Property Testing Lower Bounds Via Communication Complexity

Eric Blais
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA, USA 15213
eblais@cs.cmu.edu

Joshua Brody
Computer Science Department
Aarhus University
Aarhus, Denmark
joshua.e.brody@gmail.com

Kevin Matulef
IIIS, Tsinghua University
matulef@gmail.com

June 29, 2012

Abstract

We develop a new technique for proving lower bounds in property testing, by showing a strong connection between testing and communication complexity. We give a simple scheme for reducing communication problems to testing problems, thus allowing us to use known lower bounds in communication complexity to prove lower bounds in testing. This scheme is general and implies a number of new testing bounds, as well as simpler proofs of several known bounds.

For the problem of testing whether a boolean function is k-linear (a parity function on k variables), we achieve a lower bound of $\Omega(k)$ queries, even for adaptive algorithms with two-sided error, thus confirming a conjecture of Goldreich (2010b). The same argument behind this lower bound also implies a new proof of known lower bounds for testing related classes such as k-juntas. For some classes, such as the class of monotone functions and the class of s-sparse GF(2) polynomials, we significantly strengthen the best known bounds.

1 Introduction

The field of property testing seeks to formalize the question: what can we determine about a large object, with limited access to the object itself? In general the large object may be anything—for instance a graph on n nodes, or a function on n variables. In a typical property testing setup, a tester who has unbounded computational power is given query access to the large object. The tester's goal is to accept the object if it has some property $\mathcal{P}$, and reject it if it is "far" from having property $\mathcal{P}$.

In this paper we will primarily concern ourselves with the case when the large object is a boolean function f on n bits. In this case, the tester's goal is to accept f with probability at least $2/3$ if f has property $\mathcal{P}$, and reject with probability at least $2/3$ if f must be modified on an ϵ fraction of the 2^n possible inputs in order to have property $\mathcal{P}$. The query complexity (i.e. the number of times the testing algorithm must query f) should hopefully be a small function of ϵ and n.

The notion of testing boolean functions in this framework goes back to the seminal work of Rubinfeld and Sudan (1996) and has several connections to complexity theory (in particular PCPs and hardness of approximation), as well as computational learning theory (Ron, 2008). Over the last two decades, researchers have exerted a considerable amount of effort to determine the query complexity for testing properties of a function f, such as whether f is a linear function (Blum et al., 1993), whether f is isomorphic to a given function (Blais and O'Donnell, 2010; Chakraborty et al., 2011a; Alon and Blais, 2010), whether f is a k-junta (Fischer et al., 2004; Blais, 2008, 2009), a monotone function (Goldreich et al., 2000; Fischer et al., 2002),

a dictator (Parnas et al., 2002), a halfspace (Matulef et al., 2009), an s-sparse polynomial, a size-s decision tree, etc. (Diakonikolas et al., 2007). Starting with the ground-breaking work of Goldreich et al. (1998), there has also been much effort directed at determining the query complexity for testing properties of graphs and, more generally, of combinatorial objects. (See, e.g., Ron 2009; Goldreich 2010a.)

Over the course of this effort, a variety of techniques have been developed for designing property testing algorithms, thus proving testing upper bounds. However, as is often the case in theoretical computer science, lower bounds are harder to come by. Although several lower bounds for specific problems are known, few general techniques are known beyond the use of Yao's minimax lemma.

Communication complexity is an area which has collected many effective techniques for proving lower bounds in other areas of computer science. In a typical setup, two parties, Alice and Bob, each have an input and they would like to decide something about their joint input. Their computational power is unbounded, but they would like to compute the answer with as little *communication* as possible.

The communication complexity framework has been extensively studied; in particular, several problems are known to require a large amount of communication. These include SET-DISJOINTNESS, INDEX, INNER-PRODUCT, and GAP-HAMMING-DISTANCE. The hardness of these and related problems has been used to obtain lower bounds in many areas such as streaming algorithms, circuit complexity, data structures, and proof complexity (Kushilevitz and Nisan 1997; Indyk and Woodruff 2003; Miltersen et al. 1995).

Property testing and communication complexity have striking similarities. Both involve parties with unbounded computational power (in one case, the tester, and in the other case, the communicating players), and both involve algorithms which are restricted by the parties' limited access to their input. Despite these similarities, no previous connection between these fields has been made.

In this work we show that there is indeed a strong connection between testing and communication complexity. More specifically, we show how to reduce certain communication problems to property testing problems. This reduction method represents a new approach to proving testing lower bounds. This approach turns out to be quite fruitful, both for proving new bounds and for giving simpler proofs of known bounds in property testing.

1.1 Our Results

Testing k-linear functions. The function $f : \{0,1\}^n \to \{0,1\}$ is *linear*, i.e. a parity function, when there exists a set $S = \{i_1, \ldots, i_s\} \subseteq [n]$ such that for every $x \in \{0,1\}^n$, $f(x) = x_{i_1} \oplus \cdots \oplus x_{i_s}$. When $|S| = k$, we say that f is a *k-linear* function. The problem of testing k-linear functions was first studied by Fischer et al. (2004). The best lower bound is due to Goldreich (2010b), who showed that $\Omega(\sqrt{k})$ queries are required to test k-linear functions. He also showed that non-adaptive testers require $\Omega(k)$ queries to test the same property, and conjectured that this stronger lower bound holds for all testers (adaptive or not).[1]

We confirm Goldreich's conjecture. Using the same construction, we also obtain lower bounds on the query complexity for testing juntas,[2] testing functions of low Fourier degree, and testing sparse polynomials.

Theorem 1.1. *Fix $1 < k < n - 1$. At least $\Omega(\min\{k, n-k\})$ queries are required to test*

(i) k-linear functions,

(ii) k-juntas,

(iii) functions of Fourier degree at most k, and

(iv) functions with k-sparse polynomial representation in GF(2).

Remark. In parallel work, Blais and Kane (2012) simultaneously obtained a different proof of Goldreich's conjecture using Fourier-analytic methods.

[1] The conjecture and results of Goldreich (2010b) are stated in terms of testing $\leq k$-linear functions (the class of functions that are parities on at most k bits) but the proofs of Goldreich (2010b) give identical lower bounds for testing k-linearity. Similarly, the technique in our lower bounds for testing k-linearity gives identical bounds for testing $\leq k$-linearity.

[2] Informally, a function is a *k-junta* if it has at most k relevant variables. For a complete definition of the properties stated in Theorem 1.1, see Section 3. Similarly, the properties introduced in the rest of the introduction are defined formally in the sections containing the proofs of the corresponding theorems.

Class of functions	**Our bound**	**Previous lower bounds**	**Upper bounds**
k-linear	$\boldsymbol{\Omega(k)}$	$\Omega(\sqrt{k})$ [a] $\Omega(k)$ (n.a.) [a]	$O(k \log k)$ [b] $O(n)$ [c]
k-juntas	$\Omega(k)$	$\Omega(k)$ [d]	$O(k \log k)$ [e]
Fourier degree $\leq d$	$\Omega(d)$	$\Omega(d)$ [b]	$2^{O(d)}$ [b,f]
s-sparse GF(2)-polynomials	$\boldsymbol{\Omega(s)}$	$\Omega(\sqrt{s})$ [g]	$\tilde{O}(s)$ [g]
monotone $f : \{0,1\}^n \to R$	$\boldsymbol{\Omega(\min\{n, \lvert R\rvert^2\})}$	$\Omega(\log n)$ (n.a.) [h] $\Omega(n)$ (n.a., 1-s.) [i]	$O(n \log \lvert R\rvert)$ [j]
submodular $f : \{0,1\}^n \to \mathbb{R}$	$\boldsymbol{\Omega(n)}$	$\Omega(\log n)$ (n.a.) [h,k] $\Omega(n)$ (n.a., 1-s.) [i,k]	$2^{O(\sqrt{n} \log n)}$ [k]
size-s branching programs, size-s boolean formulas	$\Omega(\log s)$	$s^{\Omega(1)}$ [g]	$\tilde{O}(s)$ [g]
s-term DNF formulas	$\Omega(\log s)$	$\Omega(\log s)$ [g]	$\tilde{O}(s)$ [g]
size-s decision trees	$\boldsymbol{\Omega(s)}$ **(1-s.)**	$\Omega(\log s)$ [g]	$\tilde{O}(s)$ [g]
signed k-majority	$\boldsymbol{\Omega(k/\log k)}$ **(1-s.)** for $k \leq \gamma n$, $\gamma \in (0,1)$	$\Omega(k^{1/12})$(n.a.) [l,m] for $k \leq \frac{3}{4}n$	$O(\sqrt{n})$ [m] for $k = n$

Table 1: Summary of our results. Bold font indicates an improvement over the previous bounds. Bounds labeled with (n.a.) apply only to non-adaptive testers; bounds marked with (1-s.) only apply to testers with one-sided error. All other bounds apply to adaptive testers with two-sided error.

[a] Goldreich (2010b).
[b] Chakraborty, Soriano, and Matsliah (2011a).
[c] Folklore.
[d] Chockler and Gutfreund (2004).
[e] Blais (2009).
[f] Diakonikolas, Lee, Matulef, Onak, Rubinfeld, Servedio, and Wan (2007).
[g] Chakraborty, Soriano, and Matsliah (2011b).
[h] Fischer, Lehman, Newman, Raskhodnikova, Rubinfeld, and Samorodnitsky (2002).
[i] Briët, Chakraborty, Soriano, and Matsliah (2010).
[j] Dodis, Goldreich, Lehman, Raskhodnikova, Ron, and Samorodnitsky (1999).
[k] Seshadhri and Vondrák (2011).
[l] Blais and O'Donnell (2010). The lower bound stated in the table is not found explicitly in (Blais and O'Donnell, 2010), but it can be obtained using the arguments from that paper. See Section 6 for more details.
[m] Matulef, O'Donnell, Rubinfeld, and Servedio (2009).

Theorem 1.1 has implications for the problem of *isomorphism testing*, or testing whether an unknown function f is equivalent, up to permutation of variables, to a fixed function $g : \{0,1\}^n \to \{0,1\}$. Alon and Blais (2010) showed that for most functions g, testing g-isomorphism non-adaptively requires $\Omega(n)$ queries. Similarly, Chakraborty et al. (2011a) showed that for every $k \leq n$, there exists a k-junta g such that testing g-isomorphism requires $\Omega(k)$ queries. Both of these results are non-constructive, and they raise the question of whether we can identify an *explicit* function for which the same lower bounds apply. Theorem 1.1 shows that for every $k \leq \frac{n}{2}$, the lower bound applies to the function $g : x \mapsto x_1 \oplus \cdots \oplus x_k$ since testing isomorphism to this function is equivalent to testing k-linearity.

Testing and OBDDs. Fix some finite sets X and Y. An *ordered binary decision diagram* (OBDD) is a directed acyclic graph with a single root and at most $n+1$ levels of nodes. The sink nodes in the last level are each associated with some element from Y. All other levels are associated with an index from $\{1, \ldots, n\}$. The nodes in these levels have out-degree $|X|$, with one edge associated with each element of X. For $x = (x_1, \ldots, x_n) \in X^n$, define $f(x)$ to be the value of the sink node that we reach when we start at the source, then at each level we follow the edge associated with x_i, where i is the index associated with the current level. The resulting function $f : X^n \to Y$ is the function *computed* by the OBDD. The *width* of an OBDD is the maximum number of nodes at any level.

Ron and Tsur (2009) first studied the problems of determining the query complexity for testing whether a function is computable by small-width OBDDs. Goldreich (2010b) continued this line of research and also asked whether we can establish stronger lower bounds for the query complexity of testing properties that include a subset of the functions computable by small-width OBDDs. In particular, he showed that there is a property consisting of functions computable by width-2 OBDDs that requires $\Theta(n)$ queries to test. Theorem 1.1 gives a different proof of the same result, since $\frac{n}{2}$-linear functions are computable by width-2 OBDDs.

Goldreich (2010b) conjectured in that an identical lower bound also held for testing an explicit subclass of functions computable by width-3 OBDDs and for testing functions computable by width-4 OBDDs. Recently, he observed (private communication) that the method of the proof of Theorem 1.1 can also be used to prove these conjectures:

Theorem 1.2. *Testing the class of linear functions from* $\mathrm{GF}(3)^n$ *to* $\mathrm{GF}(3)$ *that have only* 0-1 *coefficients requires* $\Theta(n)$ *queries.*

Note that this class of functions can be trivially computed by a width-3 OBDD by maintaining a partial sum of the inputs.

Theorem 1.3. *Testing the class of functions that are computable by width-4 OBDDs requires* $\Theta(n)$ *queries.*

Remark. Theorem 1.3 was proved independently by Brody et al. (2011).

Testing monotonicity. Fix $R \subseteq \mathbb{R}$. The function $f : \{0,1\}^n \to R$ is *monotone* if for any two inputs $x, y \in \{0,1\}^n$ where $x_1 \leq y_1, \ldots, x_n \leq y_n$, we have that $f(x) \leq f(y)$. The problem of testing monotonicity was first studied by Goldreich et al. (2000), who introduced a natural tester: sample random edges from the hypercube and verify that the function is monotone on those edges. This algorithm makes $O(n \log |R|)$ queries (Dodis et al., 1999). An important open problem in property testing is to determine whether there exist more efficient monotonicity testers.

Despite much attention to monotonicity testing (Batu et al., 1999; Ergun et al., 2000; Goldreich et al., 2000; Dodis et al., 1999; Fischer et al., 2002; Bhattacharyya et al., 2009; Briët et al., 2010), lower bounds for the query complexity of this problem have been elusive. Previously, the best bound for non-adaptive testers, due to Fischer et al. (2002), was only $\Omega(\log n)$. This translates to a $\Omega(\log \log n)$ lower bound for general (adaptive) testers.[3] We provide a significant improvement to this lower bound for functions with large ranges:

[3]Stronger bounds have been established for testers with one-sided error. See (Fischer et al., 2002; Briët et al., 2010) for details.

Theorem 1.4. *Testing $f : \{0,1\}^n \to R$ for monotonicity requires $\Omega(\min\{n, |R|^2\})$ queries.*

Notably, Theorem 1.4 gives the first progress on the natural-monotonicity-tester problem mentioned above: it shows that for $\sqrt{n} \leq |R| \leq \text{poly}(n)$, no monotonicity tester can improve on the query complexity of the natural tester by more than a logarithmic factor. We note, however, that this problem is still open in the important special case when $R = \{0,1\}$.

By a recent result of Seshadhri and Vondrák (2011), Theorem 1.4 also gives a new lower bound for the query complexity of testing submodularity.

Corollary 1.5. *Testing $f : \{0,1\}^n \to \mathbb{R}$ for submodularity requires $\Omega(n)$ queries.*

Testing concise representations. Parnas, Ron, and Samorodnitsky (2002) showed that testing whether a function can be represented by a monotone DNF with at most s terms can be done with a number of queries that depends only on s. This result was generalized by Diakonikolas et al. (2007), who introduced the method of *testing by implicit learning* and showed that this method can be used to test whether a function can be represented by a DNF with few terms, by a small decision tree, by a small branching program, etc. Our technique gives lower bounds on the query complexity for testing two of these properties.

Theorem 1.6. *At least $\Omega(\min\{s, n-s\})$ queries are required to test*

(i) size-2^s decision trees, and

(ii) size-s branching programs.

Testing juntas. The proof of Theorem 1.6 can also be extended to answer a question of Fischer et al. (2004): they asked if the query complexity of testing k-juntas can be reduced if the tester is only required to reject functions that are far from $(k+t-2)$-juntas for some $t > 0$. We show that the answer to this question is "no" for any $t \leq O(\sqrt{k})$.

Theorem 1.7. *Fix $k \leq \frac{n}{2}$ and $t > 0$. Any algorithm that accepts k-juntas and rejects functions $\frac{1}{4}$-far from $(k+t-2)$-juntas with high probability must make $\Omega\big(\min\{(\frac{k}{t})^2, k\}\big)$ queries.*

Remark. Subsequently, the result of Theorem 1.7 was essentially strengthened by Ron and Tsur (2011), who showed that for any constants ϵ and γ, a nearly linear lower bound of $\Omega(k/\log k)$ queries holds even if we are only required to accept k-juntas and reject functions that are ϵ-far from $(1+\gamma)k$-juntas.

Testers with one-sided error. The technique we introduce for proving new lower bounds can also be used to prove lower bounds for testers with one-sided error (that is, testers which accept functions with probability 1 if they have property $\mathcal{P}$, and reject them with probability at least 2/3 if they are far from having property $\mathcal{P}$). As a first application, we get a much stronger lower bound for the query complexity of testing decision trees with one-sided error.

Theorem 1.8. *At least $\Omega(s)$ queries are required to test size-s decision trees with one-sided error.*

We also obtain a lower bound on the query complexity of one-sided testers for a subclass of halfspaces, the class of "signed" majority functions on k variables.

Theorem 1.9. *Fix any constant $\gamma \in (0,1)$. For $k \leq \gamma n$, at least $\Omega(k/\log k)$ queries are required to test signed k-majorities with one-sided error.*

1.2 Techniques

The main idea behind all of our lower bounds is to show that the query complexity for testing a property $\mathcal{P}$ is bounded below by the randomized communication complexity of some well-studied communication game G. To do so, we introduce a $\mathcal{P}$-testing game. The definition of this game varies according to the situation, but typically looks like this: Alice receives a boolean function f, Bob receives a boolean function g, and they must test whether the joint function $h = f \oplus g$ has the property $\mathcal{P}$. We can then relate the number of queries required to test whether h has this property to the number of bits Alice and Bob need to communicate. Finally, we show that the $\mathcal{P}$-testing game requires large communication by using it to solve G.

This technique is best illustrated by example. In fact, we can give a very simple sketch of the proof of Theorem 1.1 (i), by showing how to reduce a version of the well-known SET-DISJOINTNESS problem to testing k-linearity. Suppose Alice and Bob each have sets of size k from a universe of size n. Suppose further that their sets are guaranteed to either intersect in exactly one place, or not at all, and they want to decide which is the case. We let k-DISJ denote this particular setting for SET-DISJOINTNESS. It is well-known that the communication complexity of k-DISJ is $\Omega(k)$ (Kalyanasundaram and Schnitger, 1992; Håstad and Wigderson, 2007).

One way Alice and Bob can solve k-DISJ is by forming linear functions based on their two sets. For a set $S \subseteq [n]$, define Parity_S to be the linear function on the bits indexed from S; i.e., $\text{Parity}_S(x) := \bigoplus_{i \in S} x_i$. Given input sets A and B, Alice forms the function $f = \text{Parity}_A$ and Bob forms the function $g = \text{Parity}_B$. The joint function $h = f \oplus g$ is $2k$-linear if the sets do not intersect, and $(2k-2)$-linear if they do. Note that every $2k$-linear function is $\frac{1}{2}$-far from being $(2k-2)$-linear (see Fact 3.4). Therefore, they can determine if their sets intersect by emulating a *testing* algorithm for $(2k-2)$-linearity on h. The testing algorithm requires oracle access to h, which neither Alice nor Bob have. However, they do know f and g, so Alice and Bob can simulate oracle access to h by exchanging $f(x)$ and $g(x)$, at a cost of two bits of communication per query. The total number of bits communicated is then twice the number of queries of the tester. Since we can lower bound the number of bits communicated by $\Omega(k)$, this implies that testing $(2k-2)$-linearity also requires $\Omega(k)$ queries. By scaling k, we achieve the first part of Theorem 1.1.

To summarize, our lower bound for testing k-linearity follows from three inequalities. Letting $\mathrm{C}_{\oplus}^{k\text{-LINEAR}}$ denote the communication game where Alice and Bob get f and g as input and wish to determine if $f \oplus g$ is k-linear or far from k-linear, and using $Q(\mathcal{P})$ to denote the query complexity of testing for $\mathcal{P}$ and $R(G)$ to denote the randomized communication complexity of a communication game G, we achieve a lower bound on testing k-linearity via the following chain of inequalities:

$$2Q(k\text{-LINEAR}) \geq R(\mathrm{C}_{\oplus}^{k\text{-LINEAR}}) \geq R(k\text{-DISJ}) = \Omega(k) \ . \tag{1}$$

All of the testing lower bounds in this paper follow the above structure. A crucial aspect of this proof technique is that emulating the property testing algorithm must be done in a *communication efficient* manner. In the example above, the joint function h was just the XOR of Alice's and Bob's functions, so simulating each query required only two bits of communication. For other lower bounds, we require more complicated ways to build a joint functions. However, as long as each query $h(x)$ can be simulated with low communication, a similar lower bound will hold. We formalize this statement in Lemma 2.4.

Note that in most situations, it is possible to use problems such as k-DISJ whose communication complexity is well understood, and therefore we get the equality in (1) essentially for free. The reduction in the first inequality is captured by Lemma 2.4, so for most proofs, the bulk of the actual work is in proving a so-called "distance lemma"—that YES instances for the communication problem map to instances where the combined function has property $\mathcal{P}$, and that NO instances will map to functions that are far from having $\mathcal{P}$. In most cases, as in Fact 3.4, these are simple to prove.

1.3 Organization

In Section 2, we introduce the communication complexity and property testing definitions, the Main Reduction Lemma (Lemma 2.4), and the communication complexity lower bounds that we use in the later sections.

The proofs of Theorems 1.1–1.3, as well as the formal definitions of the properties defined in those theorems, are presented in Section 3. The lower bound in Theorem 1.4 for testing monotonicity and the lower bound in Corollary 1.5 for testing submodularity are presented in Section 4. We complete the proof of Theorem 1.6 regarding the query complexity for testing functions computable by small decision trees or by small branching programs in Section 5. In the same section, we also complete the proof of Theorem 1.7. Finally, we present the lower bounds on the query complexity of one-sided testers for decision trees (Theorem 1.8) and for signed k-majority functions (Theorem 1.9) in Section 6.

2 From Communication Complexity to Property Testing

2.1 Property Testing Definitions

Recall that for a fixed range $R \subseteq \mathbb{R}$, a *property* of the functions $\{0,1\}^n \to R$ is a subset of those functions. The *distance* between two functions $f, g : \{0,1\}^n \to R$ is the fraction of inputs $x \in \{0,1\}^n$ for which $f(x) \neq g(x)$. The *distance* between f and a property $\mathcal{P}$ is the minimum distance between f and any function g in $\mathcal{P}$. When the distance from f to $\mathcal{P}$ is at least ϵ, we say that f is *ϵ-far* from $\mathcal{P}$.

Definition 2.1 (Tester)**.** An *(ϵ, q)-tester* for the property $\mathcal{P}$ of functions $\{0,1\}^n \to R$ is a randomized algorithm that queries a function $f : \{0,1\}^n \to R$ on at most q inputs from $\{0,1\}^n$ and

1. Accepts with probability at least $\frac{2}{3}$ when f is in $\mathcal{P}$; and
2. Rejects with probability at least $\frac{2}{3}$ when f is ϵ-far from $\mathcal{P}$.

A tester is said to be *non-adaptive* if it selects its q queries before observing the value of f on any of those queries; otherwise it is *adaptive*. A tester that always accepts functions in $\mathcal{P}$ has *one-sided error*; a tester that accepts functions in $\mathcal{P}$ with probability p for some $\frac{2}{3} \leq p < 1$ has *two-sided error*.

For any $0 < \epsilon < 1$, the *query complexity* of the property $\mathcal{P}$ at distance ϵ, denoted $Q_\epsilon(\mathcal{P})$, is the minimum value of q for which $\mathcal{P}$ has an (ϵ, q)-tester. Similarly, $Q^1_\epsilon(\mathcal{P})$ and $Q^{\mathrm{na}}_\epsilon(\mathcal{P})$ denote the minimum number of queries required to ϵ-test $\mathcal{P}$ with one-sided error and non-adaptive testers, respectively. Throughout this work, we will assume that ϵ is a small fixed constant—say, $\epsilon = 0.01$ for concreteness—and for simplicity we state all query complexity bounds only in terms of the other parameters and will omit ϵ from the notation.

2.2 Communication Complexity Definitions

In this subsection, we review the basic communication complexity setup and highlight some of the terms and concepts particularly relevant to this article. For more details, we refer the interested reader to the standard textbook by Kushilevitz and Nisan (1997).

In a typical communication game, there are two parties—Alice, who receives an input x, and Bob, who receives some input y. Alice and Bob wish to jointly compute some function $f(x, y)$ of their inputs. Neither player sees all the information needed to compute f, so they must communicate together to solve the problem. Communication complexity is the study of how much communication is necessary to compute f, for various functions f.

A *protocol* is a distributed algorithm that Alice and Bob use to compute $f(x, y)$; in particular, it specifies what messages Alice and Bob send to each other. In a *deterministic* protocol, Alice's messages are a function only of her input x and the previous communication in the protocol. Similarly, Bob's messages are a function of y and the previous communication. The *cost* of a protocol is the maximum (over all inputs) number of bits sent by Alice and Bob. The deterministic communication complexity of f, denoted $D(f)$, is the minimum cost of a deterministic protocol computing f.

In a *randomized* protocol, Alice and Bob have shared access to a (public coin) random string $r \in \{0,1\}^*$. We say that P is a δ-error protocol for f if for any input pair x, y, P computes $f(x, y)$ with probability at least $1 - \delta$, where the probability is taken over the random string r. We use $R_\delta(f)$ to denote the minimum cost of a δ-error protocol for f and define $R(f) := R_{1/3}(f)$. When f is a binary function, we say that a

protocol computes f with *one-sided error* if there exists $z \in \{0,1\}$ such that P computes f with certainty whenever $f(x,y) \neq z$, and with probability at least $1-\delta$ when $f(x,y) = z$. When considering randomized protocols with one-sided error, it is important to note which "side" the error guarantee is on. We use $R^z_\delta(f)$ to denote the minimum cost of a randomized protocol for f that correctly computes f whenever $f(x,y) \neq z$ and computes f with probability at least $1-\delta$ whenever $f(x,y) = z$. We define $R^z(f) := R^z_{1/3}(f)$.

A protocol is *one-way* if the communication consists of a single message from Alice to Bob, who then outputs an answer. We use $R^{\rightarrow}_\delta(f)$ to denote the minimum communication cost of a randomized, δ-error, one-way protocol for f. Finally, we use $R^{\rightarrow,z}_\delta(f)$ to denote the minimum communication cost of randomized one-way protocols for f with with one-sided error δ, and we define $R^{\rightarrow,z}(f) := R^{\rightarrow,z}_{1/3}(f)$.

2.3 The Main Reduction

Below we define a class of property testing communication games and show how communication lower bounds for these games yield query complexity lower bounds for property testers. Our communication games are based on what we call *combining operators*.

Definition 2.2 (Combining operator)**.** A *combining operator* is an operator ψ that takes as input two functions $f, g : \{0,1\}^n \to Z$ and returns a function $h : \{0,1\}^n \to R$.

We refer to the inputs f and g as the *base functions* of ψ. By convention, we use h to refer to the output of ψ. Given a combining operator ψ and a property $\mathcal{P}$, we define $\mathrm{C}^{\mathcal{P}}_\psi$ to be the following property testing communication game. Alice receives f. Bob receives a function g. They need to compute

$$\mathrm{C}^{\mathcal{P}}_\psi(f,g) := \begin{cases} 1 & \text{if } \psi(f,g) \in \mathcal{P} \\ 0 & \text{if } \psi(f,g) \text{ is } \epsilon\text{-far from } \mathcal{P}. \end{cases}$$

We prove all of our testing lower bounds by reducing from an associated communication game $\mathrm{C}^{\mathcal{P}}_\psi$. As mentioned in Section 1.2, this reduction is simple—Alice and Bob solve $\mathrm{C}^{\mathcal{P}}_\psi$ by emulating a $\mathcal{P}$-testing algorithm on $h := \psi(f,g)$. Note that neither Alice nor Bob have enough information to evaluate a query $h(x)$, because h depends on both f and g. Instead, they must communicate to jointly compute $h(x)$. For this reduction to give a strong query complexity lower bound for the property testing problem, it is essential that the joint computation of $h(x)$ occurs in a communication-efficient manner.

The following definition gives a sufficient condition on combining operators that yield strong reductions to testing problems.

Definition 2.3 (Simple combining operator)**.** A combining operator ψ is *simple* if for all f, g, and for all x, the query $h(x)$ can be computed given only x and the queries $f(x)$ and $g(x)$.

For example, when the base functions are boolean, the combining operator defined by $\psi(f,g) := f \oplus g$ is clearly simple—each $h(x) = f(x) \oplus g(x)$ can trivially be computed from $f(x)$ and $g(x)$. On the other hand, the combining operator ψ that returns the function defined by $h(x) := \bigoplus_{y \in T} [f(y) \cdot g(y)]$ is not simple when T is a large set of strings (say a Hamming ball centered at x), since computing $h(x)$ requires knowledge of $f(y)$ and $g(y)$ for several y.

All of the property testing communication games we use in this paper are based on simple combining operators and give us a tight connection between property testing and communication complexity via the following lemma.

Lemma 2.4 (Main Reduction Lemma)**.** *Fix Z to be a finite set. For any simple combining operator ψ with base functions $f, g : \{0,1\}^n \to Z$ and any property $\mathcal{P}$, we have*

(i) $R(\mathrm{C}^{\mathcal{P}}_\psi) \leq 2\, Q(\mathcal{P}) \cdot \lceil \log |Z| \rceil$,

(ii) $R^0(\mathrm{C}^{\mathcal{P}}_\psi) \leq 2\, Q^1(\mathcal{P}) \cdot \lceil \log |Z| \rceil$,

(iii) $R^{\rightarrow}(\mathrm{C}^{\mathcal{P}}_\psi) \leq Q^{\mathrm{na}}(\mathcal{P}) \cdot \lceil \log |Z| \rceil$, *and*

(iv) $R^{\rightarrow,0}(\mathrm{C}_\psi^{\mathcal{P}}) \leq Q^{\mathrm{na},1}(\mathcal{P}) \cdot \lceil \log |Z| \rceil$.

Proof. We begin by proving (iii). Let $\mathcal{A}$ be a q-query non-adaptive tester for $\mathcal{P}$. We create a one-way protocol P for $\mathrm{C}_\psi^{\mathcal{P}}$ in the following manner. Alice and Bob use public randomness to generate queries $x^{(1)}, \ldots, x^{(q)}$. Then, Alice computes $f(x^{(1)}), \ldots, f(x^{(q)})$ and sends them to Bob in a single $(q \cdot \lceil \log |Z| \rceil)$-bit message. For each i, Bob computes $g(x^{(i)})$ and combines it with $f(x^{(i)})$ to compute $h(x^{(i)})$. Finally, Bob emulates $\mathcal{A}$ using the responses $h(x^{(1)}), \ldots, h(x^{(q)})$ and outputs 1 if and only if $\mathcal{A}$ accepts h.

If $\mathcal{A}$ has two-sided error, then by the correctness of $\mathcal{A}$, P computes $\mathrm{C}_\psi^{\mathcal{P}}$ with probability at least 2/3. Hence, $R^{\rightarrow}(\mathrm{C}_\psi^{\mathcal{P}}) \leq q \cdot \lceil \log |Z| \rceil$. In particular, if $\mathcal{A}$ is an optimal non-adaptive tester with two-sided error, then $q = Q^{\mathrm{na}}(\mathcal{P})$, and part (iii) of the lemma is proved.

If $\mathcal{A}$ has one-sided error, then whenever $h \in \mathcal{P}$, the protocol P correctly outputs 1, and when h is ϵ-far from $\mathcal{P}$, the protocol correctly outputs 0 with probability at least 2/3. Therefore, $R^{\rightarrow,0}(\mathrm{C}_\psi^{\mathcal{P}}) \leq q \cdot \lceil \log |Z| \rceil$. In particular, when $\mathcal{A}$ is an optimal non-adaptive tester with one-sided error, $R^{\rightarrow,0}(\mathrm{C}_\psi^{\mathcal{P}}) \leq Q^1(\mathcal{P}) \cdot \lceil \log |Z| \rceil$.

Now, suppose $\mathcal{A}$ is a q-query adaptive tester for $\mathcal{P}$. Again, Alice and Bob use public randomness to generate queries $x^{(1)}, \ldots, x^{(q)}$. However, since $\mathcal{A}$ is adaptive, the distribution of the ith query $x^{(i)}$ depends on $h(x^{(j)})$ for all $j < i$. Instead of generating all queries in advance, Alice and Bob generate queries one at a time. Each time a query $x^{(i)}$ is generated, Alice and Bob exchange $f(x^{(i)})$ and $g(x^{(i)})$. Since ψ is a simple combining operator, this is enough information for Alice and Bob to individually compute $h(x^{(i)})$, which in turn gives them enough information to generate the next query with the appropriate distribution. When $h(x^{(1)}), \ldots, h(x^{(q)})$ have all been computed, Bob outputs 1 if and only if $\mathcal{A}$ accepts h. This protocol costs $2q \cdot \lceil \log |Z| \rceil$ bits of communication, and if $\mathcal{A}$ is an optimal adaptive tester, then $R(\mathrm{C}_\psi^{\mathcal{P}}) \leq 2\, Q(\mathcal{P}) \cdot \lceil \log |Z| \rceil$. Similarly, if $\mathcal{A}$ is an optimal adaptive tester with one-sided error, then $R^0(\mathrm{C}_\psi^{\mathcal{P}}) \leq 2\, Q^1(\mathcal{P}) \cdot \lceil \log |Z| \rceil$. □

2.4 Communication Complexity Problems

We achieve all of our testing lower bounds via Lemma 2.4. To prove lower bounds for $\mathrm{C}_\psi^{\mathcal{P}}$, we reduce from one of several standard communication complexity problems. However, we often require special flavors of these problems—either we need protocols with one-sided error, or we require the input to be restricted in some *balanced* way. We describe the variants that we will need for our reductions in this section.

Let $n \in \mathbb{N}, t := t(n)$, and $x, y \in \{0,1\}^n$. We use $\circ$ to denote string concatenation and 0^k (1^k) to denote the string of k consecutive zeros (ones). Let $x \oplus y$ denote the bitwise exclusive-or of x and y. We use $1 - x$ to denote the bitwise complement of x. The Hamming weight of a string x, denoted $|x|$, is the number of i such that $x_i = 1$. The Hamming distance between strings x and y, denoted $\Delta(x, y)$, is the number of coordinates i such that $x_i \neq y_i$. Note that $\Delta(x, y) = |x \oplus y|$.

We are interested in the following functions:

Set-Disjointness. Alice and Bob are given n-bit strings x and y respectively and wish to compute

$$\textsc{disj}_n(x, y) := \bigvee_{i=1}^{n} x_i \wedge y_i \,.$$

Equivalently, Alice and Bob's inputs can be viewed as sets $A, B \subseteq [n]$. In this case, $\textsc{disj}(A, B) = 1$ if and only if their sets intersect.

When n is clear from context, we drop the subscript. A celebrated result of Kalyanasundaram and Schnitger (1992), later simplified by Razborov (1990) and Bar-Yossef et al. (2002), showed that $R(\textsc{disj}_n) = \Omega(n)$, even under the promise that A and B intersect in at most one element.

Theorem 2.5 (Kalyanasundaram and Schnitger 1992)**.**

$$R(\textsc{disj}_n) = \Omega(n).$$

We use a balanced version of disjointness called k-BAL-DISJ. In this version, Alice receives a set $A \subseteq [n]$ of size $|A| = \lfloor k/2 \rfloor + 1$, Bob receives a set $B \subseteq [n]$ of size $\lceil k/2 \rceil + 1$, and there is a promise that $|A \cap B| \leq 1$.

Lemma 2.6. *For all $0 \le k \le n-2$, we have $R(k\text{-BAL-DISJ}) = \Omega(\min\{k, n-k\})$.*

Proof. If $n - k = O(1)$, there is nothing to prove. Otherwise, let $m := \min\{\lfloor k/2 \rfloor + 1, n-k-2\}$. We reduce from DISJ_m. Partition the elements of $[n] \setminus [m]$ into sets $I := \{m+1, \ldots, m+1+\lfloor k/2 \rfloor\}$ and $J := \{m+2+\lfloor k/2 \rfloor, \ldots, n\}$. Note that $|I| = \lfloor k/2 \rfloor + 1$. Furthermore, we have $|J| \ge \lceil k/2 \rceil + 1$, since

$$\begin{aligned} |J| &= n - (m+2+\lfloor k/2 \rfloor) + 1 \\ &= n - 1 - m - \lfloor k/2 \rfloor \\ &= n - 1 - m + \lceil k/2 \rceil - k \\ &= \lceil k/2 \rceil + 1 + n - 2 - m - k \\ &\ge \lceil k/2 \rceil + 1 \ , \end{aligned}$$

where the penultimate equality holds because $k = \lfloor k/2 \rfloor + \lceil k/2 \rceil$, and the inequality comes from the fact that $m \le n - k - 2$.

Let A' and B' be the sets received by Alice and Bob respectively as inputs to DISJ_m. Alice pads her input with elements from I until she gets a set of size $\lfloor k/2 \rfloor + 1$. Bob similarly pads his input with elements from J. Let $a := \lfloor k/2 \rfloor + 1 - |A'|$ and $b := \lceil k/2 \rceil + 1 - |B'|$. Specifically, Alice sets $A = A' \cup \{m+1, \ldots, m+a\}$ and Bob sets $B = B' \cup \{n, n-1, \ldots, n-b+1\}$.

Note that $|A| = \lfloor k/2 \rfloor + 1$, $|B| = \lceil k/2 \rceil + 1$, and $A \cap B = A' \cap B'$. Therefore, a solution to $k\text{-BAL-DISJ}(A, B)$ gives a solution to $\text{DISJ}_m(A', B')$, hence

$$R(k\text{-BAL-DISJ}) \ge R(\text{DISJ}_m) = \Omega(m) = \Omega(\min\{k, n-k\}) \ . \qquad \square$$

Gap-Equality. Alice and Bob are given n-bit strings x and y respectively and wish to compute

$$\text{GEQ}_{n,t}(x,y) := \begin{cases} 1 & \text{if } x = y \, , \\ 0 & \text{if } \Delta(x,y) = t \, , \\ * & \text{otherwise.} \end{cases}$$

We drop the subscripts when n is clear from context and $t = n/8$. When $\text{GEQ}(x,y) = *$, we allow the protocol to output 0 or 1. We are interested in $R^z(\text{GEQ})$; recall that $R^z(\text{GEQ})$ is the minimum communication cost of a protocol for GEQ that only makes mistakes when $\text{GEQ}(x,y) = z$. The standard public-coin EQUALITY protocol gives $R^0(\text{GEQ}) = O(1)$. For protocols that only err when $\text{GEQ}(x,y) = 1$, the complexity is drastically different.

Buhrman et al. (1998) proved an $\Omega(n)$ lower bound on the deterministic communication complexity of $\text{GEQ}_{n,n/2}$; their result extends to other gap sizes and to randomized protocols with one-sided error.

Lemma 2.7 (Buhrman et al. 1998)**.** *For all even $t = \Theta(n)$, we have $R^1(\text{GEQ}_{n,t}) = \Omega(n)$.*[4]

Gap-Hamming-Distance. Alice and Bob are given n-bit strings x and y respectively and wish to compute

$$\text{GHD}_{n,t}(x,y) := \begin{cases} 1 & \text{if } \Delta(x,y) \ge n/2 + t \, , \\ 0 & \text{if } \Delta(x,y) \le n/2 - t \, , \\ * & \text{otherwise.} \end{cases}$$

As in the definition of SET-DISJOINTNESS, it will occasionally be useful to view inputs to GHD as sets $A, B \subseteq [n]$ and to express GHD in terms of the size of the symmetric difference $|A \Delta B|$ rather than Hamming distance $\Delta(x,y)$. The standard gap size for GHD is $t = \sqrt{n}$. In this case, we drop the subscripts and use just GHD. A tight lower bound of $R(\text{GHD}) = \Omega(n)$ is known, due to Chakrabarti and Regev (2011).

[4] Curiously, the parity of t turns out to be necessary. Since $\Delta(x,y) = |x| + |y| - 2|x \wedge y|$, Alice and Bob can deterministically distinguish $x = y$ from $\Delta(x,y)$ being odd with a single bit of communication—Alice sends Bob the parity of $|x|$, and Bob computes the parity of $|x| + |y|$. This does not affect our property testing lower bounds.

Theorem 2.8 (Chakrabarti and Regev 2011)**.** $R(\text{GHD}) = \Omega(n)$.

For larger gap sizes, a padding argument[5] implicit in Brody et al. (2010), together with the aforementioned $\Omega(n)$ bound for GHD, shows that $R(\text{GHD}_{n,t}) = \Omega((n/t)^2)$ for all $t = \Omega(\sqrt{n})$.

When we require one-sided error, the situation changes.

Lemma 2.9. *For all $z \in \{0,1\}$ and all constant $0 < \delta < 1/2$, $R^z(\text{GHD}_{n,\delta n}) = \Omega(n)$.*

Proof. First, we prove a lower bound for $R^0(\text{GHD}_{n,\delta n})$. Let d be the least integer greater than or equal to δn, and let $m := n/2 + d$. We reduce from $\text{GEQ}_{m,2d}$. Specifically, let P be a protocol for $\text{GHD}_{n,\delta n}$ that only makes errors when $\text{GHD}_{n,\delta n}(x,y) = 0$. We use it to construct a protocol Q for $\text{GEQ}_{m,2d}$ that makes mistakes only when $x = y$. Given inputs $x, y \in \{0,1\}^m$, Alice constructs $\hat{x} := x \circ 0^{n-m}$, and Bob builds $\hat{y} := y \circ 1^{n-m}$. Then, they run protocol P and output $Q(x,y) := 1 - P(\hat{x},\hat{y})$. Note that if $x = y$, then $\Delta(\hat{x},\hat{y}) = n/2 - d \leq n/2 - \delta n$, hence $\text{GHD}(\hat{x},\hat{y}) = 0$. Similarly, when $\Delta(x,y) = 2d$, we have $\Delta(\hat{x},\hat{y}) = n/2 + d \geq n/2 + \delta n$, and $\text{GHD}(\hat{x},\hat{y}) = 1$. In either case, the new protocol correctly outputs $\text{GEQ}_{m,2d}(x,y)$ whenever P correctly computes $\text{GHD}_{n,\delta n}(\hat{x},\hat{y})$. Since P only makes mistakes when $\text{GHD}_{n,\delta n}(\hat{x},\hat{y}) = 0$, it follows that Q only makes mistakes when $\text{GEQ}_{m,2d}(x,y) = 1$. Therefore, $R^0(\text{GHD}_{n,\delta n}) \geq R^1(\text{GEQ}_{m,2d}) = \Omega(m) = \Omega(n)$.

Next, we prove a lower bound for $R^1(\text{GHD}_{n,\delta n})$. Observe that $\text{GHD}_{n,\delta n}(x,y) = 1 - \text{GHD}_{n,\delta n}(x, 1-y)$. Therefore, Alice and Bob can build a protocol for $\text{GHD}_{n,\delta n}$ which errs only when $\text{GHD}_{n,\delta n}(x,y) = 0$ from one which errs only when $\text{GHD}_{n,\delta n}(x,y) = 1$ by computing $\text{GHD}_{n,\delta n}(x, 1-y)$ and inverting the output. It follows that $R^1(\text{GHD}_{n,\delta n}) \geq R^0(\text{GHD}_{n,\delta n}) = \Omega(n)$.

In this way, we get a lower bound for $R^z(\text{GHD}_{n,\delta n})$ by embedding an instance of GEQ into *either side* of the GHD problem. □

We also consider an extended version of GHD. In $\text{EGHD}_{n,k,t}$, Alice and Bob's inputs x, y are n-bit strings, with the promise that $|x| = |y| = k$, and they wish to distinguish $\Delta(x,y) \geq k + t$ from $\Delta(x,y) \leq k - t$.

Lemma 2.10. *For all $k, t \leq n/2$, we have*

$$R(\text{EGHD}_{n,k,t}) = \Omega(\min\{(k/t)^2, k\}) \ .$$

In particular, we show that $\text{GHD}_{n,t}$ remains hard even under the promise that $|x| = |y| = n/2$.

Proof. First, we prove the lemma for the case $k = n/2$ by reduction from $\text{GHD}_{n/2,t/2}$. Let P be a protocol for $\text{EGHD}_{n,n/2,t}$. Given inputs $\hat{x}, \hat{y} \in \{0,1\}^{n/2}$ to $\text{GHD}_{n/2,t/2}$, Alice and Bob create n-bit strings x, y by mapping each bit $0 \to 01$ and each bit $1 \to 10$.[6] Then, they run protocol P on input (x,y) and output the result. Note that $|x| = |y| = n/2$. Furthermore, $\Delta(x,y) = 2\Delta(\hat{x},\hat{y})$. Therefore, if $\Delta(\hat{x},\hat{y}) \geq n/4 + t/2$ then $\Delta(x,y) \geq n/2 + t$, and similarly if $\Delta(\hat{x},\hat{y}) \leq n/4 - t/2$ then $\Delta(x,y) \leq n/2 - t$. It follows that a correct answer for $\text{EGHD}_{n.n/2,t}$ gives a correct answer to $\text{GHD}_{n/2,t/2}$, hence

$$\begin{aligned} R(\text{EGHD}_{n,n/2,t}) &\geq R(\text{GHD}_{n/2,t/2}) \\ &= \Omega\left(\min\left\{(n/t)^2, n\right\}\right) \\ &= \Omega\left(\min\left\{(k/t)^2, k\right\}\right). \end{aligned}$$

Proving the general case follows from a simple padding argument. Specifically, we reduce $\text{EGHD}_{2k,k,t}$ to $\text{EGHD}_{n,k,t}$. Given $2k$-bit strings $\hat{x}$ and $\hat{y}$, Alice and Bob construct n-bit strings x and y by setting $x := \hat{x} \circ 0^{n-2k}$ and $y := \hat{y} \circ 0^{n-2k}$. It is easy to see that $|x| = |y| = k$ and that $\Delta(x,y) = \Delta(\hat{x},\hat{y})$. Therefore, an answer to $\text{EGHD}_{n,k,t}(x,y)$ gives an answer to $\text{EGHD}_{2k,k,t}(\hat{x},\hat{y})$, hence

$$R(\text{EGHD}_{n,k,t}) \geq R(\text{EGHD}_{2k,k,t}) = \Omega\left(\min\{(k/t)^2, k\}\right). \qquad \square$$

[5]This padding argument reduces $\text{GHD}_{n,\sqrt{n}}$ to $\text{GHD}_{n,t}$ for any $\sqrt{n} < t \leq O(n)$. Choose $\hat{n}$ such that $n = (\hat{n}/t(\hat{n}))^2$, and let $m := \hat{n}/n$. Then, given inputs $x, y \in \{0,1\}^n$, Alice and Bob can compute $\text{GHD}_{n,\sqrt{n}}$ by repeating each string m times. The resulting strings $\hat{x}$ and $\hat{y}$ have length $\hat{n}$, and $\text{GHD}_{\hat{n},t(\hat{n})}(\hat{x},\hat{y}) = \text{GHD}_{n,\sqrt{n}}(x,y)$, hence a protocol for the former can be used to solve the latter. It follows that $R(\text{GHD}_{\hat{n},t(\hat{n})}) \geq R(\text{GHD}_{n,\sqrt{n}}) = \Omega(n) = \Omega((\hat{n}/t(\hat{n}))^2)$.

[6]Formally, Alice creates an n-bit string x by setting $x_{2i-1} := \hat{x}_i$ and $x_{2i} := 1 - \hat{x}_i$ for all $1 \leq i \leq n/2$. Similarly, Bob defines y by setting $y_{2i-1} := \hat{y}_i$ and $y_{2i} := 1 - \hat{y}_i$.

3 Testing k-Linearity and Related Properties

In this section we prove Theorem 1.1. Recall that a k-linear function is a function of the form $f(x) = \sum_{i \in S} x_i \pmod 2$ for some set $S \subseteq [n]$ of size $|S| = k$. We use k-LINEAR to denote the property that a function is k-linear. The definitions of the other properties in the statement of Theorem 1.1 are as follows:

Definition 3.1 (Junta)**.** The function $f : \{0,1\}^n \to \{0,1\}$ is a k-junta if there is a set $J \subseteq [n]$ of size $|J| \leq k$ such that for every $x, y \in \{0,1\}^n$ that satisfy $x_i = y_i$ for each $i \in J$, we have $f(x) = f(y)$. We use k-JUNTA to denote the property that a function is a k-junta.

Definition 3.2 (Low Fourier degree)**.** For convenience when discussing Fourier degree we will represent boolean functions using range $\{-1,1\}$ instead of $\{0,1\}$. It is well known that every boolean function $f : \{0,1\}^n \to \{-1,1\}$ has a unique representation of the form $f(x) = \sum_{S \subseteq [n]} \hat{f}(S)\chi_S(x)$, where $\chi_S = (-1)^{\sum_{i \in S} x_i}$ and $\hat{f}(S) \in \mathbb{R}$. The terms $\hat{f}(S)$ are the *Fourier coefficients* of f, and the *Fourier degree* of f is the maximum value of $k \geq 0$ such that $\hat{f}(S) \neq 0$ for some set S of size $|S| = k$.[7] We use DEGREE-k to denote the property that the Fourier degree of a function is at most k.

Definition 3.3 (Sparse polynomials)**.** Every boolean function $f : \{0,1\}^n \to \{0,1\}$ also has a unique representation as a polynomial over GF(2). We say that f is a *k-sparse polynomial* if its representation over GF(2) has at most k terms. Let k-SPARSE denote the property that a function has a k-sparse GF(2) representation.

The following facts about k-linear functions will be used in the proof of Theorem 1.1:

Proposition 3.4. *Fix $n > 2$ and $1 \leq k \leq n-2$. If $f : \{0,1\}^n \to \{0,1\}$ is $(k+2)$-linear, then f is*

(i) $\frac{1}{2}$*-far from k-linear functions,*

(ii) $\frac{1}{2}$*-far from k-juntas,*

(iii) $\frac{1}{2}$*-far from functions of Fourier degree at most k, and*

(iv) $\frac{1}{20}$*-far from k-sparse polynomials.*

Proof. We first prove part (iii). Parts (i) and (ii) will follow immediately from part (iii) and the observation that k-juntas and k-linear functions have Fourier degree at most k.

Let f be a $(k+2)$-linear function over the variables of some set $T \subseteq [n]$ of size $|T| = k+2$, and let g be any function of Fourier degree at most k. For convenience, we will represent f and g as functions from $\{0,1\}^n$ to $\{-1,1\}$. Since f is a linear function over the variables in T, we know that $\hat{f}(T) = 1$, and $\hat{f}(S) = 0$ for all $S \neq T$. Moreover, since g has Fourier degree k and $|T| > k$, we know by definition that $\hat{g}(T) = 0$. Thus by Parseval's theorem

$$\mathop{\mathrm{E}}_{x}[f(x)g(x)] = \sum_{S \subseteq [n]} \hat{f}(S)\hat{g}(S) = 0 \ ,$$

which implies $\Pr_x[f(x) \neq g(x)] = 1/2$.

Finally, part (iv) is a special case of a more general theorem of Diakonikolas et al. (2007, Thm. 36). For convenience, we provide a self-contained proof as Lemma A.1 in Section A. □

Theorem 1.1 (Restated)**.** *Fix $1 < k < n-1$. Then, at least $\Omega(\min\{k, n-k\})$ queries are required to test (i) k-linear functions, (ii) k-juntas, (iii) functions of Fourier degree at most k, and (iv) functions with k-sparse polynomial representation in* GF(2)*.*

Proof. We prove the lower bound for k-linear functions by reducing from the k-BAL-DISJ problem. Recall that $\mathrm{C}_{\oplus}^{k\text{-LINEAR}}$ is the communication game where the inputs are the functions $f, g : \{0,1\}^n \to \{0,1\}$ and the players must test whether the function $h = f \oplus g$ is k-linear. Lemmas 2.4 and 2.6 imply that

[7] For more details on the Fourier representation of boolean functions see, e.g., de Wolf 2008; O'Donnell 2008.

$$2\,Q(k\text{-LINEAR}) \geq R(\mathrm{C}_{\oplus}^{k\text{-LINEAR}})$$

and

$$R(k\text{-BAL-DISJ}) = \Omega(\min\{k, n-k\})\ .$$

To complete the proof, we show that $R(\mathrm{C}_{\oplus}^{k\text{-LINEAR}}) \geq R(k\text{-BAL-DISJ})$ with a reduction from k-BAL-DISJ to $\mathrm{C}_{\oplus}^{k\text{-LINEAR}}$.

Let $A, B \subseteq [n]$ be the two sets of size $|A| = \lfloor \frac{k}{2} \rfloor + 1$ and $|B| = \lceil \frac{k}{2} \rceil + 1$ received by Alice and by Bob, respectively, as the input to an instance of k-BAL-DISJ. Alice and Bob can construct the functions $\mathrm{Parity}_A, \mathrm{Parity}_B : \{0,1\}^n \to \{0,1\}$. When $|A \cap B| = 1$, the symmetric difference of the two sets has size $|A \triangle B| = |A| + |B| - 2|A \cap B| = k$, and the function $\mathrm{Parity}_A \oplus \mathrm{Parity}_B = \mathrm{Parity}_{A \triangle B}$ is k-linear. Conversely, when A and B are disjoint, the function $\mathrm{Parity}_A \oplus \mathrm{Parity}_B$ is a $(k+2)$-parity function and, by Fact 3.4, it is $\frac{1}{2}$-far from k-linear functions. So Alice and Bob can solve their instance of k-BAL-DISJ with a communication protocol for $\mathrm{C}_{\oplus}^{k\text{-LINEAR}}$. This implies that $R(\mathrm{C}_{\oplus}^{k\text{-LINEAR}}) \geq R(k\text{-BAL-DISJ})$, as we wanted to show.

The same reduction from k-BAL-DISJ yields lower bounds for testing the other properties. Let $\mathrm{C}_{\oplus}^{k\text{-JUNTA}}$ denote the communication game where Alice and Bob receive boolean functions f and g as inputs and must decide if $h = f \oplus g$ is a k-junta. Similarly, define $\mathrm{C}_{\oplus}^{\text{DEGREE-}k}$ and $\mathrm{C}_{\oplus}^{k\text{-SPARSE}}$ to be the corresponding communication games where Alice and Bob must decide if h has Fourier degree at most k or can be represented by a k-sparse GF(2) polynomial. In the k-BAL-DISJ reduction above, Alice and Bob create a joint function $\mathrm{Parity}_{A \triangle B}$ that is $(k+2)$-linear if A and B are disjoint, and k-linear if A and B intersect. By Fact 3.4, $(k+2)$-linear functions are $\frac{1}{2}$-far from k-juntas, $\frac{1}{2}$-far from functions with Fourier degree at most k, and $\frac{1}{20}$-far from k-sparse polynomials. Recalling that a k-linear function is a k-junta, has Fourier degree at most k, and can be represented by a k-sparse GF(2) polynomial, it follows that

$$R(k\text{-BAL-DISJ}) \leq \min\{R(\mathrm{C}_{\oplus}^{k\text{-JUNTA}}), R(\mathrm{C}_{\oplus}^{\text{DEGREE-}k}), R(\mathrm{C}_{\oplus}^{k\text{-SPARSE}})\}.$$

Together with Lemmas 2.4 and 2.6, we have

$$\begin{aligned} 2\,Q(k\text{-JUNTA}) &\geq R(\mathrm{C}_{\oplus}^{k\text{-JUNTA}}) = \Omega(\min\{k, n-k\})\ , \\ 2\,Q(\text{DEGREE-}k) &\geq R(\mathrm{C}_{\oplus}^{\text{DEGREE-}k}) = \Omega(\min\{k, n-k\})\ , \text{ and} \\ 2\,Q(k\text{-SPARSE}) &\geq R(\mathrm{C}_{\oplus}^{k\text{-SPARSE}}) = \Omega(\min\{k, n-k\})\ . \end{aligned}$$

□

Testing linear functions with 0-1 coefficients. With Oded Goldreich's kind permission, we present his proof of Theorem 1.2.

Theorem 1.2 (Restated)**.** *Testing the class of linear functions from* $\mathrm{GF}(3)^n$ *to* $\mathrm{GF}(3)$ *that have only* 0-1 *coefficients requires* $\Theta(n)$ *queries.*

Proof. The upper bound in the theorem follows from the query complexity of learning subclasses of linear functions over $\mathrm{GF}(3)^n$. See Goldreich (2010b) for the details.

For the lower bound, we establish a reduction from the SET-DISJOINTNESS problem. Let $\mathrm{C}_{+}^{\{0,1\}\text{-LIN}}$ be the communication game where Alice and Bob receive the functions $f, g : \mathrm{GF}(3)^n \to \mathrm{GF}(3)$, respectively, and must determine if the function $h = f + g$ (where the sum is taken pointwise over GF(3)) is linear and has only $\{0,1\}$-coefficients. By Lemma 2.4 and Theorem 2.5,

$$4\,Q(\{0,1\}\text{-LIN}) \geq R(\mathrm{C}_{+}^{\{0,1\}\text{-LIN}}) \quad \text{and} \quad R(\text{DISJ}) = \Omega(n)\ .$$

To complete the proof, it suffices to show that $R(\mathrm{C}_{+}^{\{0,1\}\text{-LIN}}) \geq R(\text{DISJ})$. We do so with a reduction from SET-DISJOINTNESS to the $\{0,1\}$-LIN testing communication game.

Let $a, b \in \{0,1\}^n$ be the strings received by Alice and Bob as input to the SET-DISJOINTNESS problem. Alice and Bob build the functions $f, g : \mathrm{GF}(3)^n \to \mathrm{GF}(3)$ defined by $f(x) = \sum_{i=1}^n a_i x_i$ and $g(x) = \sum_{i=1}^n b_i x_i$, respectively. The combined function $h = f + g$ is defined by $h(x) = \sum_{i=1}^n (a_i + b_i) x_i$. This function is clearly

linear. When a and b are disjoint, then every coefficient $a_i + b_i$ of h takes value 0 or 1. Conversely, when a and b are not disjoint, there is an index i for which $a_i + b_i = 2$. Then for any linear function ℓ with $\{0,1\}$-valued coefficients, the function $h - \ell$ is a non-zero linear function. The Schwartz-Zippel Lemma states that every non-zero linear function over $\mathrm{GF}(3)^n$ takes the value 0 on at most $\frac{1}{3}$ of the inputs from $\mathrm{GF}(3)^n$. Thus, h is $\frac{2}{3}$-far from all the linear functions with only $\{0,1\}$-valued coefficients. Therefore, Alice and Bob can run a protocol for $\mathrm{C}_+^{\{0,1\}\text{-LIN}}$ to solve their instance of SET-DISJOINTNESS and $R(\mathrm{C}_+^{\{0,1\}\text{-LIN}}) \geq R(\mathrm{DISJ})$. □

Testing computability by width-4 OBDDs. Again with the kind permission of Oded Goldreich, we present his proof of Theorem 1.3. We again remark that this result was obtained independently by Brody et al. (2011), who gave a similar proof.

The hard instances we use in this proof are those introduced by Goldreich (2010b). Let $\hat{n} := \lfloor \frac{n-1}{4} \rfloor$. We develop base functions from the following primitive functions. Consider the following four-bit functions $\phi_0, \phi_1, \phi_2, \phi_3 : \{0,1\}^4 \to \{0,1\}$:

$$\begin{aligned}
\phi_0(x_1, x_2, x_3, x_4) &:= 0 \ , \\
\phi_1(x_1, x_2, x_3, x_4) &:= x_1 x_3 \ , \\
\phi_2(x_1, x_2, x_3, x_4) &:= x_2 x_4 \ , \\
\phi_3(x_1, x_2, x_3, x_4) &:= x_1 x_3 \oplus x_2 x_4 \ .
\end{aligned}$$

Given $z = (z_1, \ldots, z_{\hat{n}}) \in \{0,1,2,3\}^{\hat{n}}$, define the function $h_z : \{0,1\}^n \to \{0,1\}$ by setting

$$h_z(x_1, \ldots, x_n) := x_1 \oplus \sum_{j=1}^{\hat{n}} \phi_{z_j}(x_{4j-2}, x_{4j-1}, x_{4j}, x_{4j+1}) \ .$$

Lemma 3.5 (Goldreich 2010b, Thm. 4.2)**.** *Fix $z \in \{0,1,2,3\}^{\hat{n}}$. If every coordinate $j \in [\hat{n}]$ of z satisfies $z_j \in \{0,1,2\}$ then h_z can be computed by a width-4 OBDD. Otherwise, if there exists $j \in [\hat{n}]$ such that $z_j = 3$ then h_z is $\frac{1}{16}$-far from all functions computable by width-4 OBDDs.*

Goldreich (2010b) uses Lemma 3.5 to show that testing computability by width-4 OBDDs requires $\Omega(\sqrt{n})$ queries. We combine this construction with our technique and get an $\Omega(n)$ lower bound.

Theorem 1.3 (Restated)**.** *Testing the class of functions that are computable by width-4 OBDDs requires $\Theta(n)$ queries.*

Proof. We reduce from $\mathrm{DISJ}_{\hat{n}}$. Given sets $A, B \subseteq [\hat{n}]$, Alice and Bob first build strings $a, b \in \{0,1,2,3\}^{\hat{n}}$ by setting

$$a_j := \begin{cases} 1 & \text{if } j \in A \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad b_j := \begin{cases} 2 & \text{if } j \in B \\ 0 & \text{otherwise} \end{cases}$$

Alice and Bob then define functions $f := h_a$ and $g := h_b$. We define the combining operator $\psi(f,g)$ to return the function h_z, where $z_j := a_j + b_j$. By our choice of base functions, for every $x \in \{0,1\}^n$ we have $h_z(x) = h_a(x) \oplus h_b(x) \oplus x_1$, so ψ is a simple combining operator. Also, our definitions of a, b, and z imply that $z_j = 3$ iff $j \in A \cap B$. By Lemma 3.5, this means that h_z is computable by a width-4 OBDD when A and B is disjoint and h_z is $\frac{1}{16}$-far from all functions computable by width-4 OBDDs when A and B intersect. Thus, $R(\mathrm{C}_\psi^{4\text{-OBDD}}) \geq R(\mathrm{DISJ}_{\hat{n}})$. This inequality, together with Lemma 2.4, Theorem 2.5, and the fact that $\hat{n} = \Theta(n)$, yields

$$2\,Q(4\text{-OBDD}) \geq R(\mathrm{C}_\psi^{4\text{-OBDD}}) \geq R(\mathrm{DISJ}_{\hat{n}}) = \Omega(n) \ .$$ □

4 Testing Monotonicity and Submodularity

Fix $R \subseteq \mathbb{R}$. Recall that the function $f : \{0,1\}^n \to R$ is *monotone* if for any two inputs $x, y \in \{0,1\}^n$ that satisfy $x_1 \leq y_1, \ldots, x_n \leq y_n$, we have $f(x) \leq f(y)$. In this section, we prove the following lower bound on the query complexity for testing monotonicity.

Theorem 1.4 (Restated)**.** *Testing $f : \{0,1\}^n \to R$ for monotonicity requires $\Omega(\min\{n, |R|^2\})$ queries.*

Proof. We first consider the case where $R = \mathbb{R}$. We prove the lower bound for testing monotonicity in this case with a reduction from SET-DISJOINTNESS. Let ψ be the combining operator that, given two functions $f, g : \{0,1\}^n \to \{-1,1\}$, returns the function $h : \{0,1\}^n \to \mathbb{Z}$ defined by $h(x) := 2\,|x| + f(x) + g(x)$. Define $\mathrm{C}_\psi^{\text{MONO}}$ be the communication game where Alice and Bob are given two functions $f, g : \{0,1\}^n \to \{-1,1\}$ and they must test whether h is monotone. By Lemma 2.4 and Theorem 2.5,

$$2\,Q(\text{MONO}) \geq R(\mathrm{C}_\psi^{\text{MONO}}) \quad \text{and} \quad R(\text{DISJ}) = \Omega(n)\ .$$

We complete the proof by showing that $R(\mathrm{C}_\psi^{\text{MONO}}) \geq R(\text{DISJ})$.

Let $A, B \subseteq [n]$ be the subsets received by Alice and Bob as the input to an instance of the SET-DISJOINTNESS problem. Alice and Bob build functions $\chi_A, \chi_B : \{0,1\}^n \to \{-1,1\}$, respectively, by setting $\chi_A(x) = (-1)^{\sum_{i\in A} x_i}$ and $\chi_B(x) = (-1)^{\sum_{i\in B} x_i}$. Let $h := \psi(\chi_A, \chi_B)$. Note that $h(x) = 2\,|x| + \chi_A(x) + \chi_B(x)$. We claim that (a) when A and B are disjoint, h is monotone, and (b) when A and B are not disjoint, h is $\frac{1}{8}$-far from monotone. If this claim is true, then we have completed our lower bound since it implies that Alice and Bob can run a protocol for $\mathrm{C}_\psi^{\text{MONO}}$ to solve their instance of SET-DISJOINTNESS and, therefore, $R(\mathrm{C}_\psi^{\text{MONO}}) \geq R(\text{DISJ})$.

We now prove Claim (a). Fix $i \in [n]$. For $x \in \{0,1\}^n$, let $x^0, x^1 \in \{0,1\}^n$ be the vectors obtained by fixing the ith coordinate of x to 0 and to 1, respectively. Note that for any set $S \subseteq [n]$, $\chi_S(x^1) = -\chi_S(x^0)$ if $i \in S$, and $\chi_S(x^1) = \chi_S(x^0)$ otherwise. So when $i \notin A$ and $i \notin B$,

$$h(x^1) - h(x^0) = 2\,|x^1| - 2\,|x^0| = 2 > 0\ ,$$

when $i \in A$ and $i \notin B$,

$$h(x^1) - h(x^0) = 2\,|x^1| - 2\,|x^0| - 2\,\chi_A(x^0) \geq 0\ ,$$

and when $i \notin A$ and $i \in B$,

$$h(x^1) - h(x^0) = 2\,|x^1| - 2\,|x^0| - 2\,\chi_B(x^0) \geq 0\ .$$

Those three inequalities imply that when $i \notin A \cap B$, the function h is monotone on each edge (x^0, x^1) in the ith direction. As a result, when A and B are disjoint the function h is monotone.

Let us now prove Claim (b). Let $A \cap B \neq \emptyset$. When $i \in A \cap B$,

$$h(x^1) - h(x^0) = 2\,|x^1| - 2\,|x^0| - 2\,\chi_A(x^0) - 2\,\chi_B(x^0).$$

This implies that for each x that satisfy $\chi_A(x^0) = \chi_B(x^0) = 1$, it holds that $h(x^1) < h(x^0)$. Partition $\{0,1\}^n$ into 2^{n-1} pairs that form the endpoints to all the edges in the ith direction. We claim that at least $\frac{1}{4}$ of these pairs satisfy the condition $\chi_A(x^0) = \chi_B(x^0) = 1$. To see this, note that when $A = \{i\}$, then $\chi_A(x^0) = 1$ with certainty. On the other hand, if $j \in A$ for some $j \neq i$, then take any x^0, and let $\hat{x}^0$ be x^0 with the jth bit flipped. Then, $\chi_A(x^0) = -\chi_A(\hat{x}^0)$. It follows that $\chi_A(x^0) = 1$ with probability exactly $\frac{1}{2}$. A similar argument holds independently for $\chi_B(x^0)$. Therefore, at least $\frac{1}{4}$ of these pairs will satisfy the condition $\chi_A(x^0) = \chi_B(x^0) = 1$, and for each of these pairs, either $h(x^0)$ or $h(x^1)$ must be modified to make h monotone. Therefore, when A and B intersect, we need to modify at least $2^n/8$ entries, just to correct the violated edges in the ith direction. It follows that h is at least $\frac{1}{8}$-far from monotone.

Suppose now that $|R| \geq 12\sqrt{n}+5$. Without loss of generality assume $R \supseteq \{n - 6\sqrt{n} - 2, \ldots, n + 6\sqrt{n} + 2\}$.[8] As before, we do a reduction from SET-DISJOINTNESS. Alice and Bob receive sets $A, B \subseteq [n]$, respectively,

[8]This essentially boils down to a renaming of R. Formally, we prove a lower bound for testing $\hat{f} : \{0,1\}^n \to \hat{R}$ for an arbitrary $\hat{R}$ with $|\hat{R}| \geq 12\sqrt{n} + 5$ by reducing from the problem of testing $f : \{0,1\}^n \to R$. Let $\phi : R \to \hat{R}$ be the bijection that maps the ith least element of R to the ith least element of $\hat{R}$, and define $\hat{f}(x) := \phi(f(x))$. $\hat{f}$ is monotone if f is monotone, and $\hat{f}$ is ϵ-far from monotone if f is ϵ-far from monotone. Thus, testing $\hat{f}$ for monotonicity has the same query complexity as testing f for monotonicity.

and build the functions $\chi_A, \chi_B : \{0,1\}^n \to \{-1,1\}$. The modification to the construction is in the definition of the combining operator. We now define $\psi(\chi_A, \chi_B)$ to return the function h' defined by

$$h'(x) := \begin{cases} h(x) & \text{if } \frac{n}{2} - 3\sqrt{n} \leq |x| \leq \frac{n}{2} + 3\sqrt{n} \ , \\ n - 6\sqrt{n} - 2 & \text{if } |x| < \frac{n}{2} - 3\sqrt{n} \ , \\ n + 6\sqrt{n} + 2 & \text{if } |x| > \frac{n}{2} + 3\sqrt{n} \ . \end{cases}$$

Note that when $\frac{n}{2} - 3\sqrt{n} \leq |x| \leq \frac{n}{2} + 3\sqrt{n}$, we have $n - 6\sqrt{n} - 2 \leq h(x) \leq n + 6\sqrt{n} + 2$. The definition of h' takes strings with low Hamming weight and "rounds $h(x)$ up" to $n - 6\sqrt{n} - 2$. In the same way, it takes strings of high Hamming weight and "rounds $h(x)$ down" to $n + 6\sqrt{n} + 2$. We claim that this preserves monotonicity when h is monotone, while ensuring that when h is far from monotone, our new function h' remains reasonably far from monotone.

Claim 4.1. *If A and B are disjoint, then h' is monotone. If A and B intersect, then h' is $\frac{1}{16}$-far from monotone.*

Proof. Suppose that $A \cap B = \emptyset$. We proceed in a manner similar to the general case. Fix any $i \in [n]$, and for $x \in \{0,1\}^n$, let x^0 and x^1 be the vectors obtained by setting the ith bit to 0 and 1 respectively. If A and B are disjoint, then h is monotone. When $|x^0|$ and $|x^1|$ both lie in the range $\{\frac{n}{2} - 3\sqrt{n}, \ldots, \frac{n}{2} + 3\sqrt{n}\}$, then $h'(x^1) = h(x^1) \geq h(x^0) = h'(x^0)$, so monotonicity is preserved. If $|x^0|$ and $|x^1|$ are either both less than $\frac{n}{2} - 3\sqrt{n}$ or both greater than $\frac{n}{2} + 3\sqrt{n}$, then monotonicity is trivially preserved, as h' is constant on each of these ranges. It remains to show that monotonicity is preserved when $|x^0|$ and $|x^1|$ lie in different ranges. But $h'(x) = h(x) \in \{n - 6\sqrt{n} - 2, \ldots, n + 6\sqrt{n} + 2\}$ for all x in this middle range. Therefore, $h'(x) \leq h'(y) \leq h'(z)$ for all x, y, z such that $|x| < \frac{n}{2} - 3\sqrt{n}$, $\left||y| - \frac{n}{2}\right| \leq 3\sqrt{n}$, and $|z| > \frac{n}{2} + 3\sqrt{n}$.

If $A \cap B \neq \emptyset$ then h is 1/8-far from monotone. We claim that h' is at most 1/16-far from h. To see this, let $x \in \{0,1\}^n$ be a uniform random string. By the Chernoff Bound,

$$\Pr\left[\left||x| - \tfrac{n}{2}\right| > 3\sqrt{n}\right] < 0.03 < 1/16 \ .$$

Furthermore, $h'(x) = h(x)$ whenever $\frac{n}{2} - 3\sqrt{n} \leq |x| \leq \frac{n}{2} + 3\sqrt{n}$, hence $\Pr[h'(x) \neq h(x)] \leq 1/16$.

Suppose for the sake of contradiction that h' is *not* $\frac{1}{16}$-far from monotone, and let $\tilde{h}$ denote the monotone function closest to h'. By the triangle inequality and the fact that h' is at most 1/16-far from h, we have

$$\Pr[h(x) \neq \tilde{h}(x)] \leq \Pr[h(x) \neq h'(x)] + \Pr[h'(x) \neq \tilde{h}(x)] < 1/8 \ .$$

This violates the fact that h is 1/8-far from monotone. □

It is possible to make the above argument much tighter, and get the corresponding linear query complexity lower bound for a smaller range of $|R|$, although $|R|$ remains $\Omega(\sqrt{n})$. We chose the above range size to maximize clarity.

Finally, suppose that $|R| = o(\sqrt{n})$. We prove the lower bound for testing monotonicity in this case via a reduction from the $|R| \geq 12\sqrt{n} + 5$ case. Let $\mathcal{A}$ be an optimal monotonicity testing algorithm for functions $f : \{0,1\}^n \to R$, and let m be the greatest integer such that $|R| \geq 12\sqrt{m} + 5$. We will use $\mathcal{A}$ to construct a monotonicity testing algorithm $\mathcal{A}'$ for functions $g : \{0,1\}^m \to R$. The construction of $\mathcal{A}'$ depends on the following claim.

Claim 4.2. *Given a function $g : \{0,1\}^m \to R$, there exists a function $h : \{0,1\}^n \to R$ with the following properties:*

1. *If g is monotone, then h is monotone.*
2. *If g is ϵ-far from monotone, then h is ϵ-far from monotone.*
3. *For all $x \in \{0,1\}^n$, the value of $h(x)$ can be determined with one query to g.*

Proof. We construct $h : \{0,1\}^n \to R$ from g by padding. Specifically, define $h(x,y) := g(x)$ for strings $x \in \{0,1\}^m$ and $y \in \{0,1\}^{n-m}$. This construction clearly satisfies the first and third conditions of the claim. For the second condition, we prove the contrapositive. Suppose that h is *not* ϵ-far from monotone. Let $\tilde{h}$ be the monotone function closest to h; thus, $\Pr_{x,y}[\tilde{h}(x,y) \neq h(x,y)] < \epsilon$. By an averaging argument, there exists $\tilde{y} \in \{0,1\}^{n-m}$ such that $\Pr_x[\tilde{h}(x,\tilde{y}) \neq h(x,\tilde{y})] < \epsilon$. Define $\tilde{g} : \{0,1\}^m \to R$ as $\tilde{g}(x) := \tilde{h}(x,\tilde{y})$. Then $\tilde{g}$ is monotone and $\Pr_x[\tilde{g}(x) \neq g(x)] = \Pr_x[\tilde{h}(x,\tilde{y}) \neq h(x,\tilde{y})] < \epsilon$, so g is *not* ϵ-far from monotone. □

The construction of $\mathcal{A}'$ is simple. Given input $g : \{0,1\}^m \to R$, let h be the function guaranteed by Claim 4.2. $\mathcal{A}'$ runs $\mathcal{A}$ on h and accepts if and only if $\mathcal{A}$ accepts h. By Claim 4.2, if g is monotone, then h is monotone, and if g is ϵ-far from monotone, then h is ϵ-far from monotone. The correctness of $\mathcal{A}'$ then follows from the correctness of $\mathcal{A}$. Furthermore, by Claim 4.2, $\mathcal{A}'$ uses one query per query of $\mathcal{A}$. However, testing $g : \{0,1\}^m \to R$ for monotonicity requires $\Omega(m)$ queries, because g has range size $|R| \geq 12\sqrt{m} + 5$. Since $\mathcal{A}$ is an optimal monotonicity tester and uses the same number of queries as $\mathcal{A}'$, it follows that testing $f : \{0,1\}^n \to R$ for monotonicity requires $\Omega(m) = \Omega(|R|^2)$ queries when $|R| = o(\sqrt{n})$. □

Testing submodularity. The function $f : \{0,1\}^n \to \mathbb{R}$ is *submodular* if for every $x, y \in \{0,1\}^n$, $f(x \vee y) + f(x \wedge y) \geq f(x) + f(y)$, where $(x \vee y)_i = \max\{x_i, y_i\}$ and $(x \wedge y)_i = \min\{x_i, y_i\}$. Testing submodularity was first studied by Parnas, Ron, and Rubinfeld (2003) for functions over rectangles. Seshadhri and Vondrák (2011) initiated the study of submodularity testing for functions over the boolean hypercube and, in particular, showed that testing submodularity is at least as difficult as testing monotonicity. Specifically, they established the following result.

Lemma 4.3 (Seshadhri and Vondrák (2011))**.** *Given the function $f : \{0,1\}^n \to \mathbb{R}$, there exists a function $g : \{0,1\}^{n+1} \to \mathbb{R}$ with the following properties:*

1. *If f is monotone, then g is submodular.*
2. *If f is ϵ-far from monotone, then g is $\frac{\epsilon}{2}$-far from submodular.*
3. *For each $x \in \{0,1\}^{n+1}$, the value of $g(x)$ can be determined with 2 queries to f.*

Combining the lemma with the lower bound of Fischer et al. (2002) on testing monotonicity yields a lower bound of $\Omega(\log n)$ queries for testing submodularity non-adaptively. This implies a weak lower bound of $\Omega(\log \log n)$ queries for general (i.e., adaptive) submodularity testers. Combining Lemma 4.3 with Theorem 1.4 instead, we get a stronger lower bound.

Corollary 1.5 (Restated)**.** *Testing $f : \{0,1\}^n \to \mathbb{R}$ for submodularity requires $\Omega(n)$ queries.*

Proof. Consider the task of testing whether $f : \{0,1\}^{n-1} \to \mathbb{R}$ is monotone. Let $g : \{0,1\}^n \to \mathbb{R}$ be the corresponding function whose existence is guaranteed by Lemma 4.3. We can test whether f is monotone by simulating a submodularity tester T on g. If T makes q queries, the resulting monotonicity tester makes a total of $2q$ queries. By Theorem 1.4, all monotonicity testers must make at least $\Omega(n)$ queries, so our submodularity tester must also make $q = \Omega(n)$ queries. □

5 Testing Concise Representations

We begin with formal definitions for decision trees and branching programs.

Definition 5.1 (Decision tree)**.** A *decision tree* is a directed binary tree in which each internal node is labelled with some element from $[n]$, the two edges going out of an internal node are labelled with 0 and 1, and each leaf node has a label from $\{0,1\}$. The decision tree D computes the function $f : \{0,1\}^n \to \{0,1\}$ if for every $x \in \{0,1\}^n$, the path defined in D by querying the value of x_i at each internal node labelled with i and following the corresponding edge leads to a leaf node with value $f(x)$. The *size* of a decision tree is the total number of leaves it contains.

Definition 5.2 (Branching program)**.** A *branching program* is a directed acyclic graph with two sink nodes labelled with 0 and 1, respectively, and where all other nodes have out-degree 2. Each non-sink node has a label from $[n]$ and the two edges leaving a node are labelled with 0 and 1, respectively. The branching program P computes the function $f : \{0,1\}^n \to \{0,1\}$ if each $x \in \{0,1\}^n$ defines a path in the branching program that leads to the sink labelled with $f(x)$. The *size* of a branching program is the total number of nodes it contains.

The proof of Theorem 1.6 relies on the following two simple lemmas.

Lemma 5.3. *Fix $s \geq 1$ and $0 < \alpha < 1$. Let $f : \{0,1\}^n \to \{0,1\}$ be an s-linear function. Then f can be computed by a decision tree of size 2^s and is $\frac{1-\alpha}{2}$-far from all functions that are computable by decision trees of size at most $\alpha\, 2^s$.*

Proof. To construct a decision tree of size 2^s that computes the function $f : x \mapsto x_{i_1} \oplus \cdots \oplus x_{i_s}$, create a complete tree of depth s where each node at level j of the tree queries x_{i_j}. This tree has 2^s leaves and, by setting the value of each leaf appropriately, computes the function f exactly.

Consider now a decision tree T of size at most $\alpha\, 2^s$, and let $g : \{0,1\}^n \to \{0,1\}$ be the function computed by this tree. We want to show that $\Pr[f(x) \neq g(x)] \geq \frac{1-\alpha}{2}$ when the probability is over the uniform distribution of $x \in \{0,1\}^n$. For each leaf ℓ of T, let $\mathrm{depth}(\ell)$ denote the number of unique variables queried by the nodes in the path from the root of T to ℓ and let $R_\ell \subseteq \{0,1\}^n$ represent the set of inputs $x \in \{0,1\}^n$ that define a path in T that reaches ℓ. (Note that the sets R_ℓ form a partition of $\{0,1\}^n$.) Define $B := \bigcup_{\ell\,:\,\mathrm{depth}(\ell)<s} R_\ell$ to be the union of the sets R_ℓ for all the leaves in T of depth strictly less than s. Then

$$\begin{aligned} \Pr[f(x) \neq g(x)] &\geq \Pr[f(x) \neq g(x) \cap x \in B] \\ &= \Pr[x \in B] \cdot \Pr[f(x) \neq g(x) \mid x \in B]. \end{aligned}$$

For any leaf ℓ of T, the probability that an input x chosen uniformly at random from $\{0,1\}^n$ reaches ℓ is $2^{-\mathrm{depth}(\ell)}$. By the union bound, the probability that x reaches a leaf of depth at least s in T is at most $\alpha\, 2^s \cdot 2^{-s} = \alpha$, so $\Pr[x \in B] \geq 1 - \alpha$.

Let ℓ be a leaf in T of depth at most $s-1$. Then there is some index $i \in \{i_1, \ldots, i_s\}$ that is not queried in the path from the root of T to ℓ. We can partition R_ℓ into pairs $(x, x^{(i)})$ where each pair is identical in all but the i-th coordinate. For each such pair, $f(x) \neq f(x^{(i)})$ so no matter what label is attached to the leaf ℓ, we have $\Pr[f(x) \neq g(x) \mid x \in R_\ell] = \frac{1}{2}$. This also implies that $\Pr[f(x) \neq g(x) \mid x \in B] = \frac{1}{2}$ and, therefore, $\Pr[f(x) \neq g(x)] \geq (1-\alpha) \cdot \frac{1}{2} = \frac{1-\alpha}{2}$, as we wanted to show. □

Lemma 5.4. *Let $\mathcal{P}$ be the class of all boolean functions computable by branching programs of size $2s$. Then every s-linear function is in $\mathcal{P}$ while every $(s+2)$-linear function is $\frac{1}{6}$-far from $\mathcal{P}$.*

Proof. Again, the first assertion is almost immediate: consider a branching program of width 2 that queries x_{i_1} at the start node and queries x_{i_j} on both nodes at level $1 < j \leq s$. We can arrange the edges of this branching program so that the left (resp., right) node at level j is reached when $x_{i_1} \oplus \cdots \oplus x_{i_{j-1}}$ equals 0 (resp., equals 1). This branching program has size $2s-1$ and computes the s-linear function.

For the second assertion, let P be a branching program of size $2s$, and suppose it is close to some $(s+2)$-linear function h. Note that if one of the $s+2$ variables in h does not appear in P, then h and P are $\frac{1}{2}$-far, since for every input there is a variable whose value we can flip to change the value of h without changing the output of P.

Thus, we assume that every variable in h appears in P. Moreover, since P has only $2s$ nodes, there must be at least two variables in h that are queried only once in P. Let x_1 and x_2 denote two such variables, and let u_1 and u_2 denote the corresponding nodes in P. The graph of P is directed and acyclic, so we can assume without loss of generality that no path reaches the node u_1 after reaching u_2.

Consider the paths in P generated by strings $x, x^{(1)} \in \{0,1\}^n$, where x is generated uniformly at random and $x^{(1)}$ is generated from x by flipping x_1. Note that $x^{(1)}$ is also uniform. If the random path generated by x reaches u_2 with probability less than $\frac{2}{3}$, then with probability at least $\frac{1}{3}$, flipping the value of x_2 changes the value of h without changing the output of P; hence, P is $\frac{1}{6}$-far from h. On the other hand, if this random

path reaches u_2 with probability at least $\frac{2}{3}$, then the path generated by $x^{(1)}$ also reaches u_2 with probability $\frac{2}{3}$. By the union bound, the probability that *both* x and $x^{(1)}$ describe paths in P reaching u_2 is at least $\frac{1}{3}$. But since u_1 cannot be reached after u_2, this means that both x and $x^{(1)}$ describe paths to the same terminal in P even though they have different values in h. Therefore, P is $\frac{1}{6}$-far from h in this case too. □

We are now ready to complete the proof of Theorem 1.6.

Theorem 1.6 (Restated)**.** *At least* $\Omega(\min\{s, n-s\})$ *queries are required to test (i) size-*2^s *decision trees and (ii) size-*s *branching programs.*

Proof. The proof is nearly identical to that of Theorem 1.1. We prove the lower bound with a reduction from the balanced version of the SET-DISJOINTNESS problem. Let $\mathrm{C}_{\oplus}^{2^s\text{-DT}}$ and $\mathrm{C}_{\oplus}^{s\text{-BP}}$ be the communication games where Alice and Bob receive the functions $f, g : \{0,1\}^n \to \{0,1\}$ and must test whether the function $h = f \oplus g$ is computable by size-2^s decision trees or by size-s branching programs, respectively. By Lemmas 2.4 and 2.6,

$$\begin{aligned} 2\,Q(\text{size-}2^s\text{ D.T.'s}) &\geq R(\mathrm{C}_{\oplus}^{2^s\text{-DT}}), \\ 2\,Q(\text{size-}s\text{ B.P.'s}) &\geq R(\mathrm{C}_{\oplus}^{s\text{-BP}}),\text{ and} \\ R(s\text{-BAL-DISJ}) &= \Omega(\min\{s, n-s\}). \end{aligned}$$

To complete the proof, it suffices to show that $R(s\text{-BAL-DISJ})$ is a lower bound for $R(\mathrm{C}_{\oplus}^{2^s\text{-DT}})$ and $R(\mathrm{C}_{\oplus}^{s\text{-BP}})$.

Let $A, B \subseteq [n]$ be the two sets of size $|A| = \lfloor \frac{s}{2} \rfloor + 1$ and $|B| = \lceil \frac{s}{2} \rceil + 1$ received by Alice and by Bob, respectively, as the input to an instance of s-BAL-DISJ. Alice and Bob can construct the functions $\mathrm{Parity}_A, \mathrm{Parity}_B : \{0,1\}^n \to \{0,1\}$. When $|A \cap B| = 1$, the function $h = \mathrm{Parity}_A \oplus \mathrm{Parity}_B = \mathrm{Parity}_{A \triangle B}$ is s-linear. Such a function can be computed by size-2^s decision trees and by size-s branching programs. When A and B are disjoint, the function h is $(s{+}2)$-linear. By Lemmas 5.3 and 5.4, when h is $(s{+}2)$-linear, it is $\frac{3}{8}$-far from all functions computable by decision trees of size 2^s $(= \frac{1}{4}2^{s+2})$ and it is $\frac{1}{6}$-far from all functions computable by branching programs of size s. So Alice and Bob can solve their instance of s-BAL-DISJ with a communication protocol for $\mathrm{C}_{\oplus}^{2^s\text{-DT}}$ or for $\mathrm{C}_{\oplus}^{s\text{-BP}}$. □

Testing juntas. Fischer et al. (2004) asked if it is easier to test k-juntas if we are only required to reject functions that are far from $(k{+}t)$-juntas for some $t > 0$. The lower bound of Chockler and Gutfreund (2004) gives a lower bound of $\Omega(k/t)$ queries for this task. (See also (Diakonikolas et al., 2007, App. E).) This bound is not sufficiently strong to answer Fischer et al.'s question for any $t = \omega(1)$.

The following result shows that for any $t \leq O(\sqrt{k})$, the task of distinguishing k-juntas from functions that are far from $(k+t)$-juntas requires (asymptotically) as many queries as the standard k-junta testing problem.

Theorem 1.7 (Restated)**.** *Fix* $k \leq \frac{n}{2}$ *and* $t > 0$*. Any algorithm that accepts* k*-juntas and rejects functions far from* $(k+t-2)$*-juntas with high probability must make* $\Omega\big(\min\{(\frac{k}{t})^2, k\}\big)$ *queries.*

Proof. We prove the theorem with a reduction from the extended Gap Hamming Distance problem. If $t = \Omega(k)$, there is nothing to prove. Otherwise, suppose $t = o(k)$, and let $\mathrm{C}_{\oplus}^{(k,t)\text{-JUNTA}}$ be the communication game where Alice and Bob receive the functions $f, g : \{0,1\}^n \to \{0,1\}$ and must distinguish between the case where $h = f \oplus g$ is a k-junta from the case where h is far from $(k+t-2)$-juntas. By Lemmas 2.4 and 2.10 and the fact that $k + t/2 = \Theta(k)$, we have

$$2\,Q\big((k,t)\text{-JUNTA}\big) \geq R(\mathrm{C}_{\oplus}^{(k,t)\text{-JUNTA}})$$

and

$$R(\mathrm{EGHD}_{n,k+\frac{t}{2},\frac{t}{2}}) = \Omega\big(\min\{(\tfrac{k}{t})^2, k\}\big)\ .$$

To complete the proof, we want to show that $R(\mathrm{C}_{\oplus}^{(k,t)\text{-JUNTA}}) \geq R(\mathrm{EGHD}_{n,2k+t,\frac{t}{2}})$.

Let x and y be the strings received by Alice and Bob, respectively, as input to an instance of the $\textsc{eghd}_{n,k+\frac{t}{2},\frac{t}{2}}$ problem. Alice and Bob then construct the functions $\text{Parity}_A, \text{Parity}_B : \{0,1\}^n \to \{0,1\}$, where $A := \{i : x_i = 1\}$ and $B := \{i : y_i = 1\}$. The function $h = \text{Parity}_A \oplus \text{Parity}_B = \text{Parity}_{A \triangle B}$ is $|A \triangle B|$-linear. When $\Delta(x,y) = |A \triangle B| \leq k + t/2 - t/2 = k$, the function h is a k-parity function. Conversely, when $\Delta(x,y) \geq k + t/2 + t/2 = k + t$, by Fact 3.4(ii), the function h is $\frac{1}{2}$-far from all $(k+t-2)$-juntas. Therefore, Alice and Bob can run a protocol for the game $\text{C}_{\oplus}^{(k,t)\text{-JUNTA}}$ to solve their instance of the Extended Gap Hamming Distance problem and, as we wanted to show, $R(\text{C}_{\oplus}^{(k,t)\text{-JUNTA}}) \geq R(\textsc{eghd}_{n,k+\frac{t}{2},\frac{t}{2}})$. □

Remark. The conference version of this paper (Blais et al., 2011) used a different argument to prove Theorems 1.6 and 1.7. During the review process, a flaw was found in that argument. For a retraction of the earlier version of Theorem 1.6 and a discussion of the error, see (Blais et al., 2012).

6 Testers with One-Sided Error

Testing decision trees. We saw in the last section that $\Omega(\log s)$ queries are required to test whether a function can be represented as a boolean decision tree with at most s nodes. For testers with one-sided error, we get an exponentially larger bound.

Theorem 1.8 (Restated)**.** *At least $\Omega(s)$ queries are required to test size-s decision trees with one-sided error.*

Proof. We first consider the case where $s = 2^{n-1}$ for some $n \geq 5$. We prove this case with a reduction from the GAP-EQUALITY problem on s-bit strings. Let $\text{C}_{\oplus}^{s\text{-DT}}$ be the communication game where Alice and Bob receive the functions $f, g : \{0,1\}^n \to \{0,1\}$ and they must test whether the function $h = f \oplus g$ is computable by a decision tree of size s. By Lemmas 2.4 and 2.7,

$$2\,Q^1\big(s\text{-DT}\big) \geq R^1(\text{C}_{\oplus}^{s\text{-DT}}) \quad \text{and} \quad R^1(\textsc{geq}_{s,\frac{s}{8}}) = \Omega\big(s\big)\ .$$

We complete the proof by showing that $R^1(\text{C}_{\oplus}^{s\text{-DT}}) \geq R^1(\textsc{geq}_{s,\frac{s}{8}})$.

Let $a, b \in \{0,1\}^s$ be received by Alice and Bob as input to an instance of the GAP-EQUALITY problem. They must determine if $a = b$ or whether $\Delta(a,b) = \frac{s}{8}$. Alice and Bob can solve their instance of the GEQ problem with the following protocol. Let the set of vectors $x \in \{0,1\}^n$ with even parity $\text{Parity}(x) = x_1 \oplus \cdots \oplus x_n = 0$ define an indexing of the bits of a. (I.e., fix a bijection between those strings and $[s]$.) Alice and Bob build the functions $f, g : \{0,1\}^n \to \{0,1\}$ by setting

$$f(x) = \begin{cases} a_x & \text{when } \text{Parity}(x) = 0, \\ 0 & \text{when } \text{Parity}(x) = 1, \end{cases}$$

and

$$g(x) = \begin{cases} b_x & \text{when } \text{Parity}(x) = 0, \\ 1 & \text{when } \text{Parity}(x) = 1. \end{cases}$$

Alice and Bob then test whether $f \oplus g$ can be represented with a decision tree of size at most $\frac{15}{16}2^n$; when it can, they answer $\Delta(a,b) = \frac{s}{8}$.

Let us verify the correctness of this protocol. For any $x \in \{0,1\}^n$ where $\text{Parity}(x) = 0$, we have that $(f \oplus g)(x) = a_x \oplus b_x$. Furthermore, for each x where $\text{Parity}(x) = 1$, we get $(f \oplus g)(x) = 1$. So when $a = b$, then $f \oplus g$ is the Parity function. By Lemma 5.3, this function is $\frac{1}{32}$-far from every decision tree of size at most $\frac{15}{16}2^n$. When $\Delta(a,b) = \frac{s}{8}$, consider the (complete) tree that computes $f \oplus g$ by querying x_i in every node at level i. This tree has 2^n leaves, but for every input x where $a_x \neq b_x$, we have that the corresponding leaf has the same value as its sibling. So for each such input, we can eliminate one leaf. Therefore, we can compute $f \oplus g$ with a decision tree of size at most $2^n - 2^{n-1}/8 < \frac{15}{16}2^n$. □

Testing signed k-majorities. Our next bound is for testing whether a function $f : \{-1,1\}^n \to \{-1,1\}$ is a signed k-majority (for convenience, in this section we will switch notation and represent boolean values with ± 1 notation). A *signed majority* is a majority function with some variables negated, i.e. it is a halfspace of the form $f(x) = sgn(w \cdot x)$, where $w \in \{-1,1\}^n$. If $w \in \{-1,0,1\}^n$ and exactly k of the w_i's are non-zero, we say it is a *signed k-majority*. (A signed majority function is thus also a signed n-majority function.)

Signed majorities were studied by Matulef et al. (2009), who referred to them as $\{-1,1\}$-*weight halfspaces*. In that work, they showed a non-adaptive lower bound of $\Omega(\log n)$ queries to test whether a function is a signed majority on all n variables. Blais and O'Donnell (2010) studied the related problem of testing whether a function is a (non-signed) majority on exactly k out of n variables. When $k \leq \frac{3}{4}n$, they showed a lower bound of $\Omega(k^{1/12})$ queries for non-adaptive algorithms with two-sided error.

We show that $\Omega(k/\log k)$ queries are required to test whether f is a signed k-majority with one-sided error. The argument of Blais and O'Donnell (2010) can be adapted to show a non-adaptive, two-sided lower bound of $\Omega(k^{1/12})$ queries for this problem as well. Our bound is incomparable; it is asymptotically stronger and applies to adaptive algorithms, but only ones with one-sided error. The proof of our result relies on the following lemma.

Lemma 6.1. *For every $\alpha > 0$, there exist $k_0 \in \mathbb{N}$ and $\epsilon > 0$ such that for every $k \geq k_0$ and $k' \geq (1+\alpha)k$, all signed k'-majorities are ϵ-far from signed k-majorities.*

We defer the proof of Lemma 6.1 to Appendix B. We are now ready to complete the proof of Theorem 1.9.

Theorem 1.9 (Restated)**.** *Fix any constant $\gamma \in (0,1)$. For any $k \leq \gamma n$, testing signed k-majorities with one-sided error requires at least $\Omega(k/\log k)$ queries.*

Proof. We again use a reduction from the GAP-EQUALITY problem. Let $k' = \frac{k}{1-\gamma}$. Let ψ be a combining operator that takes functions $f, g : \{-1,1\}^n \to \{-k', -k'+1, \ldots, k'\}$ and returns the function h defined by $h(x) := \operatorname{sgn}\left(\frac{f(x)+g(x)}{2}\right)$. Define $\mathrm{C}_{\psi}^{k\text{-MAJ}}$ to be the communication game where Alice and Bob receive functions f, g and must test if $h = \psi(f,g)$ is a signed k-majority function. By Lemmas 2.4 and 2.7,

$$\log(2k'+1) \cdot Q^1\big(k\text{-MAJ}\big) \geq R^1(\mathrm{C}_{\psi}^{k\text{-MAJ}})$$

and

$$R^1(\mathrm{GEQ}_{k',\gamma k'}) = \Omega\big(k'\big) \ .$$

We complete the proof by showing that $R^1(\mathrm{C}_{\psi}^{k\text{-MAJ}}) \geq R^1(\mathrm{GEQ}_{k',\gamma k'})$. The theorem then follows by noting that $\Omega(\frac{k'}{\log k'}) = \Omega(\frac{k}{\log k})$.

Let $a, b \in \{0,1\}^{k'}$ be received by Alice and Bob, respectively, as the input to an instance of the $\mathrm{GEQ}_{k',\gamma k'}$ problem. Alice and Bob generate the functions $f, g : \{-1,1\}^n \to \{-k, -k+1, \ldots, k\}$ by setting $f(x) := \sum_{i=1}^{k'} (-1)^{a_i} x_i$ and $g(x) := \sum_{i=1}^{k'} (-1)^{b_i} x_i$, respectively. Note that $f(x) + g(x) = \sum_{i=1}^{k'} \left((-1)^{a_i} + (-1)^{b_i}\right) x_i$, so h can now be written as $h(x) = \operatorname{sgn}(w \cdot x)$, where

$$w_i = \begin{cases} 1 & \text{if } a_i = b_i = 0 \ , \\ 0 & \text{if } a_i \neq b_i \ , \\ -1 & \text{if } a_i = b_i = 1 \ . \end{cases}$$

When $a = b$, h is a signed k'-majority function. Lemma 6.1, shows that signed k'-majority functions are a constant distance from all signed k-majority functions. When $\Delta(a,b) = \gamma k'$, then $a_i = b_i$ for exactly k indices $i \in [k']$ and h is a signed k-majority function. So Alice and Bob can solve their instance of the GAP-EQUALITY problem with a protocol for the $\mathrm{C}_{\psi}^{k\text{-MAJ}}$ game and, as we wanted to show, $R^1(\mathrm{C}_{\psi}^{k\text{-MAJ}}) \geq R^1(\mathrm{GEQ}_{k',\gamma k'})$. $\square$

Acknowledgments

We thank Oded Goldreich for communicating the proof of Theorems 1.2 and 1.3 with us and for giving us permission to include these proofs. We also thank Sourav Chakraborty, David García Soriano, and Arie Matsliah for sharing an early manuscript version of (Chakraborty et al., 2011b) with us. In addition, E.B. wishes to thank Ryan O'Donnell for several helpful discussions during the course of this research.

We thank Amit Weinstein, Tom Gur, and the anonymous referees for insightful feedback on an earlier draft of this article. We offer special thanks to the referee who pointed out an error in our earlier version of Theorem 1.6 and to the referee who communicated the simplified proof of Lemma 2.10.

The final version of this article is published in pages 311–358 of Volume 21(2) of *Computational Complexity*, 2012. A preliminary version of this article appeared in pages 210–220 of the proceedings of the 26th Annual IEEE Conference on Computational Complexity, 2011.

Much of this work was performed while the second author was a postdoctoral researcher at Tsinghua University Institute for Interdisciplinary Information Sciences (IIIS). This work was supported in part by the National Basic Research Program of China Grant 2007CB807900, 2007CB807901, and the National Natural Science Foundation of China Grant 61033001, 61061130540, 61073174. The second author also acknowledges support from the Danish National Research Foundation and The National Science Foundation of China (under the grant 61061130540) for the Sino-Danish Center for the Theory of Interactive Computation and from the CFEM research center (supported by the Danish Strategic Research Council) within which part of this work was performed.

References

Noga Alon and Eric Blais. Testing boolean function isomorphism. In *Proc. 14th International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 394–405, 2010.

Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. In *Proc. 43rd Annual IEEE Symposium on Foundations of Computer Science*, pages 209–218, 2002.

Tugkan Batu, Ronitt Rubinfeld, and Patrick White. Fast approximate PCPs for multidimensional bin-packing problems. In *Proc. 3rd International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 245–256, 1999.

Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners of the hypercube and the hypergrid. Technical Report TR09-046, ECCC, 2009.

Eric Blais. Improved bounds for testing juntas. In *Proc. 12th International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 317–330, 2008.

Eric Blais. Testing juntas nearly optimally. In *Proc. 41st Annual ACM Symposium on the Theory of Computing*, pages 151–158, 2009.

Eric Blais and Daniel Kane. Tight bounds for testing k-linearity. In *Proc. 16th International Workshop on Randomization and Approximation Techniques in Computer Science*, 2012. (To Appear).

Eric Blais and Ryan O'Donnell. Lower bounds for testing function isomorphism. In *Proc. 25th Annual IEEE Conference on Computational Complexity*, pages 235–246, 2010.

Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. In *Proc. 26th Annual IEEE Conference on Computational Complexity*, 2011.

Eric Blais, Joshua Brody, and Kevin Matulef. Erratum to property testing lower bounds via communication complexity, 2012.

Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *J. Comput. Syst. Sci.*, 47:549–595, 1993. Earlier version in STOC'90.

Jop Briët, Sourav Chakraborty, David García Soriano, and Arie Matsliah. Monotonicity testing and shortest-path routing on the cube. In *Proc. 14th International Workshop on Randomization and Approximation Techniques in Computer Science*, 2010.

Joshua Brody, Amit Chakrabarti, Oded Regev, Thomas Vidick, and Ronald de Wolf. Better Gap-Hamming lower bounds via better round elimination. In *Proc. 14th International Workshop on Randomization and Approximation Techniques in Computer Science*, 2010.

Joshua Brody, Kevin Matulef, and Chenggang Wu. Lower bounds for testing computability by small-width branching programs. In *Proc. 8th Annual Theory and Applications of Models of Computation*, 2011.

Harry Buhrman, Richard Cleve, and Avi Wigderson. Quantum vs. classical communication and computation. In *Proc. 30th Annual ACM Symposium on the Theory of Computing*, pages 63–68, 1998.

Amit Chakrabarti and Oded Regev. An optimal lower bound on the communication complexity of Gap-Hamming-Distance. In *Proc. 43rd Annual ACM Symposium on the Theory of Computing*, 2011.

Sourav Chakraborty, David García Soriano, and Arie Matsliah. Nearly tight bounds for testing function isomorphism. In *Proc. 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, 2011a.

Sourav Chakraborty, David García Soriano, and Arie Matsliah. Efficient sample extractors for juntas with applications. In *Proc. 38th International Colloquium on Automata, Languages and Programming*, 2011b.

Hana Chockler and Dan Gutfreund. A lower bound for testing juntas. *Information Processing Letters*, 90 (6):301–305, 2004.

Ronald de Wolf. A brief introduction to fourier analysis on the boolean cube. *Theory of Computing, Graduate Surveys*, 1:1–20, 2008.

Ilias Diakonikolas, Homin Lee, Kevin Matulef, Krzysztof Onak, Ronitt Rubinfeld, Rocco Servedio, and Andrew Wan. Testing for concise representations. In *Proc. 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 549–558, 2007.

Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved testing algorithms for monotonicity. In *Proc. 3rd International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 97–108, 1999.

Funda Ergun, Sampath Kannan, Ravi Kumar, Ronitt Rubenfeld, and Mahesh Viswanathan. Spot-checkers. *J. Comput. Syst. Sci.*, 60:717–751, 2000.

W. Feller. *An introduction to probability theory and its applications*, volume 2. John Wiley & Sons, 1968.

Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In *Proc. 34th Annual ACM Symposium on the Theory of Computing*, pages 474–483, 2002.

Eldar Fischer, Guy Kindler, Dana Ron, Shmuel Safra, and Alex Samorodnitsky. Testing juntas. *J. Comput. Syst. Sci.*, 68:753–787, 2004.

Oded Goldreich, editor. *Property Testing: Current Research and Surveys*, volume 6390 of *LNCS*. Springer, 2010a.

Oded Goldreich. On testing computability by small width OBDDs. In *Proc. 14th International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 574–587, 2010b.

Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.

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

Johan Håstad and Avi Wigderson. The randomized communication complexity of set disjointness. *Theory of Computing*, pages 211–219, 2007.

Piotr Indyk and David Woodruff. Tight lower bounds for the distinct elements problem. In *Proc. 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 283–289, 2003.

Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM J. Disc. Math.*, 5(4):547–557, 1992.

Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, Cambridge, 1997.

Kevin Matulef, Ryan O'Donnell, Ronitt Rubinfeld, and Rocco Servedio. Testing {-1,1}-weight halfspaces. In *Proc. 13th International Workshop on Randomization and Approximation Techniques in Computer Science*, 2009.

Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On data structures and asymmetric communication complexity. In *Proc. 27th Annual ACM Symposium on the Theory of Computing*, pages 103–111, 1995.

Ryan O'Donnell. Some topics in analysis of boolean function. In *Proc. 40th Annual ACM Symposium on the Theory of Computing*, pages 569–578, 2008.

Michal Parnas, Dana Ron, and Alex Samorodnitsky. Testing basic boolean formulae. *SIAM J. Disc. Math.*, 16(1):20–46, 2002.

Michal Parnas, Dana Ron, and Ronitt Rubinfeld. On testing convexity and submodularity. *SIAM J. Comput.*, 32(5):1158–1184, 2003.

Alexander Razborov. On the distributional complexity of disjointness. In *Proc. 17thInternational Colloquium on Automata, Languages and Programming*, pages 249–253, 1990.

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

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

Dana Ron and Gilad Tsur. Testing computability by width two obdds. In *Proc. 13th International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 686–699, 2009.

Dana Ron and Gilad Tsur. On approximating the number of relevant variables in a function. In *Proc. 15thInternational Workshop on Randomization and Approximation Techniques in Computer Science*, pages 676–687, 2011.

Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25:252–271, 1996.

C. Seshadhri and Jan Vondrák. Is submodularity testable? In *Proc. 2nd Innovations in Computer Science*, 2011.

A Distance from sparse polynomials

In this section we provide a self-contained proof that $(k+2)$-linear functions are far from k-sparse polynomials. This lemma is a special case of Diakonikolas et al. (2007, Theorem 36). By considering only a special case of the theorem of Diakonikolas et al. (2007), we obtain a slightly stronger bound on the distance of $(k+2)$-linear functions to k-sparse polynomials but the proof itself is essentially identical. We include the proof here primarily for completeness.

Lemma A.1 (Diakonikolas et al. 2007)**.** *Every $(k+2)$-linear function is $\frac{1}{20}$-far from a k-sparse polynomial over* GF(2)*.*

Proof. Let f be a $(k+2)$-linear function, and without loss of generality assume f is a linear function on the first $k+2$ variables, i.e. $f(x) = x_1 \oplus \cdots \oplus x_{k+2}$. Let g be a k-sparse polynomial, i.e. $g = T_1 \oplus \cdots \oplus T_k$ where each T_i is a monomial. We want to show that f and g are far. We can assume without loss of generality that g does not contain any length-1 terms, since if it did we could just subtract those terms off of both f and g to create f' and g', which have the same distance from each other. We could then prove the theorem for f', g', and a smaller value of k.

Define the *influence* of a variable x_i in f, denoted $\mathrm{Inf}_i(f)$, in the standard way- i.e. $\mathrm{Inf}_i(f) = \Pr_x[f(x) \neq f(x^{\oplus i})]$ where $x^{\oplus i}$ denotes x with the ith bit flipped. Define the *total influence* of f to be $\sum_i \mathrm{Inf}_i(f)$.

For any f and g, it is straightforward to show that if for some i the difference $|\mathrm{Inf}_i(f) - \mathrm{Inf}_i(g)|$ is at least δ, then f and g must have distance at least $\delta/2$. When f is the $(k+2)$-linear function defined above, each variable x_1 through x_{k+2} has influence 1. Thus, to complete the proof, we will show that in g one of these variables must have influence at most 0.9.

If the total influence of x_1 through x_{k+2} in g is less than $0.9(k+2)$, then we are done, since the pigeonhole principle implies the existence of a variable x_i with influence at most 0.9. Thus, in what follows, we assume

$$\sum_{i}^{k+2} \mathrm{Inf}_i(g) \geq 0.9(k+2) \ . \tag{2}$$

We can bound the total influence of x_1 through x_{k+2} in g as follows. First, we write $g = g_2 \oplus g_3$ where g_2 is the collection of terms in g that have length 2, and g_3 is the collection of terms in g that have length at least 3. Now note:

- Each variable x_i that appears in g_2 has $\mathrm{Inf}_i(g_2) = 1/2$. The reason is because since every term of g_2 has length 2, x_i is influential exactly when the other variables it appears with have parity 1, which happens exactly half the time.
- For each term in g_3, the total contribution of that term to the influences of all the variables is at most 3/4. To see why, suppose the term has length m, then on a random assignment the probability that a variable is relevant to that term is $\frac{1}{2^{m-1}}$, so the total effect the term can have on all the influences is at most $m \cdot \frac{1}{2^{m-1}}$. If $m \geq 3$, this is at most 3/4.

Let R_2 be the number of terms of g_2, and R_3 be the number of terms in g_3. By hypothesis, $R_2 + R_3 \leq k$. Since each term of g_2 contributes at most 1 to the total influence of g, and each term of g_3 contributes at most 3/4 to the total influence of g, we have that

$$\sum_{i}^{k+2} \mathrm{Inf}_i(g) \leq R_2 + (3/4)R_3 \ . \tag{3}$$

Combining equations (2) and (3) we get that $R_2 + (3/4)R_3 \geq (9/10)k$. Using the fact that $R_2 + R_3 \leq k$, this implies that $R_3 \leq (4/10)k$, in other words there cannot be too many terms of length 3 or more in g.

Now we can bound the influence of variables x_1 through x_{k+2} in g.

$$\begin{aligned}\sum_i^{k+2} \mathrm{Inf}_i(g) &\le \sum_i^{k+2} [\mathrm{Inf}_i(g_2) + \mathrm{Inf}_i(g_3)] \\ &\le \sum_i^{k+2} \mathrm{Inf}_i(g_2) + \sum_i^{n} \mathrm{Inf}_i(g_3) \\ &\le \frac{1}{2}(k+2) + \frac{3}{4} \cdot R_3 \\ &\le \frac{1}{2}(k+2) + \frac{3}{4} \cdot \frac{4}{10} \cdot k \\ &< 0.9(k+2)\ .\end{aligned}$$

By the pigeonhole principle, there must exist a variable x_i with influence at most 0.9 in g. □

B Distance between majority functions

We complete the proof of Lemma 6.1 in this section. A key ingredient in this proof is the Berry-Esseen theorem, a version of the Central Limit Theorem with error bounds (see e.g. Feller 1968):

Theorem B.1 (Berry-Esseen)**.** *Let $\ell(x) = c_1 x_1 + \cdots + c_n x_n$ be a linear form over the random ± 1 bits x_i. Assume $|c_i| \le \tau$ for all i and write $\sigma = \sqrt{\sum c_i^2}$. Write F for the c.d.f. of $\ell(x)/\sigma$; i.e., $F(t) = \Pr[\ell(x)/\sigma \le t]$. Then for all $t \in \mathbb{R}$,*

$$|F(t) - \Phi(t)| \le O(\tau/\sigma) \cdot \frac{1}{1+|t|^3},$$

where Φ denotes the c.d.f. of X, a standard Gaussian random variable. In particular, if $A \subseteq \mathbb{R}$ is any interval then $|\Pr[\ell(x)/\sigma \in A] - \Pr[X \in A]| \le C_1(\tau/\sigma)$, where C_1 is an absolute constant.

Lemma 6.1 (Restated)**.** *For every $\alpha > 0$, there exist $k_0 \in \mathbb{N}$ and $\epsilon > 0$ such that for every $k \ge k_0$ and $k' \ge (1+\alpha)k$, all signed k'-majorities are ϵ-far from signed k-majorities.*

Proof. Let f be a signed k-majority, and g be a signed k'-majority. It is easy to see that f and g have minimum distance when they have the same sign pattern on their common variables. So without loss of generality, assume $f(x) = \mathrm{sgn}(x_1 + \cdots + x_k)$ and $g(x) = \mathrm{sgn}(x_1 + \cdots + x_{k'})$ (in other words, f is a majority function on the first k variables, and g is a majority function on the first k' variables). To simplify, we will write $S(x) = \sum_{i=1}^{k} x_i$ and $T(x) = \sum_{i=k+1}^{k'} x_i$. Thus, $f(x) = \mathrm{sgn}(S(x))$ and $g(x) = \mathrm{sgn}(S(x) + T(x))$.

For any positive real number t, we have

$$\begin{aligned}\Pr_x[f(x) \ne g(x)] &\ge \Pr_x[S(x) \in [0,t) \text{ and } T(x) < -t] \\ &= \Pr_x[S(x) \in [0,t)] \cdot \Pr_x[T(x) < -t]\ ,\end{aligned}$$

where the equality follows from the fact that S and T are functions on disjoint sets of variables.

Note that S is a linear form on k variables, so we can use the Berry-Esseen theorem on S with $\sigma = \sqrt{k}$ to get

$$\begin{aligned}\Pr_x[S(x) \in [0,t)] &\ge (\Phi(t/\sqrt{k}) - \Phi(0)) - C_1/\sqrt{k} \\ &\ge (\Phi(t/\sqrt{k}) - 1/2) - C_1/\sqrt{k}\ , \qquad (4)\end{aligned}$$

where C_1 is the constant from the Berry-Esseen theorem.

Similarly, T is a linear form on αk variables, so we can use the Berry-Esseen theorem on T with $\sigma = \sqrt{\alpha k}$ to get

$$\Pr_x[T(x) < -t] \geq \Phi(-t/\sqrt{\alpha k}) - C_1/\sqrt{\alpha k} \ . \tag{5}$$

Setting t to be, say, $\sqrt{k}$, and then choosing k large enough insures that the quantities in both (4) and (5) are positive, and bigger than a constant which only depends on α. □