}{24}.$$

The probability that f depends on more than half of the blocks is therefore smaller than $\frac{2}{24}$ using the Markov inequality. (See [6], Lemma 4.2).

Condition (b) fails with probability at most:

$$r\left(1 - \frac{1}{25J(\tau^\star)}\right)^s = r\left(1 - \frac{1}{25J(\tau^\star)}\right)^{25J(\tau^\star)(7+\ln r)} < r \cdot S\frac{1}{1000r} = \frac{1}{1000},$$

(see [6], Lemma 4.2, which uses $s = 20J(3 + \ln r)$ instead).

Condition (i): Since we assume that f is a $J(\tau^\star)$-junta we may apply Lemma 21, and thus the probability that any variable x_i that has $\mathrm{Vr}_f(\{i\}) \geq \theta$ occurs in a subset I_ℓ that is declared variation-free by **Identify-Critical-Subsets** is at most $1/400$.

Condition (ii): Let $\mathcal{L}$ denote the relevant variables for f that are not in $\mathcal{K}$, and let $\mathcal{T}$ denote $[n]\backslash(\mathcal{K}\cup\mathcal{L})$. By Lemma 10 we have

$$\mathrm{BinVr}_f(\mathcal{L}) \leq \sum_{i\in\mathcal{L}} \mathrm{BinVr}_f(\{i\}) \leq J(\tau^\star)\theta = J(\tau^\star)\frac{\epsilon_1 J(\tau^\star)}{24e \cdot 25J(\tau^\star)^2} < \frac{\epsilon_1}{4}.$$

We have that $\overline{\mathcal{K}} = \mathcal{L} \cup \mathcal{T}$, so by Lemma 10 we get

$$\mathrm{BinVr}_f(\overline{\mathcal{K}}) = \mathrm{BinVr}_f(\mathcal{L} \cup \mathcal{T}) \leq \mathrm{BinVr}_f(\mathcal{L}) + \mathrm{BinVr}_f(\mathcal{T}) = \mathrm{BinVr}_f(\mathcal{L}) \leq \epsilon_1/4.$$

Condition (iii): Suppose there are precisely $j' \leq J(\tau^\star)$ many relevant variables. Then by Lemma 22, the probability that any subset $I_1, \ldots, I_r$ ends up with two or more relevant variables is at most $1/25$.

Summing failure probabilities, we find that all the required conditions are fulfilled with probability at least $1 - (1/12 + 1/1000 + 1/400 + 1/25)$ which is greater than $6/7$. ■

We are ultimately interested in what happens when **Identify-Critical-Subsets** is run on a function from $\mathcal{C}$. Using the above, we have:

Corollary 25. *Suppose f is $\tau^\star$-close to some $J(\tau^\star)$-junta f'. Then with probability at least $5/6$, algorithm* **Identify-Critical-Subsets** *outputs a list of $j \leq J(\tau^\star)$ subsets $I_{i_1}, \ldots, I_{i_j}$ with the property that*

(i') each variable x_i which has $\mathrm{BinVr}_{f'}(\{i\}) \geq \theta$ occurs in some subset I_ℓ that is output;

(ii') $\mathrm{BinVr}_{f'}(\overline{\mathcal{K}}) < \epsilon_1/4$;

(iii') Every subset I_{i_ℓ}, $1 \leq \ell \leq j$, contains at most one relevant variable for f'.

Proof. The crucial observation is that each of the $2sh$ queries that **Identify-Critical-Subsets** performs is on an input that is selected *uniformly at random* from Ω^n (note that the query points are not all independent of each other, but each one considered individually is uniformly distributed). Since f and f' disagree on at most a $\tau^\star$ fraction of all inputs, the probability that **Identify-Critical-Subsets** queries any point on which f and f' disagree is at most $2sh \cdot \tau^\star < 1/100$. Since by Lemma 24 we know that conditions (i'), (ii') and (iii') would hold with probability at least $6/7$ if the black-box function were f', we have that conditions (i), (ii) and (iii) hold with probability at least $6/7 - 1/100 > 5/6$ with f as the black-box function. ■

Construct-Sample (input is the list $I_{i_1}, \ldots, I_{i_j}$ output by **Identify-Critical-Subsets** and black-box access to f)

1. Repeat the following m times to construct a set S of m labeled examples $(x, y) \in \Omega^{J(\tau^\star)} \times X$, where $\Omega = \{\omega_0, \omega_1, \ldots, \omega_{|\Omega|-1}\}$:
   (a) Draw z uniformly from Ω^n. Let $X_q \stackrel{\text{def}}{=} \{i : z_i = \omega_q\}$, for each $0 \le q \le |\Omega| - 1$.
   (b) For $\ell = 1, \ldots, j$
      i. $w \stackrel{\text{def}}{=} 0$
      ii. For $k = 1, \ldots, \lceil \lg |\Omega| \rceil$
         A. $\Omega_0 \stackrel{\text{def}}{=}$ union of $(X_q \cap I_{i_\ell})$ taken over all $0 \le q \le |\Omega| - 1$ such that the k-th bit of q is zero
         B. $\Omega_1 \stackrel{\text{def}}{=}$ union of $(X_q \cap I_{i_\ell})$ taken over all $0 \le q \le |\Omega| - 1$ such that the k-th bit of q is one
         C. Apply g iterations of the *independence test* to Ω_0. If any of the g iterations reject, mark Ω_0. Similarly, apply g iterations of the *independence test* to Ω_1; if any of the g iterations reject, mark Ω_1.
         D. If exactly one of Ω_0, Ω_1 (say Ω_b) is marked, set the k-th bit of w to b.
         E. If neither of Ω_0, Ω_1 is marked, set the k-th bit of w to unspecified.
         F. If both Ω_0, Ω_1 are marked, halt and output "no".
      iii. If any bit of w is unspecified, choose w at random from $\{0, 1, \ldots, |\Omega| - 1\}$.
      iv. If $w \notin [0, |\Omega| - 1]$, halt and output "no."
      v. Set $x_\ell = \omega_w$.
   (c) Evaluate f on z, assign the remaining $J(\tau^\star) - j$ coordinates of x randomly, and add the pair $(x, f(z))$ to the sample of labeled examples being constructed.

Figure 5. The subroutine Construct-Sample.

B.2. Step 2: Constructing a sample.

Step 2 of the algorithm consists of running the procedure **Construct-Sample**. The algorithm makes $(2gj\lceil \lg |\Omega| \rceil + 1)m$ many queries to f, and either outputs "no" or else outputs a sample of m labeled examples (x, y) where each x belongs to $\Omega^{J(\tau^\star)}$.

We introduce some notation. Given functions $f : \Omega^n \to X$ and $f' : \Omega^j \to X$ with $j \le n$ and a permutation $\sigma : [n] \to [n]$, we write $f \stackrel{\sigma}{\sim} f'$ to indicate that $\forall x \in \Omega^n : f'(x_{\sigma(1)}, \ldots, x_{\sigma(j)}) = f(x_1, \ldots, x_n)$. If $f : \Omega^n \to X$ is a function with j relevant variables, we use f_j^σ to mean the function over j variables that results by mapping the i-th relevant variable under f to the i-th character of a j-character string over Ω; i.e. if σ is a permutation which induces such a mapping, then f_j^σ is the function satisfying $f \stackrel{\sigma}{\sim} f_j^\sigma$. Given a function $f : \Omega^j \to X$ and permutation $\sigma : [n] \to [n]$, we write $f_\uparrow^\sigma$ to denote the j-junta satisfying $f_\uparrow^\sigma \stackrel{\sigma}{\sim} f$.

Lemma 26. *Given $f : \Omega^n \to X$ and some $J(\tau^\star)$-junta f' that is $\tau^\star$-close to f, let $\mathcal{K}$ be the set of variables satisfying $\mathrm{BinVr}_{f'}(\{i\}) \ge \theta$. Suppose* **Construct-Sample** *is given oracle access to f and inputs $I_{i_1}, \ldots, I_{i_j}$, with $j \le J(\tau^\star)$, where:*

1. *Each variable $x_i \in \mathcal{K}$ is contained in one of $I_{i_1}, \ldots, I_{i_j}$;*

2. $\mathrm{BinVr}_{f'}(\overline{\mathcal{K}}) < \epsilon_1/4$;

3. *Every subset I_{i_ℓ}, $1 \le \ell \le j$, contains at most one relevant variable for f'.*

Let h be the function defined as in Equation 6 using the set $\mathcal{K}$. Let $\mathcal{H} \subseteq \mathcal{K}$ be the set of relevant variables for h, and let $\sigma : [n] \to [n]$ be some permutation which maps the variable from $\mathcal{H}$ in bin I_{i_ℓ} to bit ℓ. Then with probability at least $1 - 3/100$, **Construct-Sample** *outputs a set of m uniform, random examples labeled according to a $J(\tau^\star)$-junta g which depends on no variables outside of $\mathcal{K}$ and satisfies $\Pr_{z \in \Omega^n}[g^\sigma_\uparrow(z) \neq f'(z)] \le \epsilon_1$.*

Proof. By Lemma 16 we have that $\Pr_{z \in \Omega^n}[h(z) \neq f'(z)] \le \epsilon_1$. We now show that except with probability less than $3/100$, **Construct-Sample** produces a set S of m examples that are uniform, random, and labeled according to $g \stackrel{\text{def}}{=} h^\sigma_{J(\tau^\star)}$ (note that $g^\sigma_\uparrow \equiv h$).

Consider a particular iteration of Step 1 of **Construct-Sample**. The iteration generates an example x that is uniform random and labeled according to g if

(a) for every bin I_{i_ℓ} which contains a variable from $\mathcal{H}$, Step 1(b)ii constructs the index w such that X_w contains that variable;

(b) for every bin I_{i_ℓ} that contains no variable from $\mathcal{H}$, in every iteration of Step 1(b)ii(C) at most one of Ω_0, Ω_1 is marked, and the value w that is considered in Step 1(b)iv lies in $[0, |\Omega| - 1]$; and

(c) $h(z) = f(z)$.

Item (a) ensures that if I_{i_ℓ} contains a variable from $\mathcal{H}$, then x_ℓ takes the value of that variable under the assignment z (and, since z is a uniform random value, so is x_ℓ). Item (b) ensures that if I_{i_ℓ} contains no variable from $\mathcal{H}$, **Construct-Sample** does not output "no" and assigns x_ℓ a uniform random value, because x_ℓ either gets a fresh uniform random value in Step 1(b)iii or gets the value of z (which is uniform random). Together, these ensure that $g(x) = g(z_{\sigma(1)}, \ldots, z_{\sigma(J(\tau^\star))})$, and item (c) ensures that the label for the example x will be $h(z) = g(x)$.

It remains to bound the probability that any of (a), (b), or (c) fail to hold. Suppose first that every query of every iteration of the independence test is answered according to f'. Then item (3) implies that (a) can only fail to hold if we do not manage to figure out some bit of w in Step 1(b)ii for some ℓ for which I_{i_ℓ} contains a variable from $\mathcal{H}$ (which means that all g executions of the independence test pass for that bit failed), and it also implies that condition (b) holds (it is possible for a bit of w to be unspecified, but not for both Ω_0, Ω_1 to be marked or for w to be set to an out-of-range value). Thus the probability that either (a) or (b) fails to hold is at most

$$j\lceil \lg|\Omega|\rceil (1 - \theta/2)^g + 2jg\lceil \lg|\Omega|\rceil \cdot \tau^\star,$$

where the first term bounds the probability that all $g\lceil \lg|\Omega|\rceil$ executions of the independence test pass for some ℓ and the second term bounds the probability that any execution of the independence test queries a point z such that $f(z) \neq f'(z)$. Finally, the probability that (c) fails to hold is at most $\epsilon_1 + \tau^\star$.

Now considering all m iterations, we have that the overall probability of either outputting "no" or obtaining a bad example in the m-element sample is at most $mj\lceil \lg|\Omega|\rceil (1 - \theta/2)^g + 2jgm\lceil \lg|\Omega|\rceil \cdot \tau^\star + (\epsilon_1 + \tau^\star)m \le 1/100 + 1/100 + 1/100$, and the lemma is proved. ■

B.3. Step 3: Checking consistency.

The final step of the algorithm, Step 3, is to run **Check-Consistency**. This step makes no queries to f.

The following two lemmata establish completeness and soundness of the overall test and conclude the proof of Theorem 4.

Check-Consistency (input is the sample S output by **Identify-Critical-Subsets**)

1. Check every function in $\mathcal{C}(\tau^\star)_{J(\tau^\star)}$ to see if any of them are consistent with sample S. If so output "yes" and otherwise output "no."

Figure 6. The subroutine Check-Consistency.

Lemma 27. *Suppose that $f \in \mathcal{C}$. Then with probability at least 2/3, algorithm $\mathcal{A}$ outputs yes.*

Proof. Let f' be some $J(\tau^\star)$-junta in $\mathcal{C}(\tau^\star)$ that is $\tau^\star$-close to f. By Corollary 25, we have that except with probability at most $1/6$, f passes **Identify-Critical-Subsets** and the inputs $I_{i_1}, \ldots, I_{i_j}$ given to **Construct-Sample** will satisfy conditions (i')-(iii'). Let $\mathcal{K}$ be the set consisting of those variables that have binary variation at least θ under f'. We use Lemma 26 to conclude that with probability at least $1 - 3/100$, **Construct-Sample** outputs m uniform, random examples labeled according to some $J(\tau^\star)$-junta g satisfying $\Pr_z[g^\sigma_\uparrow(z) \neq f'(z)] \leq \epsilon_1$. Let σ' map the variables in $\mathcal{K}$ to the same values as σ, but also map the remaining, possibly relevant variables of f' to the remaining $J(\tau^\star) - j$ bits. Clearly $\Pr_z[g^{\sigma'}_\uparrow(z) \neq f'(z)] \leq \epsilon_1$, and since the relevant variables of $g^{\sigma'}_\uparrow$ (which are contained in $\mathcal{K}$) are a subset of the relevant variables of f', we have that $\Pr_x[g(x) \neq (f')^{\sigma'}_{J(\tau^\star)}(x)] \leq \epsilon_1$.

Assuming that **Construct-Sample** outputs m uniform random examples labeled according to g, they are also labeled according to $f'^{\sigma'}_{J(\tau^\star)} \in \mathcal{C}(\tau^\star)_{J(\tau^\star)}$ except with probability at most $\epsilon_1 m$. Summing all the failure probabilities, we have that **Check-Consistency** does not output "yes" with probability at most $1/6 + 3/100\epsilon_1 m < 1/3$, and the lemma is proved. ∎

Lemma 28. *Suppose that f is ϵ-far from $\mathcal{C}$. Then the probability that algorithm $\mathcal{A}$ outputs "yes" is less than 1/3.*

Proof. We assume that f passes **Identify-Critical-Subsets** with probability greater than 1/3 (otherwise we are done), and show that if f passes **Identify-Critical-Subsets**, it will be rejected by **Construct-Sample** or **Check-Consistency** with probability at least 2/3.

Assume f passes **Identify-Critical-Subsets** and outputs $I_{i_1}, \ldots, I_{i_j}$. Using Lemma 23, we know that except with probability at most 1/7, $\mathcal{J}$, the set of variables with binary variation at least θ under f, satisfies:

- $\mathrm{BinVr}_f(\overline{\mathcal{J}}) < \epsilon_1/4$;
- each variable in $\mathcal{J}$ is contained in some bin I_{i_ℓ} that is output;
- each bin I_{i_ℓ} contains at most one variable from $\mathcal{J}$.

As in Lemma 26, we construct a function h using the variables in $\mathcal{J}$ according to Equation 6 in Section B.1. Let $\mathcal{H} \subseteq \mathcal{J}$ be the set of relevant variables for h, and let $\sigma : [n] \to [n]$ be as in Lemma 26. We have that $\Pr_{z \in \Omega^n}[h(z) \neq f(z)] \leq \epsilon_1$. We show that with probability greater than $1 - 2/100$, **Construct-Sample** either outputs "no" or a set of m uniform, random examples labeled according to $g \stackrel{\mathrm{def}}{=} h^\sigma_{J(\tau^\star)}$.

Consider a particular random draw of $z \in \Omega^n$ As in Lemma 26, this draw will yield a uniform, random example $x \in \Omega^{J(\tau^\star)}$ for g as long as

(a) for every bin I_{i_ℓ} which contains a variable from $\mathcal{H}$, Step 1(b)ii constructs the index w such that X_w contains that variable;

(b) for every bin I_{i_ℓ} that contains no variable from $\mathcal{H}$, in every iteration of Step 1(b)ii(C) at most one of Ω_0, Ω_1 is marked, and the value w that is considered in Step 1(b)iv lies in $[0, |\Omega| - 1]$; and

(c) $h(z) = f(z)$.

The probability of (c) failing is bounded by ϵ_1. The probability of (a) failing is at most $j\lceil \lg |\Omega| \rceil (1-\theta/2)^g < \frac{1}{100m}$. If neither (a) nor (c) occurs, then the example satisfies (a), (b) and (c) unless it fails to satisfy (b), but if it fails to satisfy (b) **Construct-Sample** outputs "no" in Step 1(b).ii.F or Step 1(b).iv. a Thus if f passes **Identify-Critical-Subsets**, we have that with probability at least

$$1 - 1/7 - 1/100 - \epsilon_1 m \geq 1 - 1/7 - 2/100 > 1 - 1/6$$

Construct-Sample either outputs "no" or it outputs a set of m uniform random examples for g.

Suppose **Construct-Sample** outputs such a set of examples. We claim that with probability at least $1 - 1/6$ over the choice of random examples for g, **Check Consistency** will output "no". Suppose that **Check Consistency** finds some $g' \in \mathcal{C}(\tau^\star)_{J(\tau^\star)}$ consistent with all m examples. Then g' cannot be ϵ_2-close to g. (Otherwise, we have that $\Pr_z[g'^{\sigma}_{\uparrow}(z) \neq g^{\sigma}_{\uparrow}(z)] \leq \epsilon_2$, from which it follows that $\Pr_z[g'^{\sigma}_{\uparrow}(z) \neq f(z)] \leq \epsilon_2 + \epsilon_1 < \epsilon$ since $g^{\sigma}_{\uparrow}(z)$ is ϵ_1-close to f. But $g' \in \mathcal{C}(\tau^\star)_{J(\tau^\star)}$, so $g'^{\sigma}_{\uparrow} \in \mathcal{C}(\tau^\star) \subseteq \mathcal{C}$ which contradicts our assumption that f is ϵ-far from $\mathcal{C}$.) By choice of m, the probability there exists a $g' \in \mathcal{C}(\tau^\star)_{J(\tau^\star)}$ consistent with all m examples that is not ϵ_2-close to g is at most $|\mathcal{C}(\tau^\star)_{J(\tau^\star)}|(1-\epsilon_2)^m = 1/6$. Thus, if f passes **Identify-Critical-Subsets**, then **Construct-Sample** and **Check-Consistency** output "yes" with probability less than $1/6 + 1/6 < 1/3$. This proves the lemma. ■

C. Making the algorithm non-adaptive

The algorithm $\mathcal{A}$ presented in the previous section is adaptive. In this section, we show that $\mathcal{A}$ can be made non-adaptive without considerably increasing its query complexity.

The only part of our current algorithm that fails to be non-adaptive is Step 2, the **Construct-Sample** subroutine, which relies on knowledge of the critical subsets identified in Step 1. To remove this reliance, one approach is to modify the **Construct-Sample** subroutine (in particular the `for`-loop in step 1(b)) so that it iterates over every subset rather than just the critical ones. This modified subroutine can be run before the critical subsets are even identified, and the queries it makes can be stored for future use. Later, when the critical subsets are identified, the queries made during the iterations over non-critical subsets can be ignored. Since there are $\Theta(J(\tau^\star)^2)$ total subsets compared to the $\Theta(J(\tau^\star))$ critical ones, the cost of this modified algorithm is an additional factor of $\Theta(J(\tau^\star))$ in the query complexity given in Theorem 4. For all of our applications, this translates to only a small polynomial increase in query complexity (in most cases, merely an additional factor of $\Theta(s)$).

We briefly sketch a more efficient approach to nonadaptivity; this is done essentially by combining Steps 1 and 2. Specifically, each of the m examples that we currently generate in Step 2 can be generated using the techniques from Step 1. To generate a single example, we take a random assignment to all of the variables, and we split each set I_i of variables into $|\Omega|$ sets $I_{i,\omega}$, where $I_{i,\omega}$ consists of those variables in I_i that were assigned ω. We get $\Theta(|\Omega| J(\tau^\star)^2)$ sets of variables. Now, as in the **Identify-Critical-Subsets** subroutine, we create $k = O(J(\tau^\star) \log(|\Omega| J(\tau^\star)))$ blocks, each consisting of exactly $|\Omega| J(\tau^\star)$ sets $I_{i,\omega}$ chosen at random. We run the independence test $\Theta(\frac{1}{\theta} \log(km))$ times on each of these blocks, and declare variation free those not rejected even once. If for each critical subset I_i, at least $|\Omega| - 1$ sets $I_{i,\omega}$ are declared variation free on behalf of some block, the remaining $I_{i,\omega}$ which are not declared variation free give us the values of the influential variables. One can show that this happens with probability $1 - O(1/m)$. Therefore when the procedure is repeated to generate all m examples, the probability of overall success is constant. Without going into a detailed analysis, the query complexity of this modified algorithm is essentially the same as that given in Theorem 4, namely $\tilde{O}\left(\frac{\ln |\Omega|}{\epsilon^2} J(\tau^\star)^2 \ln^2(|\mathcal{C}(\tau^\star)_{J(\tau^\star)}|)\right)$. Thus, for all of our applications, we can achieve non-adaptive testers with the same complexity bounds stated in Theorems 29 and 33.

D. Applications to Testing Classes of Functions

The algorithm $\mathcal{A}$ in Theorem 4 can be applied to many different classes of functions that were not previously known to be testable. The following two subsections state and prove our results for Boolean and non-Boolean functions, respectively.

D.1. Boolean Functions

Theorem 29. *For any s and any $\epsilon > 0$, Algorithm $\mathcal{A}$ yields a testing algorithm for*

(i) decision lists using $\tilde{O}(1/\epsilon^2)$ queries;

(ii) size-s decision trees using $\tilde{O}(s^4/\epsilon^2)$ queries;

(iii) size-s branching programs using $\tilde{O}(s^4/\epsilon^2)$ queries;

(iv) s-term DNF using $\tilde{O}(s^4/\epsilon^2)$ queries;

(v) size-s Boolean formulas using $\tilde{O}(s^4/\epsilon^2)$ queries;

(vi) size-s Boolean circuits using $\tilde{O}(s^6/\epsilon^2)$ queries;

(vii) functions with Fourier degree at most d using $\tilde{O}(2^{6d}/\epsilon^2)$ queries.

Proof. We describe each class of functions and apply Theorem 4 to prove each part of the theorem.

Decision Lists. A *decision list* L of length m is described by a list $(\ell_1, b_1), \ldots, (\ell_m, b_m), b_{m+1}$ where each ℓ_i is a Boolean literal and each b_i is an output bit. Given an input $x \in \{0,1\}^n$ the value of L on x is b_j, where $j \geq 1$ is the first value such that ℓ_j is satisfied by x. If ℓ_j is not satisfied by x for all $j = 1, \ldots, m$ then the value of $L(x)$ is b_{m+1}.

Let $\mathcal{C}$ denote the class of all Boolean functions computed by decision lists. Since only a $1/2^j$ fraction of inputs x cause the $(j+1)$-st literal ℓ_j in a decision list to be evaluated, we have that the class $\mathcal{C}(\tau) \stackrel{\text{def}}{=} \{$all functions computed by decision lists of length $\log(1/\tau)\}$ is a $(\tau, J(\tau))$-approximator for $\mathcal{C}$, where $J(\tau) \stackrel{\text{def}}{=} \log(1/\tau)$. We have $|\mathcal{C}(\tau)_{J(\tau)}| \leq 2 \cdot 4^{\log(1/\tau)}(\log(1/\tau))!$. This yields $\tau^\star = \tilde{O}(\epsilon^2)$, so Theorem 4 thus yields part (i) of Theorem 29.

Decision Trees. A *decision tree* is a rooted binary tree in which each internal node is labeled with a variable x_i and has precisely two children and each leaf is labeled with an output bit. A decision tree computes a Boolean function in the obvious way: given an input x, the value of the function on x is the output bit reached by starting at the root and going left or right at each internal node according to whether the variable's value in x is 0 or 1. The *size* of a decision tree is simply the number of leaves of the tree (which is one more than the number of internal nodes).

Let $\mathcal{C}$ denote the class of all Boolean functions computed by decision trees of size at most s. It is obvious that any size-s decision tree depends on at most s variables. We may thus take $\mathcal{C}(\tau) \stackrel{\text{def}}{=} \mathcal{C}$ and we trivially have that $\mathcal{C}(\tau)$ is a $(\tau, J(\tau))$-approximator for $\mathcal{C}$ with $J(\tau) \stackrel{\text{def}}{=} s$.

Now we bound $|\mathcal{C}(\tau)_{J(\tau)}|$ by $(8s)^s$. It is well known that the number of s-leaf rooted binary trees in which each internal node has precisely two children is the Catalan number $C_{s-1} = \frac{1}{s}\binom{2s-2}{s-1}$, which is at most 4^s. For each of these possible tree topologies there are at most s^{s-1} ways to label the $s-1$ internal nodes with variables from $x_1, \ldots, x_s$. Finally, there are precisely 2^s ways to choose the leaf labels. So the total number of decision trees of size s over variables $x_1, \ldots, x_s$ is at most $4^s \cdot s^{s-1} \cdot 2^s < (8s)^s$.

We thus have $\tau^\star = \tilde{O}(\epsilon^2/s^4)$ in Theorem 4, and we obtain part (ii) of Theorem 29.

Branching Programs. Similar results can be obtained for *branching programs.* A branching program of size s is a rooted s-node directed acyclic graph with two sink nodes labeled 0 and 1. Each internal node has fanout two (and arbitrary fan-in) and is labeled with a variable from $x_1, \ldots, x_n$. Given an input x, the value of the branching program on x is the output bit reached as described above.

Let $\mathcal{C}$ denote the class of all s-node branching programs over $\{0,1\}^n$. As with decision trees we may take $\mathcal{C}(\tau) \stackrel{\text{def}}{=} \mathcal{C}$ and $J(\tau) \stackrel{\text{def}}{=} s$. We show that $|\mathcal{C}(\tau)_{J(\tau)}| \leq s^s(s+1)^{2s}$.

The graph structure of the DAG is completely determined by specifying the endpoints of each of the two outgoing edges from each of the s internal vertices. There are at most $s+1$ possibilities for each endpoint (at most $s-1$ other internal vertices plus the two sink nodes), so there are at most $(s+1)^{2s}$ possible graph structures. There are at most s^s ways to label the s nodes with variables from $\{x_1, \ldots, x_s\}$. Thus the total number of possibilities for a size-s branching program over $x_1, \ldots, x_s$ is at most $s^s(s+1)^{2s}$.

Again we have $\tau^\star = \tilde{O}(\epsilon^2/s^4)$, so Theorem 4 yields part (iii) of Theorem 29.

DNF Formulas. An s-term DNF formula is an s-way OR of ANDs of Boolean literals. A *k-DNF* is a DNF in which each term is of length at most k.

It is well known that any s-term DNF formula over $\{0,1\}^n$ is τ-close to a $\log(s/\tau)$-DNF with at most s terms (see e.g. [15] or Lemma 30 below). Thus if $\mathcal{C}$ is the class of all s-term DNF formulas over $\{0,1\}^n$, we may take $\mathcal{C}(\tau)$ to be the class of all s-term $\log(s/\tau)$-DNF, and we have that $\mathcal{C}(\tau)$ is a $(\tau, J(\tau))$-approximator for $\mathcal{C}$ with $J(\tau) \stackrel{\text{def}}{=} s\log(s/\tau)$. An easy counting argument shows that $|\mathcal{C}(\tau)_{J(\tau)}| \leq (2s\log(s/\tau))^{s\log(s/\tau)}$. We get $\tau^\star = \tilde{O}(\epsilon^2/s^4)$, so Theorem 4 yields part (iv) of Theorem 29.

Boolean Formulas. We define a *Boolean formula* to be a rooted tree in which each internal node has arbitrarily many children and is labeled with either AND or OR and each leaf is labeled with a Boolean variable x_i or its negation $\overline{x}_i$. The size of a Boolean formula is the number of AND/OR gates it contains.

Let $\mathcal{C}$ denote the class of all Boolean formulas of size at most s. Similar to the case of DNF, we have the following easy lemma:

Lemma 30. *Any size-s Boolean formula (or size-s circuit) over $\{0,1\}^n$ is τ-close to a size-s formula (or size-s circuit) in which each gate has at most $\log(s/\tau)$ inputs that are literals.*

Proof. If a gate g has more than $\log(s/\tau)$ many inputs that are distinct literals, the gate is τ/s-approximated by a constant function (1 for OR gates, 0 for AND gates). Performing such a replacement for each of the s gates in the circuit yields a τ-approximator for the overall formula (or circuit). ■

We may thus take $\mathcal{C}(\tau)$ to be the class of all size-s Boolean formulas in which each gate has at most $\log(s/\tau)$ distinct literals among its inputs, and we have that $\mathcal{C}(\tau)$ is a $(\tau, J(\tau))$-approximator for $\mathcal{C}$ with $J(\tau) \stackrel{\text{def}}{=} s\log(s/\tau)$. An easy counting argument shows that $|\mathcal{C}(\tau)_{J(\tau)}| \leq (2s\log(s/\tau))^{s\log(s/\tau)+s}$; for each of the s gates there is a two-way choice for its type (AND or OR) and an at most s-way choice for the gate that it feeds into. There are also at most $\log(s/\tau)$ literals from $x_1, \ldots, x_{s\log(s/\tau)}, \overline{x}_1, \ldots, \overline{x}_{s\log(s/\tau)}$ that feed into the gate. Thus there are at most $(2s\log(s/\tau))^{\log(s/\tau)+1}$ possibilities for each of the s gates, and consequently at most $(2s\log(s/\tau))^{s\log(s/\tau)+s}$ possibilities overall. Again we get $\tau^\star = \tilde{O}(\epsilon^2/s^4)$, which gives part (v) of Theorem 29.

Boolean Circuits. An even broader representation scheme is that of *Boolean circuits.* A Boolean circuit of size s is a rooted DAG with s internal nodes, each of which is labeled with an AND, OR or NOT gate. (We consider circuits with arbitrary fan-in, so each AND/OR node is allowed to have arbitrarily many descendants.) Each directed path from the root ends in one of the $n+2$ sink nodes $x_1, \ldots, x_n, 0, 1$.

For $\mathcal{C}$ the class of all size-s Boolean circuits, using Lemma 30 we may take $\mathcal{C}(\tau)$ to be the class of all size-s Boolean circuits in which each gate has at most $\log(s/\tau)$ distinct literals among its inputs, and we have that $\mathcal{C}(\tau)$ is a $(\tau, J(\tau))$-approximator for $\mathcal{C}$ with $J(\tau) \stackrel{\text{def}}{=} s\log(s/\tau)$. It is easy to see that $|\mathcal{C}(\tau)_{J(\tau)}| \leq$

2^{2s^2+4s}. To completely specify a size-s Boolean circuit, it suffices to specify the following for each of the s gates: its label (three possibilities, AND/OR/NOT) and the set of nodes to which it has outgoing edges (at most 2^{2s+2} possibilities, since this set is a subset of the $s+2$ sink nodes and the s internal nodes).

This results in $\tau^\star = \tilde{O}(\epsilon^2/s^6)$, and consequently Theorem 4 yields part (vi) of Theorem 29.

Functions with bounded Fourier degree. For convenience here we take $\Omega = \{-1, 1\}$. Recall that every Boolean function $f : \{-1,1\}^n \to \{-1,1\}$ has a unique Fourier representation, i.e. a representation as a multilinear polynomial with real coefficients: $f(x) = \sum_{S \subseteq [n]} \hat{f}(S) \prod_{i \in S} x_i$. The coefficients $\hat{f}(S)$ are the *Fourier coefficients* of f. The *Fourier degree* of f is the degree of the above polynomial, i.e. the largest value d for which there is a subset $|S| = d$ with $\hat{f}(S) \neq 0$.

Let $\mathcal{C}$ denote the class of all Boolean functions over $\{-1,1\}^n$ with Fourier degree at most d. Nisan and Szegedy [11] have shown that any Boolean function with Fourier degree at most d must have at most $d2^d$ relevant variables. We thus may take $\mathcal{C}(\tau) \stackrel{\text{def}}{=} \mathcal{C}$ and $J(\tau) \stackrel{\text{def}}{=} d2^d$. The following lemma gives a bound on $|\mathcal{C}(\tau)_{J(\tau)}|$:

Lemma 31. *For any $d > 0$ we have $|\mathcal{C}(\tau)_{J(\tau)}| < 2^{d^2 \cdot 2^{2d}}$.*

Proof. We first establish the following simple claim:

Claim 32. *Suppose the Fourier degree of $f : \{-1,1\}^n \to \{-1,1\}$ is at most d. Then every nonzero Fourier coefficient of f is an integer multiple of $1/2^{d-1}$.*

Proof. Let us view $f : \{-1,1\}^n \to \{-1,1\}$ as a polynomial with real coefficients. Define the polynomial $p(x_1, \ldots, x_n)$ as

$$p(x_1, \ldots, x_n) = \frac{f(2x_1 - 1, \ldots, 2x_n - 1) + 1}{2}.$$

The polynomial p maps $\{0,1\}^n$ to $\{0,1\}$. Since f is a multilinear polynomial of degree at most d, so is p. Now it is well known that there is a unique multilinear polynomial that computes any given mapping from $\{0,1\}^n$ to $\{0,1\}$, and it is easy to see that this polynomial has all integer coefficients. Since

$$f(x_1, \ldots, x_n) = 2p\left(\frac{1+x_1}{2}, \ldots, \frac{1+x_n}{2}\right) - 1,$$

it follows that every coefficient of f is an integer multiple of $\frac{1}{2^{d-1}}$, and the claim is proved. ∎

To prove Lemma 31 we must bound the number of distinct Boolean functions with Fourier degree at most d over variables $x_1, \ldots, x_{d2^d}$. First observe that there are at most $D = \sum_{i=0}^{d} \binom{d2^d}{i} \leq (d2^d)^d$ monomials of degree at most d over these variables.

If $f : \{-1,1\}^{d2^d} \to \{-1,1\}$ has Fourier degree at most d, then by Claim 32 every Fourier coefficient is an integer multiple of $1/2^{d-1}$. Since the sum of squares of all Fourier coefficients of any Boolean function is 1, at most 2^{2d-2} of the D monomials can have nonzero Fourier coefficients, and each such coefficient takes one of at most 2^d values. Thus there can be at most

$$\binom{D}{2^{2d-2}} \cdot (2^d)^{2^{2d-2}} \leq (D2^d)^{2^{2d-2}} < 2^{d^2 \cdot 2^{2d}}$$

many Boolean functions over $x_1, \ldots, x_{d2^d}$ that have Fourier degree at most d. ∎

We thus get that $\tau^\star = \tilde{O}(\epsilon^2/2^{6d})$, and Theorem 4 yields part (vii) of Theorem 29. ∎

D.2. Non-Boolean Functions

Theorem 33. *For any s and any $\epsilon > 0$, Algorithm $\mathcal{A}$ yields a testing algorithm for*

(i) s-sparse polynomials over finite field Ω using $\tilde{O}((s|\Omega|)^4/\epsilon^2)$ queries;

(ii) size-s algebraic circuits over finite ring or field Ω using $\tilde{O}(s^4 \log^3 |\Omega|/\epsilon^2)$ queries;

(iii) size-s algebraic computation trees over finite ring or field Ω using $\tilde{O}(s^4 \log^3 |\Omega|/\epsilon^2)$ queries.

Proof. We describe class of functions and apply Theorem 4 to prove each part of the theorem.

Sparse Polynomials over Finite Fields. Let Ω denote any finite field and let $X = \Omega$. An *s-sparse polynomial over Ω* is a multivariate polynomial in variables $x_1, \ldots, x_n$ with at most s nonzero coefficients.

Let us say that the *length* of a monomial is the number of distinct variables that occur in it (so for example the monomial $3x_1^2 x_2^4$ has length two). We have the following:

Lemma 34. *Any s-sparse polynomial over Ω is τ-close to an s-sparse polynomial over Ω in which each monomial has length at most $|\Omega| \ln(s/\tau)$.*

Proof. If a monomial has length ℓ greater than $|\Omega| \ln(s/\tau)$, then it can be τ/s-approximated by 0 (for a uniform random $x \in \Omega^n$, the probability that the monomial is not 0 under x is $(1 - 1/|\Omega|)^\ell$). Performing this approximation for all s terms yields a τ-approximator for the polynomial. ■

For $\mathcal{C}$ = the class of all s-sparse polynomials in n variables over finite field Ω, we have that the class $\mathcal{C}(\tau)$ of all s-sparse polynomials over finite field Ω with all monomials of length at most $|\Omega| \ln(s/\tau)$ is a $(\tau, J(\tau))$-approximator with $J(\tau) = s|\Omega| \ln(s/\tau)$. The following counting argument shows that

$$|\mathcal{C}(\tau)_{J(\tau)}| \leq (s|\Omega|^3 \ln(s/\tau))^{s|\Omega| \ln(s/\tau)}.$$

Consider a single monomial M. To specify M we must specify a coefficient in Ω, a subset of at most ℓ of the $J(\tau)$ possible variables that have nonzero degree (at most $J(\tau)^\ell$ possibilities), and for each of these variables we must specify its degree, which we may assume is at most $|\Omega| - 1$ since $\alpha^{|\Omega|} = \alpha$ for every α in finite field Ω. Thus there are at most $|\Omega|(J(\tau)|\Omega|)^\ell$ possibilities for each monomial, and consequently at most $|\Omega|^s (J(\tau)|\Omega|)^{s\ell} = |\Omega|^s (s|\Omega|^2 \ln(s/\tau))^{s|\Omega| \ln(s/\tau)} \leq (s|\Omega|^3 \ln(s/\tau))^{s|\Omega| \ln(s/\tau)}$ possible polynomials overall.

Setting $\tau^\star = \tilde{O}(\epsilon^2/(s|\Omega|)^4)$ and applying Theorem 4 yields part (i) of Theorem 33.

Algebraic Circuits. Let Ω denote any finite ring or field and let $X = \Omega$. A size-s *algebraic circuit* (or *straight line program*) over Ω^n is a rooted directed acyclic graph with s internal nodes (each with two inputs and one output) and $n + k$ leaf nodes for some $k \geq 0$ (each with no inputs and arbitrarily many outputs). The first n leaf nodes are labeled with the input variables $x_1, \ldots, x_n$, and the last k leaf nodes are labeled with arbitrary constants α_i from Ω. Each internal node is labeled with a gate from $\{+, \times, -\}$ and computes the sum, product, or difference of its two input values (if Ω is a field we allow division gates as well).

Let $\mathcal{C}$ denote the class of all Boolean functions computed by algebraic circuits of size at most s over variables $x_1, \ldots, x_n$. (Here we analyze the simpler case of circuits with $+, \times, -$ gates; our analysis can easily be extended to handle division gates as well.) Any size-s algebraic circuit depends on at most $2s$ variables. We may thus take $\mathcal{C}(\tau) \stackrel{\text{def}}{=} \mathcal{C}$ and we trivially have that $\mathcal{C}(\tau)$ is a $(\tau, J(\tau))$-approximator for $\mathcal{C}$ with $J(\tau) \stackrel{\text{def}}{=} 2s$. Now we show that $|\mathcal{C}(\tau)_{J(\tau)}| \leq (75|\Omega|^2 s^2)^s$.

A size s algebraic circuit can read at most $2s$ leaves as each internal node has two inputs. Thus it can read at most $2s$ constant leaves, and at most $2s$ input leaves. To completely specify a size-s algebraic circuit, it suffices to specify the $2s$ constant leaf nodes and the following for each of the s gates: its label (at most three possibilities) and the two nodes to which it has outgoing edges (at most $(5s)^2$ possibilities, since it can hit two of the at most $4s$ leaves and the s internal nodes). Thus there are at most $|\Omega|^{2s}(75s^2)^s$ different algebraic circuits.

Equation 2 in Theorem 4 is satisfied for small τ's, but we do not care how large the optimum $\tau^\star$ is as $J(\tau)$ does not depend on τ. Eventually, Theorem 4 yields part (ii) of Theorem 33.

Algebraic Computation Trees. Let Ω denote any finite ring or field and let $X = \Omega$. A size-s *algebraic computation tree* over input variables $x_1, \ldots, x_n$ is a rooted binary tree with the following structure. There are s leaves, each describes an output value which is either a constant, an input variable, or one of the variables computed in the ancestors of the leaf. Each internal node has two children and is labeled with y_v, where $y_v = y_u \circ y_w$ and y_u, y_w are either inputs, the labels of ancestor nodes, or constants, and the operator $\circ$ is one of $\{+, -, \times, \div\}$ (the last one only if Ω is a field). An input that reaches such a node branches left if $y_v = 0$ and branches right if $y_v \neq 0$.

Let $\mathcal{C}$ denote the class of all functions computed by algebraic computation trees of size at most s over $x_1, \ldots, x_n$. Any size-s algebraic computation tree depends on at most $3s$ variables. So similar to algebraic circuits, we can take $\mathcal{C}(\tau) \stackrel{\text{def}}{=} \mathcal{C}$ and $J(\tau) \stackrel{\text{def}}{=} 3s$. Now we show that $|\mathcal{C}(\tau)_{J(\tau)}| \leq 16^s(|\Omega| + 4s)^{3s}$.

As in the boolean case, the number of s-leaf rooted binary trees in which each internal node has precisely two children is at most 4^s. A tree has $s-1$ internal nodes and s leaves. For each of these possible tree topologies there are at most $4(|\Omega| + 4s)^2$ ways to label the $s-1$ internal nodes (with one of 4 operations on two constants, variables or ancestor nodes). Finally, there are at most $(|\Omega| + 4s)^s$ ways to choose the leaf labels. So the total number of decision trees of size s over variables $x_1, \ldots, x_{3s}$ is at most $4^s \cdot (4(|\Omega| + 4s)^2)^{s-1} \cdot (|\Omega| + 4s)^s \leq 16^s(|\Omega| + 4s)^{3s}$.

As before we do not care what the optimal $\tau^\star$ in Theorem 4 is. Finally, we obtain query complexity $\tilde{O}(s^4 \log^3 |\Omega|/\epsilon^2)$ by Theorem 4, that is, we obtain part (iii) of Theorem 33. ■

E. Lower Bound Proofs

In this section we restate and prove the testing lower bounds discussed in Section 4. The main result in that section was Theorem 5, the lower bound for testing s-sparse polynomials over finite fields of constant size. In Subsection E.1, we prove Theorem 5. In Subsection E.2, we prove some simpler (and weaker) lower bounds for other function classes.

E.1. Lower Bound for s-Sparse Polynomials

Throughout this section we write $\mathbb{F}$ to denote the finite field with P elements, where $P = p^k$ is a prime power. For convenience, we restate Theorem 5:

Theorem 5. Let $\mathbb{F}$ be any fixed finite field, i.e. $|\mathbb{F}| = O(1)$ independent of n. There exists a fixed constant $\epsilon > 0$ (depending on $|\mathbb{F}|$) such that any *non-adaptive* ϵ-testing algorithm for the class of s-sparse polynomials over $\mathbb{F}^n$ must make $\tilde{\Omega}(\sqrt{s})$ queries.

To prove Theorem 5, we consider the following two distributions over functions mapping $\mathbb{F}^n$ to $\mathbb{F}$:

- A draw from D_{YES} is obtained as follows: independently and uniformly (with repetitions) draw s variables $x_{i_1}, \ldots, x_{i_s}$ from $x_1, \ldots, x_n$, and let $f(x) = x_{i_1} + \cdots + x_{i_s}$.
- A draw from D_{NO} is obtained as follows: independently and uniformly (with repetitions) draw $s + p$ variables $x_{i_1}, \ldots, x_{i_{s+p}}$ from $x_1, \ldots, x_n$, and let $f(x) = x_{i_1} + \cdots + x_{i_{s+p}}$.

It is clear that every draw from D_{YES} is an s-sparse polynomial over $\mathbb{F}$, and that for any $n = \omega((s+p)^2)$ almost all the probability mass of D_{NO} is on functions with $s+p$ distinct nonzero coefficients.

Theorem 5 then follows from the following two results:

Theorem 35. *Let A be any non-adaptive algorithm which is given black-box access to a function $f : \mathbb{F}^n \to \mathbb{F}$ and outputs either "yes" or "no." Then we have*

$$\left| \Pr_{f \in D_{\text{YES}}}[A^f \textit{ outputs "yes"}] - \Pr_{f \in D_{\text{NO}}}[A^f \textit{ outputs "yes"}] \right| \le \frac{1}{3}$$

unless A makes $\tilde{\Omega}(\sqrt{s})$ queries to the black-box function f.

Theorem 36. *Let*

$$\Phi(P) \stackrel{\text{def}}{=} 1/(P^{P^2+P^{10P^2+26}}).$$

Fix any $s \le n-1$. Let g be an s-sparse polynomial in $\mathbb{F}[x_1, \dots, x_n]$. Then g is $\Phi(P)$-far from every affine function over $\mathbb{F}$ in which $s+1$ or more variables have nonzero coefficients, i.e. every function of the form

$$a_1 x_1 + \cdots + a_{s+r} x_{s+r} + b \tag{7}$$

where $0 \ne a_i \in \mathbb{F}$, $b \in \mathbb{F}$, and $r \ge 1$.

Theorem 35 shows that any non-adaptive algorithm that can successfully distinguish a random linear form $x_{i_1} + \cdots + x_{i_s}$ from a random linear form $x_{i_1} + \cdots + x_{i_{s+p}}$ must make $\tilde{\Omega}(\sqrt{s})$ queries; this is a technical generalization of a similar result for $\mathbb{F}_2$ in [6]. Theorem 36 establishes that every function $x_{i_1} + \cdots + x_{i_{s+p}}$ is far from every s-sparse polynomial over $\mathbb{F}$. Together these results imply that any testing algorithm for s-sparse $\mathbb{F}$ polynomials must be able to distinguish length-s linear forms from length-$(s+p)$ linear forms, and must make $\tilde{\Omega}(\sqrt{s})$ queries. We prove these theorems in the following subsections.

We note that it is conceivable that a stronger version of Theorem 36 might be true in which $\Phi(P)$ is replaced by an absolute constant such as $1/3$; however Theorem 36 as stated suffices to give our desired lower bound.

E.1.1 Proof of Theorem 35.

First, let us recall the definition of statistical distance:

Definition 37 (statistical distance)**.** *Let S be a finite set and $\mathbb{P}, \mathbb{Q}$ be probability measures on $(S, 2^S)$. The* statistical distance *between $\mathbb{P}$ and $\mathbb{Q}$ is defined by $\|\mathbb{P} - \mathbb{Q}\| \stackrel{\text{def}}{=} \max_{A \subseteq S} |\mathbb{P}(A) - \mathbb{Q}(A)|$.*

The following fact is an immediate consequence of the definition:

Fact 38. $\|\mathbb{P} - \mathbb{Q}\| \equiv \frac{1}{2} \sum_{x \in S} |\mathbb{P}(x) - \mathbb{Q}(x)| \equiv \sum_{x \in S} \left(\mathbb{P}(x) - \mathbb{Q}(x)\right)^{\pm}$.

We now explain how Theorem 35 can be reduced to a convergence-type result about random walks on the group $\mathbb{Z}_p^q$ (Theorem 6). We remark that the argument given here is an immediate generalization of the corresponding argument in Section 6 of [6]. Our main technical contribution is in fact the proof of Theorem 6.

Recall that a non-adaptive testing algorithm queries a fixed subset $\mathcal{Q}$ of the domain $\mathbb{F}^n$, where $|\mathbb{F}| = P = p^k$ is a prime power. To prove Theorem 35, it suffices to argue that for any query set $\mathcal{Q} \subset \mathbb{F}^n$ of cardinality $q = |\mathcal{Q}| = \tilde{O}(\sqrt{s})$ the induced distributions on $\mathbb{F}^q$ (obtained by restricting the randomly chosen functions to these q points) have a statistical distance less than $1/3$.

Let us now describe the distributions induced by D_{YES} and D_{NO} on $\mathbb{F}^q$. Let $r_1, r_2, \ldots, r_q \in \mathbb{F}^n$ be the queries, and let M be a $q \times n$ matrix with rows $r_1, \ldots, r_q$. To choose an element $x \in \mathbb{F}^q$ according to the first (induced) distribution, we choose at random (with repetitions) s columns of M and sum them up. This gives us an element of $\mathbb{F}^q$. The same holds for the second distribution, the only difference being that we choose $s + p$ columns.

For $x \in \mathbb{F}^q \cong \mathbb{Z}_p^{kq}$, let $\mathbb{P}(x)$ be the probability of choosing x when we pick a column of M at random. Consider a random walk on the group $\mathbb{Z}_p^{kq}$, starting at the identity element, in which at every step we choose an element of the group according to $\mathbb{P}$ and add it to the current location. Let $\mathbb{P}_t$ be the distribution of this walk after t steps. Observe that $\mathbb{P}_s$ and $\mathbb{P}_{s+p}$ are exactly the distributions induced by D_{YES} and D_{NO}. We want to show that for s sufficiently large compared to q, the distributions $\mathbb{P}_s$ and $\mathbb{P}_{s+p}$ are close with respect to the statistical distance. To do this, it suffices to prove the following theorem (restated from Section 4):

Theorem 6. *Let r be a prime, $q \in \mathbb{N}$ and $\mathbb{P}$ be a probability measure on the additive group $\mathbb{Z}_r^q$. Consider the random walk X on $\mathbb{Z}_r^q$ with step distribution $\mathbb{P}$. Let $\mathbb{P}_t$ be the distribution of X at step t. There exists an absolute constant $C > 0$ such that for every $0 < \delta \leq 1/2$, if $t \geq C\frac{\log 1/\delta}{\delta} \cdot r^4 \log r \cdot q^2 \log^2(q+1)$ then $\|\mathbb{P}_t - \mathbb{P}_{t+r}\| \leq \delta$.*

Indeed, since the underlying additive group of the field $\mathbb{F}$ is $\mathbb{Z}_p^k$, by applying the above theorem for $r = p$ and $q' = kq$ the result follows. We prove Theorem 6 in the following subsection.

E.1.2 Proof of Theorem 6.

To prove Theorem 6, we start with some basic definitions and facts about random walks on (finite) groups. For a detailed treatment of the subject, see [4] and references therein. For basic facts about Fourier Analysis on finite groups, see [13, 14].

Let $(G, +)$ be a finite group. For any probability measures $\mathbb{P}, \mathbb{Q}$ on G, the convolution $(\mathbb{P} * \mathbb{Q})$ of $\mathbb{P}$ and $\mathbb{Q}$ is the probability measure on G defined by:

$$(\mathbb{P} * \mathbb{Q})(y) = \sum_{x \in G} \mathbb{P}(x)\mathbb{Q}(x + y)$$

Let $\mathbb{P}_1, \ldots, \mathbb{P}_n$ be probability measures on G. The *convolution product* of the $\mathbb{P}_i$'s, is defined as follows:

$$\begin{aligned} &\{* \textstyle\prod\}_{i=j}^{j} \mathbb{P}_i \stackrel{\text{def}}{=} \mathbb{P}_j \\ &\{* \textstyle\prod\}_{i=j}^{n} \mathbb{P}_i \stackrel{\text{def}}{=} \mathbb{P}_j * \{* \textstyle\prod\}_{i=j+1}^{n} \mathbb{P}_i, \quad \text{if } n > j \end{aligned}$$

Similarly, $\mathbb{P}^{*n}$, *the n-fold convolution product of $\mathbb{P}$ with itself* is defined by: $\mathbb{P}^{*1} \stackrel{\text{def}}{=} \mathbb{P}$ and $\mathbb{P}^{*n} \stackrel{\text{def}}{=} \mathbb{P}^{*(n-1)} * \mathbb{P}$, if $n > 1$.

A distribution (probability measure) $\mathbb{P}$ on G induces a random walk on G as follows: Denoting by X_n its position at time n, the walk starts at the identity element of G ($n = 0$) and at each step selects an element $\xi_n \in G$ according to $\mathbb{P}$ and goes to $X_{n+1} = \xi_n + X_n$. Denote by $\mathbb{P}_n$ the distribution of X_n. Since X_n is the sum of n independent random variables with distribution $\mathbb{P}$, it follows that $\mathbb{P}_n = \mathbb{P}^{*n}$.

We will be interested in such random walks on *finite abelian groups* and in particular on the group $(\mathbb{Z}_r^q, +)$, where $+$ denotes componentwise addition modulo r. We remark that for abelian groups, the convolution operation is commutative. In fact, commutativity is crucially exploited in the proof of the theorem.

For a function $f : \mathbb{Z}_r^q \to \mathbb{C}$, we define its Fourier transform $\widehat{f} : \mathbb{Z}_r^q \to \mathbb{C}$ by

$$\widehat{f}(x) \stackrel{\text{def}}{=} \frac{1}{r^q} \sum_{y \in \mathbb{Z}_r^q} f(y)(\omega_r)^{\langle x,y \rangle}$$

where $\omega_r \stackrel{\text{def}}{=} e^{2\pi i/r}$ and for $x, y \in \mathbb{Z}_r^q$ we denote $\langle x, y \rangle \stackrel{\text{def}}{=} (\sum_{i=1}^q x_i y_i) \mod r$.

Fact 39. *Let $\mathbb{P}$, $\mathbb{Q}$ be probability measures on $\mathbb{Z}_r^q$. Then, $\widehat{\mathbb{P} * \mathbb{Q}}(y) = r^q \cdot \widehat{\mathbb{P}}(y) \cdot \widehat{\mathbb{Q}}(y)$, $y \in \mathbb{Z}_r^q$.*

For $p \geq 1$ and $f : \mathbb{Z}_r^q \to \mathbb{C}$, the l_p norm of f is defined by $\|f\|_p \stackrel{\text{def}}{=} \{\mathbb{E}_{x \in \mathbb{Z}_r^q}[|f(x)|^p]\}^{1/p}$. The inner product of $f, g : \mathbb{Z}_r^q \to \mathbb{C}$ is defined as: $\langle f, g \rangle \stackrel{\text{def}}{=} \mathbb{E}_{x \in \mathbb{Z}_r^q}[f(x)\overline{g(x)}]$.

Fact 40 (Parseval's identity)**.** *Let $f : \mathbb{Z}_r^q \to \mathbb{C}$. Then, $\|f\|_2^2 \equiv \langle f, f \rangle = \sum_{x \in \mathbb{Z}_r^q} |\hat{f}|^2(x)$.*

Proof of Theorem 6.

The special case of this theorem for $r = 2$ was proved by Fischer *et al.* [6]. Our proof is a technical generalization of their proof. Moreover, our proof has the same overall structure as the one in [6]. However, one needs to overcome several difficulties in order to achieve this generalization.

We first give a high-level overview of the overall strategy. Any given $x \in (\mathbb{Z}_r^q)^*$ partitions the space into r non-empty subspaces $V_i^x = \{y \in \mathbb{Z}_r^q : \langle y, x \rangle = i\}$ for $i = 0, 1, \ldots, r-1$. We say that an $x \in (\mathbb{Z}_r^q)^*$ is *degenerate* if there exists some i whose probability measure $\mathbb{P}(V_i^x)$ is "large". (We note that the definition of degeneracy in the proof of [6] is quite specialized for the case $r = 2$. They define a direction to be degenerate if one of the subspaces V_0^x, V_1^x has "small" probability. Our generalized notion - that essentially reduces to their definition for $r = 2$ - is the conceptually correct notion and makes the overall approach work.)

We consider two cases: If all the Fourier coefficients of $\mathbb{P}$ are not "very large" (in absolute value), then we can show by standard arguments (see e.g. [4]) that the walk is close to stationarity after the desired number of steps. Indeed, in such a case the walk converges rapidly to the uniform distribution (in the "classical" sense, i.e. $\|\mathbb{P}_t - \mathcal{U}\| \to 0$ as t approaches infinity).

If, on the other hand, there exists a "very large" Fourier coefficient of $\mathbb{P}$, then we argue that there must also exist a degenerate direction (this is rather non-trivial) and we use induction on the dimension q. It should be noted that in such a case the walk *may not converge at all in the classical sense*. (An extreme such case would be, for example, if $\mathbb{P}$ was concentrated on one element of the group.)

Remark: It seems that our proof can be easily modified to hold for any *finite abelian group*. (We remind the reader that any such group can be uniquely expressed as the direct sum of cyclic groups.) Perhaps, such a result would be of independent interest. We have not attempted to do so here, since it is beyond the scope of our lower bound. Note that, with the exception of the inductive argument, all the other components of our proof work (in this generalized setting) without any changes. It is very likely that a more complicated induction would do the trick.

Now let us proceed with the actual proof. We make essential use of two lemmata. The first one is a simple combinatorial fact that is used several times in the course of the proof:

Lemma 41. *Let n be a positive integer greater than 1 and $\epsilon \in (0, 1/2]$ be a constant. Consider a complex number $\mathbf{v} \in \mathbb{C}$ expressible as a (non-trivial) convex combination of the n-th roots of unity all of whose coefficients are at most $1 - \epsilon$. Then, we have $|\mathbf{v}| \leq 1 - \epsilon/2n^2$.*

Proof. We can write $\mathbf{v} = \sum_{j=0}^{n-1} v_j \omega_n^j$, with $\omega_n = e^{2\pi i/n}$, $v_j \geq 0$, $\sum_{j=0}^{n-1} v_j = 1$ and $\max_j v_j \leq 1 - \epsilon$. For the proof it will be helpful to view the ω_n^j's as unit vectors in the complex plane (the angle between two "adjacent" such vectors being $\theta_n = 2\pi/n$).

By assumption, it is clear that at least two distinct coefficients must be non-zero. We claim that the length of the vector $\mathbf{v}$ is maximized (over all possible "legal" choices of the v_j's) when exactly two of the coefficients are non-zero, namely two coefficients corresponding to consecutive n-th roots of unity.

This is quite obvious, but we give an intuitive argument. We can assume that $n \geq 5$; otherwise the claim is straightforward. Consider the unit vector $\mathbf{e}$ (this vector corresponds to one of the ω_n^j's) whose coefficient $v_{\mathbf{e}}$ in $\mathbf{v}$ is maximum. We want to "distribute" the remaining "mass" $1 - v_{\mathbf{e}}$ to the other coordinates (n-th roots) so as to maximize the length $|\mathbf{v}|$. First, observe that vectors whose angle with $\mathbf{e}$ is at least $\pi/2$ do not help; so we can assume the corresponding coefficients are zero. Now consider the set of vectors "above" $\mathbf{e}$ (whose angle with $\mathbf{e}$ is less than $\pi/2$). We can assume that their "mass" (i.e. sum of coefficients) is concentrated on the unit vector $\mathbf{e}_a$ adjacent to $\mathbf{e}$ (whose angle with $\mathbf{e}$ is minimum); this maximizes their total contribution to the length of the sum. By a symmetric argument, the same holds for the set of vectors "below" $\mathbf{e}$ (denote by $\mathbf{e}_b$ the corresponding adjacent vector). Finally, it is easy to see that in order to maximize the total contribution of $\mathbf{e}_a$ and $\mathbf{e}_b$ to the length of the sum, one of them must have zero weight (given that their total mass is "fixed").

Now let us proceed with the proof of the upper bound. By symmetry, it is no loss of generality to assume that $v_0, v_1 > 0$ with $v_0 \geq v_1$. The claim now follows from the following sequence of elementary calculations:

$$\begin{aligned}
|\mathbf{v}|^2 = v_0^2 + v_1^2 + 2v_0v_1\cos\theta_n &= 1 - 2v_0v_1\big(1 - \cos\theta_n\big) \\
&= 1 - 2v_0\big(1 - v_0\big)\big(1 - \cos(2\pi/n)\big) \\
&\leq 1 - 2\epsilon(1 - \epsilon)\big(1 - \cos(2\pi/n)\big) \\
&\leq 1 - \epsilon\big(1 - \cos(2\pi/n)\big) \\
&\leq 1 - \epsilon/n^2
\end{aligned}$$

The last inequality above follows by observing that $\cos(2\pi/n) \leq 1 - 1/n^2$, $n \geq 2$. The elementary inequality $\sqrt{1-x} \leq 1 - x/2$ completes the argument. ■

Our second lemma is an analytical tool giving a (relatively sharp) upper bound on the statistical distance between two distributions. It should be noted that this result is a variant of the "upper bound lemma" [4], which has been used in numerous other random walk problems.

Lemma 42 (upper bound lemma, [4])**.** *In the context of Theorem 6, for any $t \geq 0$, we have:*

$$\|\mathbb{P}_t - \mathbb{P}_{t+r}\|^2 \leq r^q \cdot \sum_{x \in (\mathbb{Z}_r^q)^*} |\alpha(x)|^{2t}.$$

Proof. We have:

$$
\begin{aligned}
\|\mathbb{P}_t - \mathbb{P}_{t+r}\|^2 &= (r^{2q}/4) \cdot \|\mathbb{P}_t - \mathbb{P}_{t+r}\|_1^2 & (8)\\
&\leq (r^{3q}/4) \cdot \|\mathbb{P}_t - \mathbb{P}_{t+r}\|_2^2 & (9)\\
&= (r^{3q}/4) \cdot \sum_{x \in \mathbb{Z}_r^q} \left|\widehat{\mathbb{P}}_t(x) - \widehat{\mathbb{P}_{t+r}}(x)\right|^2 & (10)\\
&= (r^{3q}/4) \cdot \sum_{x \in (\mathbb{Z}_r^q)^*} \left|r^{q(t-1)}\left(\widehat{\mathbb{P}}(x)\right)^t - r^{q(t+r-1)}\left(\widehat{\mathbb{P}}(x)\right)^{t+r}\right|^2 & (11)\\
&= (r^q/4) \sum_{x \in (\mathbb{Z}_r^q)^*} \left|\alpha^t(x) - \alpha^{t+r}(x)\right|^2 & (12)\\
&\leq r^q \sum_{x \in (\mathbb{Z}_r^q)^*} |\alpha(x)|^{2t} & (13)
\end{aligned}
$$

Step (8) follows directly from the definitions of the statistical distance and the l_1 norm. Step (9) easily follows from the Cauchy-Schwarz inequality and step (10) from the Parseval identity. For Step (11) notice that $\widehat{\mathbb{P}}_t(y) = r^{q(t-1)}\left(\widehat{\mathbb{P}}(y)\right)^t$ and $\widehat{\mathbb{P}}(\mathbf{0}) = 1/r^q$. Step (12) is immediate by the definition of α and Step (13) follows from the triangle inequality. ■

Let $X_t \in \mathbb{Z}_r^q$ be the position of the random walk at time t and $\mathbb{P}_t$ its distribution. By assumption $X_0 = 0$. As previously mentioned, $\mathbb{P}_t = \mathbb{P}^{*t}$. It is easy to show that the statistical distance $||\mathbb{P}_t - \mathbb{P}_{t+r}||$ is monotone non-increasing in t; we are interested in the first time $t = t(r, q)$ for which $\mathbb{P}_t$ and $\mathbb{P}_{t+r}$ are δ-close.

Notation. For $q \in \mathbb{N}$, define $b(q) \stackrel{\text{def}}{=} q^2 \log^2(q+1)$, $d(r) \stackrel{\text{def}}{=} r^4 \log r$, $S_q \stackrel{\text{def}}{=} \sum_{j=1}^q j/b(j)$, $S \stackrel{\text{def}}{=} \lim_{j\to\infty} S_j$ and $t_q \stackrel{\text{def}}{=} C\frac{\log(1/\delta)}{\delta} d(r) b(q)$.

Throughout the proof, we assume for simplicity that t_q is an integer. If $\mathbb{P}$ is a probability measure on $\mathbb{Z}_r^q$ and $\widehat{\mathbb{P}}$ is its Fourier transform, we denote $\alpha(x) \stackrel{\text{def}}{=} r^q \widehat{\mathbb{P}}(x)$. A word concerning absolute constants. The letter C will always denote an absolute constant, but as is customary the value of C need not be the same in all its occurrences. Also note that S is an absolute constant, so C can depend on S.

Theorem 6 follows from the following claim:

Claim 43. *There exists a universal constant $C > 0$ such that for any $0 < \delta \leq 1/2$, any $t \geq t_q$ and any probability measure $\mathbb{P}$ on $\mathbb{Z}_r^q$ it holds $||\mathbb{P}_t - \mathbb{P}_{t+r}|| \leq \frac{\delta}{S} \cdot S_q < \delta$.*

We will prove the claim by induction on q.

Base case ($q = 1$). Given an arbitrary probability measure $\mathbb{P}$ on the discrete circle $\mathbb{Z}_n$, $n \in \mathbb{N}^*$, we will show that, for all $t \geq t_1 \equiv C\frac{\log 1/\delta}{\delta} \cdot n^4 \log n$, it holds $\|\mathbb{P}_t - \mathbb{P}_{t+n}\| \leq \frac{\delta}{S}$.

Set $\epsilon_0 := \frac{\delta}{Sn}$ and consider the following two cases below:

Case I (There exists a $k \in \mathbb{Z}_n$ such that $\mathbb{P}(k) \geq 1 - \epsilon_0$.) In this case, we claim that for all $t \in \mathbb{N}^*$ it holds $\|\mathbb{P}_t - \mathbb{P}_{t+n}\| \leq n\epsilon_0 = \delta/S$. (In fact, this holds independently of the value of the time t.) This should be intuitively obvious, but we give an argument.

Recall that the statistical distance $\|\mathbb{P}_t - \mathbb{P}_{t+c}\|$ is a monotone non-increasing function of t for any constant c. Hence, $\|\mathbb{P}_t - \mathbb{P}_{t+n}\| \leq \|\mathbb{P} - \mathbb{P}_{n+1}\|$ and it suffices to argue that $\|\mathbb{P} - \mathbb{P}_{n+1}\| \leq n\epsilon_0$. The crucial fact is that for all $i \in \mathbb{Z}_n$ we have $\mathbb{P}_{n+1}(i) \geq (1 - n\epsilon_0) \cdot \mathbb{P}(i)$. This directly implies that $\|\mathbb{P} - \mathbb{P}_{n+1}\| \equiv \sum_{i \in \mathbb{Z}_n} (\mathbb{P}(i) - \mathbb{P}_{n+1}(i))^+ \leq n\epsilon_0 \cdot \sum_{\{i : \mathbb{P}(i) > \mathbb{P}_{n+1}(i)\}} \mathbb{P}(i) \leq n\epsilon_0 \cdot \sum_{i \in \mathbb{Z}_n} \mathbb{P}(i) = n\epsilon_0$.

To see that the aforementioned fact is true, observe that for any $i \in \mathbb{Z}_n$, conditioned on the walk being at position i at time $t = 1$, with probability at least $(1 - \epsilon)^n$ each of the next n steps is k, so with probability at least $(1 - \epsilon_0)^n \geq 1 - n\epsilon_0$ the walk is at position i again at time $t = n + 1$.

Case II (For all $k \in \mathbb{Z}_n$ it holds $\mathbb{P}(k) \leq 1 - \epsilon_0$.) Note that, for $k \in \mathbb{Z}_n$, we can write $\alpha(k) = \sum_{l=0}^{n-1} \mathbb{P}(l) \cdot \omega_n^{k \cdot l}$, where $\omega_n = e^{2\pi i/n}$. Since $\mathbb{P}$ is a probability measure, it follows that $\alpha(0) = 1$. Now observe that for $k \in \mathbb{Z}_n^*$, $\alpha(k)$ is a convex combination of n-th roots of unity with coefficients at most $1 - \epsilon_0$. Hence, an application of Lemma 41 gives the following corollary:

Corollary 44. *For all $k \in \mathbb{Z}_n^*$, it holds $|\alpha(k)| \leq 1 - \frac{\delta}{2Sn^3}$.*

We have now set ourselves up for an application of Lemma 42. For any $t \in \mathbb{N}$ with $t \geq t_1$, we thus get:

$$
\begin{aligned}
\|\mathbb{P}_t - \mathbb{P}_{t+n}\|^2 &\leq n \sum_{i \in \mathbb{Z}_n^*} |\alpha(i)|^{2t} \\
&\leq n^2 \Big(1 - \frac{\delta}{2Sn^3}\Big)^{2t} \leq n^2 \Big(1 - \frac{\delta}{2Sn^3}\Big)^{2t_1} \\
&\leq n^2 (e^{-\frac{\delta}{2Sn^3}})^{2t_1} = n^2 e^{-Cn \log n \log(1/\delta)/S}
\end{aligned}
$$

where we used the elementary inequality $1 - x \leq e^{-x}$, for $x \in [0, 1]$. For large enough C, we have $\|\mathbb{P}_t - \mathbb{P}_{t+n}\|^2 \leq (\delta/S)^2$ and the base case is proved.

Induction Step: Assume that the claim holds for $q - 1$, i.e. that for any $t \geq t_{q-1}$ and any probability measure $\mathbb{P}$ on $\mathbb{Z}_r^{q-1}$ it holds $\|\mathbb{P}_t - \mathbb{P}_{t+r}\| \leq \frac{\delta}{S} \cdot S_{q-1}$. We will prove that the claim also holds for q.

For $x \in (\mathbb{Z}_r^q)^*$ and $i = 0, 1, \ldots, r - 1$ define $V_i^x \stackrel{\text{def}}{=} \{y \in \mathbb{Z}_r^q : \langle y, x \rangle = i\}$. At this point we are ready to formally define the notion of degenerate direction:

Definition 45. *We say that $x \in (\mathbb{Z}_r^q)^*$ is a* degenerate direction *if there exists an $i \in \{0, 1, \ldots, r - 1\}$ such that $\mathbb{P}(V_i^x) \geq 1 - \frac{2\delta q}{\sqrt{C} r^2 b(q)}$.*

We distinguish the following two cases below:

Case I (For all $x \in (\mathbb{Z}_r^q)^*$ it holds $|\alpha(x)| < 1 - \frac{\delta q}{\sqrt{C} r^4 b(q)}$.) Note that, since $\mathbb{P}$ is a probability distribution, we have $\alpha(\mathbf{0}) = 1$. Now, for $t \geq t_q$ Lemma 42 yields:

$$
\begin{aligned}
\|\mathbb{P}_t - \mathbb{P}_{t+r}\|^2 &\leq r^q \sum_{x \in (\mathbb{Z}_r^q)^*} |\alpha(x)|^{2t} \\
&\leq r^{2q} \Big(1 - \frac{\delta q}{\sqrt{C} r^4 b(q)}\Big)^{2t} \leq r^{2q} \Big(1 - \frac{\delta q}{\sqrt{C} r^4 b(q)}\Big)^{2t_q} \\
&\leq r^{2q} (e^{-\frac{\delta q}{\sqrt{C} r^4 b(q)}})^{2t_q} = r^{2q} e^{-2q \log r \sqrt{C} \log 1/\delta}
\end{aligned}
$$

Similarly, if C is large enough, we have $\|\mathbb{P}_t - \mathbb{P}_{t+r}\| \leq \delta/S \leq \frac{\delta}{S} \cdot S_q$.

Case II (There exists some $x_0 \in (\mathbb{Z}_r^q)^*$ such that $|\alpha(x_0)| \geq 1 - \frac{\delta q}{\sqrt{C} r^4 b(q)}$.)

Since r is a prime, we may assume without loss of generality that $x_0 = \varepsilon_1 = (1\mathbf{0}_{q-1})$. Then, for $i = 0, 1, \ldots, r-1$, we have $V_i \equiv V_i^{x_0} = \{y \equiv (y_1, y_2, \ldots, y_q) \in \mathbb{Z}_r^q : y_1 = i\}$; note that each V_i is isomorphic to $\mathbb{Z}_r^{q-1}$.

Now observe that we can write $\alpha(x_0) = \sum_{j=0}^{r-1} \mathbb{P}(V_j)\omega_r^j$ with $\sum_j \mathbb{P}(V_j) = 1$, $\mathbb{P}(V_j) \geq 0$. That is, $\alpha(x_0)$ is a convex combination of r-th roots of unity whose absolute value is at least $1 - \epsilon'/2r^2$, where $\epsilon' := \frac{2\delta q}{\sqrt{C}r^2 b(q)}$. Thus, (the contrapositive of) Lemma 41 implies that there must exist some $j \in \{0, 1, \ldots, r-1\}$ with $\mathbb{P}(V_j) \geq 1 - \frac{2\delta q}{\sqrt{C}r^2 b(q)}$ (i.e. x_0 is degenerate). Clearly, it is no loss of generality to assume that $j = 0$, i.e. $\mathbb{P}(V_0) \geq 1 - \frac{2\delta q}{\sqrt{C}r^2 b(q)}$.

For $i = 0, 1, \ldots, r-1$ and $j = t_q, t_q + r$, consider the conditional probability measures $\mathbb{P}_j^i = (\mathbb{P}_j | V_i)$. All the $2r$ distributions obtained in this manner can be viewed as distributions on $\mathbb{Z}_r^{q-1}$. By the law of total probability, we can write: $\mathbb{P}_j = \sum_{i=0}^{r-1} \mathbb{P}_j(V_i) \cdot \mathbb{P}_j^i$.

Since $\mathbb{P}(V_0) \geq 1 - \frac{2\delta q}{\sqrt{C}r^2 b(q)}$, it follows that $|\mathbb{P}_t(V_i) - \mathbb{P}_{t+r}(V_i)| \leq \frac{2\delta q}{\sqrt{C}rb(q)}$, for all $i \in \{0, 1, \ldots, r-1\}$. (In fact, this holds independently of the value of the time t). This can be shown by an argument similar to that in Case I of the induction basis.

We will show using the induction hypothesis that for $i = 0, 1, \ldots, r-1$ and $t \geq t_q$ it holds:

$$||\mathbb{P}_t^i - \mathbb{P}_{t+r}^i|| \leq \frac{\delta}{S} \cdot \left(S_{q-1} + \frac{q}{2b(q)}\right)$$

We claim that this will conclude the proof. This follows from the following chain of inequalities:

$$\begin{aligned}
\|\mathbb{P}_t - \mathbb{P}_{t+r}\| &\leq \sum_{i=0}^{r-1} |\mathbb{P}_t(V_i) - \mathbb{P}_{t+r}(V_i)| + \Big\| \sum_{i=0}^{r-1} \mathbb{P}_t(V_i) \cdot (\mathbb{P}_t^i - \mathbb{P}_{t+r}^i) \Big\| && (14) \\
&\leq \frac{2\delta q}{\sqrt{C}b(q)} + \frac{\delta}{S}\Big(S_{q-1} + \frac{q}{2b(q)}\Big) && (15) \\
&\leq \frac{\delta}{S} S_q && (16)
\end{aligned}$$

Step (14) follows easily from the triangle inequality (recall that the statistical distance is a norm) and by using the fact that the $\mathbb{P}_j^i$'s are distributions. For Step (15) observe that the second summand in (14) is a convex combination and Step (16) assumes that C is large enough.

To finish the proof we show that $\|\mathbb{P}_t^0 - \mathbb{P}_{t+r}^0\| \leq \frac{\delta}{S} \cdot \left(S_{q-1} + \frac{q}{2b(q)}\right)$. The proofs for the $r-1$ remaining cases are very similar.

For $i = 0, 1, \ldots, r-1$ denote $\mathbb{P}^i = (\mathbb{P}|V_i)$. Let $\mathbf{N}_j = (N_j^1, \ldots, N_j^{r-1})$ be a random vector such that the random variable N_j^l ($l = 1, 2, \ldots, r-1$) counts the number of times the walk makes a step $x \in \mathbb{Z}_r^q$ with $x_1 = l$ during the first j steps. Consider a vector $\mathbf{s} = (s_1, s_2, \ldots, s_{r-1})$ such that $|\mathbf{s}| \stackrel{\text{def}}{=} \sum_{i=1}^{r-1} s_i \leq j$ and $\sum_{k=1}^{r-1} k s_k \equiv 0 \mod r$. Then, we have:

$$(\mathbb{P}_j^0 | \mathbf{N}_j = \mathbf{s}) = \left(\{* \prod\}_{i=1}^{r-1} (\mathbb{P}^i)^{*s_i}\right) * (\mathbb{P}^0)^{*(j-|\mathbf{s}|)}$$

where by $\{* \prod\}$ we denote the convolution product. The above equality holds for the following reason: The distribution on the left hand side is the distribution on $V_0 \cong \mathbb{Z}_r^{q-1}$ given that the walk makes s_l steps x with $x_1 = l$ ($l = 1, 2, \ldots, r-1$) (and $j - |\mathbf{s}|$ steps with $x_1 = 0$). The equality follows by commutativity.

Therefore, by the law of total probability, we can write $\mathbb{P}_j^0$ as the following convex combination of conditional distributions:

$$\mathbb{P}_j^0 = \sum_{(\sum_{k=1}^{r-1} ks_k \equiv 0 \mod r) \text{ and } (|\mathbf{s}| \leq j)} \Pr[\mathbf{N}_j = \mathbf{s}] \cdot \left(\{* \prod\}_{i=1}^{r-1} (\mathbb{P}^i)^{*s_i}\right) * (\mathbb{P}^0)^{*(j-|\mathbf{s}|)}$$

Using this fact, we can bound $\|\mathbb{P}_t^0 - \mathbb{P}_{t+r}^0\|$ for $t = t_q$ as follows:

$$\begin{aligned} \|\mathbb{P}_t^0 - \mathbb{P}_{t+r}^0\| &\leq \Pr[\mathbf{N}_t \neq \mathbf{N}_{t+r}] + \Pr[|\mathbf{N}_t| \geq 4qr^2 \log r\sqrt{C} \log(1/\delta)] \\ &+ \sum_{\substack{\mathbf{s} \text{ such that} \\ (\sum_{k=1}^{r-1} ks_k \equiv 0 \mod r) \\ (|\mathbf{s}| \leq 4qr^2 \log r\sqrt{C}\log(1/\delta))}} \Pr[\mathbf{N}_t = \mathbf{s}] \cdot \left\| \left(\{* \prod\}_{i=1}^{r-1} (\mathbb{P}^i)^{*s_i}\right) * [(\mathbb{P}^0)^{*(t-|\mathbf{s}|)} - (\mathbb{P}^0)^{*(t+r-|\mathbf{s}|)}] \right\| \end{aligned}$$

The first summand is equal to the probability that a non-trivial step in the first coordinate (i.e step x with $x_1 \neq 0$) was made in one of the times $t+1, \ldots, t+r$ and this is at most $2\delta q/\sqrt{C} rb(q)$ (because $\mathbb{P}(V_0) \geq 1 - 2\delta q/\sqrt{C}r^2 b(q)$).

To upper bound the second summand, we observe that $|\mathbf{N}_t| = \sum_{i=1}^{r-1} N_t^i$ is a binomial random variable with parameters $t = t_q$ and $p \leq 2\delta q/\sqrt{C}r^2 b(q)$. Thus, by a standard Chernoff bound, we get that the second summand is also very small, so that the sum of the first two summands is at most $\frac{\delta}{S} \cdot \frac{q}{2b(q)}$ for large enough C.

Now consider the third summand. Since $|\mathbf{s}| \leq 4qr^2 \log r\sqrt{C}\log(1/\delta)$, it follows that $t_q - |\mathbf{s}| \geq t_{q-1}$ and the induction hypothesis implies:

$$\begin{aligned} \left\| \left(\{* \prod\}_{i=1}^{r-1} (\mathbb{P}^i)^{*s_i}\right) * [(\mathbb{P}^0)^{*(t-|\mathbf{s}|)} - (\mathbb{P}^0)^{*(t+r-|\mathbf{s}|)}] \right\| &\leq \left\| (\mathbb{P}^0)^{*(t-|\mathbf{s}|)} - (\mathbb{P}^0)^{*(t+r-|\mathbf{s}|)} \right\| \\ &\leq \frac{\delta}{S} \cdot S_{q-1} \end{aligned}$$

The first inequality follows from the fact that $\{* \prod\}_{i=1}^{r-1} (\mathbb{P}^i)^{*s_i}$ is a distribution. Therefore, the expression $\frac{\delta}{S} \cdot S_{q-1}$ is an upper bound for the third summand and the proof is complete.

E.1.3 Proof of Theorem 36.

Recall that the *length* of a monomial is the number of distinct variables that occur in it (so for example $x_1^2 x_2^4$ has length two). Recall that an *affine* function is simply a degree-1 polynomial.

Let $f : \mathbb{F}^n \to \mathbb{F}$ be any function. We say that the *influence* of variable x_i on f is

$$\mathrm{Inf}_i(f) \stackrel{\text{def}}{=} \Pr_{x_1,\ldots,x_n,y \in \mathbb{F}}[f(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n) \neq f(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_n)].$$

If f is a single monomial of length ℓ that contains the variable x_1, then the influence of x_1 on f is $(1 - \frac{1}{P})^\ell$ (the probability that the other $\ell - 1$ variables besides x_1 all take nonzero values is $(1 - \frac{1}{P})^{\ell-1}$, and then there is a $1 - \frac{1}{P}$ probability that the value of x_1 changes when we re-randomize). Similarly, if g is an

s-sparse polynomial in which x_1 occurs in r monomials of length $\ell_1, \dots, \ell_r$, then the influence of x_1 is at most

$$\left(1 - \frac{1}{P}\right)^{\ell_1} + \cdots + \left(1 - \frac{1}{P}\right)^{\ell_r}.$$

The *total influence* of f is the sum of the influences of all variables. Each monomial of length ℓ in a polynomial g contributes at most $\ell(1 - \frac{1}{P})^{\ell}$ to the total influence of f (i.e. if a polynomial has k monomials of lengths $\ell_1, \dots, \ell_k$ then the total influence of g is at most $\ell_1(1 - \frac{1}{P})^{\ell_1} + \cdots + \ell_k(1 - \frac{1}{P})^{\ell_k}$.

Note that each variable in an affine function of the form (7) has influence $1 - \frac{1}{P}$, and the total influence of such a function is precisely $(s + r)(1 - \frac{1}{P})$.

The following fact will be useful:

Fact 46. *Let $f, g : \mathbb{F}^n \to \mathbb{F}$ be two functions such that for some variable x_i we have $|\mathrm{Inf}_i(f) - \mathrm{Inf}_i(g)| = \tau$. Then f is $\tau/2$-far from g.*

Proof. We may assume without loss of generality that $\mathrm{Inf}_i(g) = \mathrm{Inf}_i(f) + \tau$. Let x denote a uniform random input from $\mathbb{F}$ and let x' denote x with the i-th coordinate re-randomized. We have

$$\Pr_{x,x'}[g(x) \neq g(x')] \quad \leq \quad \Pr_{x,x'}[g(x) \neq f(x)] + \Pr_{x,x'}[f(x) \neq f(x')] + \Pr_{x,x'}[f(x') \neq g(x')].$$

Rearranging, we get

$$\begin{aligned} \tau &= \Pr_{x,x'}[g(x) \neq g(x')] - \Pr_{x,x'}[f(x) \neq f(x')] \\ &\leq \Pr_{x,x'}[g(x) \neq f(x)] + \Pr_{x,x'}[f(x') \neq g(x')] = 2 \Pr_{x,x'}[g(x) \neq f(x)] \end{aligned}$$

where the final inequality holds since both x and x' are uniformly distributed. This gives the fact. ■

Finally, recall that in any polynomial $g(x_1, \dots, x_n)$ over $\mathbb{F}$, we may assume without loss of generality that no variable's degree in any monomial is greater than $P - 1$. (The multiplicative group is of size $P - 1$ and hence $\alpha^P = \alpha$ for every $\alpha \in \mathbb{F}$.)

Proof of Theorem 36.

Let g be an s-sparse polynomial in $\mathbb{F}[x_1, \dots, x_n]$ and let $A(x)$ be a fixed affine function given by equation (7). We will show that g must be $\Phi(P)$-far from A and thus prove the theorem.

First note that without loss of generality we may assume g has no term of degree 1. (Suppose g has t such terms. Let g' be the polynomial obtained by subtracting off these terms. Then g' is $(s - t)$-sparse and is $\Phi(P)$-close to the affine function $A'(x)$ obtained by subtracting off the same terms; this affine function has at least $s + r - t$ nonconstant terms. So we can run the following argument on g' with $s - t$ playing the role of "s" in the lemma.)

Now we observe that g must satisfy

$$\mathrm{Inf}_1(g) + \cdots + \mathrm{Inf}_s(g) \geq (1 - 4\Phi(P))s(1 - \frac{1}{P}). \tag{17}$$

If this were not the case, then some variable x_i in $x_1, \dots, x_s$ would necessarily have influence at most $(1 - 4\Phi(P))(1 - \frac{1}{P})$ on g. Since the influence of x_i on (7) is $1 - \frac{1}{P}$, by Fact 46 this would mean that g is at least $2\Phi(P)(1 - \frac{1}{P}) \geq \Phi(P)$-far from (7), and we would be done.

Notation. We will henceforth refer to monomials in g of length less than P^2 as *short* monomials, and we write S to denote the set of all short monomials in g. For $P^2 \leq \ell \leq P^8$, we refer to monomials in g of length ℓ as *intermediate* monomials, and we write I to denote the set of all intermediate monomials in g. Finally, for $\ell > P^8$ we refer to monomials in g of length ℓ as *long* monomials, and we write L to denote the set of all long monomials.

Observe that

- Each monomial in g that is intermediate or long contributes at most $1/4$ to $\mathrm{Inf}_1(g) + \cdots + \mathrm{Inf}_s(g)$. This is because each monomial of length $\ell \geq P^2$ contributes at most $\ell(1-\frac{1}{P})^\ell$ to this sum, and for integer ℓ the value $\max_{\ell \geq P^2} \ell(1-\frac{1}{P})^\ell$ is achieved at $\ell = P^2$ where the value is at most $1/4$ (the upper bound holds for all integer $P \geq 2$).

- Each short monomial in g contributes at most P/e to $\mathrm{Inf}_1(g) + \cdots + \mathrm{Inf}_s(g)$. This is because $\max_{\ell \geq 1} \ell(1-\frac{1}{P})^\ell \leq P/e$ (the max is achieved around $\ell \approx P$).

Since the RHS of (17) is at least $(1 - \frac{1.2}{P})s$, we have the following inequalities:

$$\frac{|I|+|L|}{4} + \frac{|S|P}{e} \geq \left(1 - \frac{1.2}{P}\right)s \qquad \text{and} \qquad |I|+|L| \leq s$$

(the second inequality holds simply because there are at most s long monomials). These inequalities straightforwardly yield $|S| \geq \frac{s}{3P}$.

Let m_ℓ denote the number of monomials in g that have length exactly ℓ. Note that we have $\sum_{\ell > P^8} m_\ell = |L| \leq s$.

Given two monomials M_1, M_2 that occur in g, we say that M_1 *covers* M_2 if all variables in M_1 are also in M_2 (note we do not care about the degrees of the variables in these monomials). We refer to such a pair (M_1, M_2) as a *coverage*; more precisely, if M_1 is of length ℓ we refer to the pair (M_1, M_2) as an *ℓ-coverage*. (One can view each ℓ-coverage as an edge in a bipartite graph.)

Let $S' \subseteq S$ be the set of those monomials M in S which are "maximal" in the sense that no other monomial $M' \in S$ (with $M' \neq M$) covers M.

Claim 47. *We have* $|S'| \geq s/(3P^{P^2})$.

Proof. Since S is finite it is clear that S' is nonempty; suppose the elements of S' are $M_1, \ldots, M_k$. Each of the (at least $s/(3P)$ many) elements of S is covered by some M_i. But each M_i is of length ℓ for some $\ell \leq P^2 - 1$, and hence can cover at most P^ℓ monomials (any monomial covered by M_i is specified by giving ℓ exponents, each between 0 and $P-1$, for the ℓ variables in M_i). ■

Fix any $\ell \geq P^2$. Each fixed monomial of length ℓ participates in at most $\binom{\ell}{P^2} P^{P^2} \leq (\ell P)^{P^2}$ many ℓ-coverages of monomials in S'. (There are $\binom{\ell}{P^2}$ ways to choose a subset of P^2 variables, and once chosen, each variable may take any exponent between 0 and $P-1$.) Consequently, the length-ℓ monomials in g collectively participate in at most $m_\ell(\ell P)^{P^2}$ many ℓ-coverages of variables in S' in total. By Claim 47, it follows that

$$\mathbb{E}_{M \in S'}[\#\ \ell\text{-coverages } M \text{ is in}] \leq \frac{m_\ell(\ell P)^{P^2}}{s/(3P^{P^2})} = \frac{3m_\ell \ell^{P^2} P^{2P^2}}{s}.$$

By Markov's inequality, we have

$$\Pr_{M \in S'}[\#\ \ell\text{-coverages } M \text{ is in} \geq 3m_\ell \ell^{P^2+2} P^{2P^2}/s] \leq 1/\ell^2.$$

So for each $\ell \geq P^2$, we have that at most a $1/\ell^2$ fraction of monomials in S' are covered by at least $3m_\ell \ell^{P^2+2} P^{2P^2}/s$ many length-ℓ monomials. Since $\sum_{\ell \geq P^2} 1/\ell^2 < 1/2$, we have that at least half of the monomials in S' have the following property:

- For all $\ell \geq P^2$, at most $3m_\ell \ell^{P^2+2} P^{2P^2}/s$ many length-ℓ monomials cover M. $\quad (\dagger)$

Fix M to be some particular monomial with property $(\dagger)$. Since M belongs to S', we know that no short monomial in g covers M; we now show that for a constant fraction of all restrictions ρ of variables outside of M, no intermediate or long monomial in g_ρ covers M. (Once this is accomplished, we will be almost done.)

First observe that for any value ℓ with $P^2 \leq \ell \leq P^8$, using the fact that m_ℓ/s is at most 1, we have that at most

$$3\ell^{P^2+2} P^{2P^2} \leq 3P^{10P^2+16} \leq P^{10P^2+18}$$

many length-ℓ monomials cover M. So in total there are at most $(P^8 - P^2 + 1)P^{10P^2+18} \leq P^{10P^2+26}$ many intermediate monomials that cover M; let T denote the set of these intermediate monomials. Each intermediate monomial in T has length strictly greater than the length of M, so each such monomial contains at least one variable that is not in M. Let V be a set of at most P^{10P^2+26} variables such that each monomial in T contains at least one variable from V, and let ρ_1 be the restriction that sets all variables in V to 0 and leaves all other variables unfixed. Note that for each long monomial in g, applying ρ_1 either kills the monomial (because some variable is set to 0) or leaves it unchanged (no variable in the monomial is set) in g_{ρ_1}. Thus the result of applying ρ_1 is that no intermediate monomial in g_{ρ_1} covers M.

Now let ρ_2 denote a random restriction over the remaining variables which leaves free precisely those variables that occur in M and fixes all other variables independently to uniformly chosen elements of $\mathbb{F}$. Suppose M' is a long monomial (of length $\ell > P^8$) from g that survived into g_{ρ_1}. It must be the case that M' contains at least $\ell - P^2$ variables that are neither in M nor in V, and consequently the probability that M' is not killed by ρ_2 (i.e. the probability that all variables in M' that are not in M are set to nonzero values under ρ_2) is at most $(1-\frac{1}{P})^{\ell-P^2}$. Consequently the expected number of length-ℓ monomials in g_{ρ_1} that cover M and are not killed by ρ_2 is at most $3m_\ell \ell^{P^2} P^{2P^2}(1-\frac{1}{P})^{\ell-P^2}/s$. Summing over all $\ell > P^8$, we have

$$\begin{aligned}
&\mathbb{E}_{\rho_2}[\text{\# long monomials that cover } M \text{ and survive } \rho_1\rho_2] &(18)\\
&\leq \sum_{\ell > P^8} \frac{3m_\ell \ell^{P^2} P^{2P^2}(1-\frac{1}{P})^{\ell-P^2}}{s}\\
&\leq \left(\sum_{\ell > P^8} \frac{m_\ell}{s}\right) \cdot \max_{\ell \geq P^8} 3\ell^{P^2} P^{2P^2}(1-\frac{1}{P})^{\ell-P^2}. &(19)
\end{aligned}$$

We have $\sum_{\ell > P^8} \frac{m_\ell}{s} \leq 1$. A routine exercise shows that for all $P \geq 2$, the max in (19) is achieved at $\ell = P^8$ where the value is at most $1/2$ (in fact it is far smaller). So (18) is certainly at most $1/2$, and we have

$$\mathbb{E}_{\rho_2}[\text{\# long monomials that cover } M \text{ and survive } \rho_1\rho_2] \leq 1/2.$$

So the probability that any long monomial that covers M survives $\rho_1\rho_2$ is at most $1/2$. Since we already showed that no short or intermediate monomial in $g_{\rho_1\rho_2}$ covers M, it follows that with probability at least $1/2$ over the random choice of ρ_2, no monomial in $g_{\rho_1\rho_2}$ covers M except for M itself.

Now let ρ denote a truly random restriction that assigns all variables not in M uniformly at random and keeps all variables in M free. Since the variables in V will be assigned according to ρ_2 with probability $1/P^{P^{10P^2+26}}$, we have that with probability at least $1/(2P^{P^{10P^2+26}}) > 1/(P^{P^{10P^2+26}+1})$ over the random choice of ρ, no monomial in g_ρ covers M. Suppose ρ is such a restriction. Since M itself clearly survives the restriction ρ, we have that the function g_ρ (a function on length$(M) \leq P^2 - 1$ many variables) is different from the function A_ρ – this is simply because the polynomial g_ρ contains the monomial M, which is not of degree 1, whereas all monomials in A_ρ have degree 1. Hence the functions g_ρ and A_ρ differ on at least one of the (at most) P^{P^2-1} possible inputs.

So, we have shown that for at least a $1/(P^{P^{10P^2+26}+1})$ fraction of all restrictions of the variables not occurring in M, the error of g under the restriction in computing A is at least $1/P^{P^2-1}$. This implies that the overall error of g in computing A is at least

$$1/(P^{P^{10P^2+26}+P^2}) = \Phi(P)$$

and we are done with the proof of Theorem 36. ∎

E.2. Lower Bounds for Boolean Function Classes

In this section we prove lower bounds on the query complexity of testing size-s decision trees, size-s branching programs, s-term DNF, and size-s Boolean formulas (Theorem 48), and Boolean functions with Fourier degree at most d (Theorem 51).

Theorem 48. *Let $\epsilon = 1/1000$. Any ϵ-testing algorithm for any of the following classes of functions over $\{0,1\}^n$ must make $\Omega(\log s/\log\log s)$ queries: (i) size-s decision trees; (ii) size-s branching programs; (iii) s-term DNF; (iv) size-s Boolean formulas.*

Proof. The proof combines a counting argument with the result of Chockler and Gutfreund [3] showing that $\Omega(J/k)$ queries are required to distinguish between J-juntas and $(J+k)$-juntas over $\{0,1\}^n$. More precisely, consider the following distributions:

1. D_{NO} is the uniform distribution over all functions (on n variables) that depend on (at most) the first $(J+k)$ variables.

2. D_{YES} is the distribution obtained in the following way. Choose a k-element subset $\mathcal{I}_k$ uniformly and randomly from the set $\{1,\ldots,J+k\}$. Then choose a uniformly random function from the set of all functions on n variables that depend on (at most) the variables indexed by the set $[J+k]\setminus\mathcal{I}_k$.

Chockler and Gutfreund show that with very high probability a random draw from D_{NO} is far from every J-junta, whereas clearly every draw from D_{YES} is a J-junta. Given any putative testing algorithm, the distributions $D_{\mathrm{YES}}, D_{\mathrm{NO}}$ over functions induce two distributions C_{YES}, C_{NO} over "query-answer histories". Chockler and Gutfreund show that for any (even adaptive) algorithm that makes fewer than $\Omega(J/k)$ queries, the statistical difference between C_{YES} and C_{NO} will be at most $1/6$. This implies that any successful testing algorithm must make $\Omega(J/k)$ queries.

We adapt this argument to prove Theorem 48 as follows. Let $\mathcal{C}_s$ be a class of functions for which we would like to prove a lower bound (e.g. $\mathcal{C}_s$ could be the class of all Boolean functions over n variables that are computed by decision trees of size at most s). We choose J (as a function of s) such that any J-junta is a function in $\mathcal{C}_s$; with this choice the distribution D_{YES} described above is indeed a distribution over functions in the class. We choose k (as a function of J) so that with very high probability, a random function drawn from D_{NO} (i.e. a random function over the first $J+k$ variables) is ϵ-far from every function in $\mathcal{C}_s$. This gives an $\Omega(J/k)$ lower bound for testing whether a black-box function is in $\mathcal{C}_s$ or is ϵ-far from every function in $\mathcal{C}_s$.

For all of the classes addressed in Proposition 48 we can take $J = \log_2 s$ and $k = \Theta(\log J)$. We work through the analysis for size-s decision trees, sketch the analysis for size-s branching programs, and leave the (very similar) analysis for s-term DNF and size-s Boolean formulas to the interested reader.

Decision Trees (of size s): We set $J = \log_2 s$ and $k = \log_2 J$. It is clear that any J-junta can be expressed as a size-s decision tree.

Lemma 49. *Fix $\epsilon = 1/1000$. With very high probability, a random $(J + \log J)$-junta over the first $(J + \log J)$ variables is ϵ-far from any size-s decision tree over the first $(J + \log J)$ variables.*

Proof. For any size-s decision tree over the first $(J + \log J)$ variables, the number of $(J + \log J)$-juntas (over these variables) ϵ-close to it equals $\sum_{i=0}^{\epsilon \cdot 2^{J+\log J}} \binom{2^{J+\log J}}{i}$. For $\epsilon = 1/1000$, this is at most $2^{0.1 \cdot 2^{J+\log J}} = 2^{(J/10)2^J}$ (recall that the sum of the binomial coefficients $\sum_{k=0}^{n/\alpha} \binom{n}{k}$ is $O(C(\alpha)^n)$, where $C(\alpha) = \alpha^{1/\alpha}(\frac{\alpha}{\alpha-1})^{\frac{\alpha-1}{\alpha}}$.)

Now we upper bound the number of size-s decision trees over the first $J + \log J$ variables. There are at most $4^s = 2^{2 \cdot 2^J})$ distinct decision tree topologies for trees with s leaves. For each topology there are at most $(J + \log J)^s \leq 2^{2s \log \log s} = 2^{(2 \log J) 2^J}$ different labellings of the nodes.

Thus, the number of $(J + \log J)$-juntas that are ϵ-close to *any* decision tree of size s (over the first $J + \log J$ variables) is at most $2^{(J/10 + 2 \log J)2^J}$. This is a vanishingly small fraction of the total number of $(J + \log J)$-juntas over the first $(J + \log J)$ variables, which is $2^{2^{J+\log J}} = 2^{J \cdot 2^J}$. ∎

We are not quite done, since we need that with very high probability a random function from D_{NO} is ϵ-far from *every* size-s decision tree, not just from size-s decision trees over the first $(J + \log J)$ variables. This follows easily from the previous lemma:

Corollary 50. *For $\epsilon = 1/1000$, with very high probability a random $(J + \log J)$-junta over the first $(J + \log J)$ variables is ϵ-far from any size-s decision tree (over n variables).*

Proof. Let f be any $(J + \log J)$-junta over the set $\{x_1, \ldots, x_{J+\log J}\}$. Suppose that g is a size-s decision tree over $\{x_1, \ldots, x_n\}$ that is ϵ-close to f. It is not hard to show that then there exists a size-s decision tree g' over the relevant variables $\{x_1, \ldots, x_{J+\log J}\}$ that is ϵ-close to f as well (g' can be obtained from g by fixing all the irrelevant variables to the values that maximize g's agreement with f). ∎

We have thus established part (i) of Theorem 48.

Branching Programs: We only sketch the required analysis. We set $J = \log_2 s$ and $k = 10 \log_2 J$. Any J-junta can be expressed as a size-s branching program. Simple counting arguments show that for $\epsilon = 1/1000$, a random $(J + k)$-junta over $\{x_1, \ldots, x_{J+k}\}$ is with high probability ϵ-far from every size-s Branching Program over $\{x_1, \ldots, x_{J+k}\}$. An analogue of Corollary 50 completes the argument.

This completes the proof of Theorem 48. ∎

Remark: We note that these simple arguments do not seem to give any non-trivial testing lower bound for the class of Boolean circuits of size s. It would be interesting to obtain lower bounds for this class.

Finally, we point out the following:

Theorem 51. *Let $0 < \epsilon < 1/2$. Any non-adaptive ϵ-testing algorithm for the class of Boolean functions over $\{0, 1\}^n$ with Fourier degree d must make $\tilde{\Omega}(\sqrt{d})$ queries.*

Proof. Consider the following two distributions over Boolean functions on $\{-1, 1\}^n$:

1. D_{NO} is the uniform distribution over all $\binom{n}{d+2}$ parities of exactly $d + 2$ variables from $x_1, \ldots, x_n$;

2. D_{YES} is the uniform distribution over all $\binom{n}{d}$ parities of exactly d variables from $x_1, \ldots, x_n$.

Every function in the D_{YES} distribution clearly has Fourier degree, whereas every function in the D_{NO} distribution has distance precisely $1/2$ from every function with Fourier degree d (this follows immediately from Parseval's identity). Fischer *et al.* showed that any non-adaptive algorithm for distinguishing draws from D_{YES} versus D_{NO} must make $\tilde{\Omega}(\sqrt{d})$ draws; this immediately gives the desired result. ∎