An Optimal Lower Bound for Monotonicity Testing over Hypergrids

Deeparnab Chakrabarty[1] and C. Seshadhri[2]

[1] Microsoft Research India
dechakr@microsoft.com
[2] Sandia National Labs, Livermore[⋆]
scomand@sandia.gov

Abstract. For positive integers n, d, consider the hypergrid $[n]^d$ with the coordinate-wise product partial ordering denoted by $\prec$. A function $f : [n]^d \to \mathbb{N}$ is monotone if $\forall x \prec y$, $f(x) \le f(y)$. A function f is ε-far from monotone if at least an ε-fraction of values must be changed to make f monotone. Given a parameter ε, a *monotonicity tester* must distinguish with high probability a monotone function from one that is ε-far.
We prove that any (adaptive, two-sided) monotonicity tester for functions $f : [n]^d \to \mathbb{N}$ must make $\Omega(\varepsilon^{-1} d \log n - \varepsilon^{-1} \log \varepsilon^{-1})$ queries. Recent upper bounds show the existence of $O(\varepsilon^{-1} d \log n)$ query monotonicity testers for hypergrids. This closes the question of monotonicity testing for hypergrids over arbitrary ranges. The previous best lower bound for general hypergrids was a non-adaptive bound of $\Omega(d \log n)$.

Keywords: Monotonicity testing, sublinear algorithms, lower bounds

1 Introduction

Given query access to a function $f : \mathbf{D} \to \mathbf{R}$, the field of *property testing* [1,2] deals with the problem of determining properties of f without reading all of it. Monotonicity testing [3] is a classic problem in property testing. Consider a function $f : \mathbf{D} \to \mathbf{R}$, where $\mathbf{D}$ is some partial order given by "$\prec$", and $\mathbf{R}$ is a total order. The function f is monotone if for all $x \prec y$ (in $\mathbf{D}$), $f(x) \le f(y)$. The *distance to monotonicity* of f is the minimum fraction of values that need to be modified to make f monotone. More precisely, define the distance between functions $d(f, g)$ as $|\{x : f(x) \ne g(x)\}|/|\mathbf{D}|$. Let $\mathcal{M}$ be the set of all monotone functions. Then the distance to monotonicity of f is $\min_{g \in \mathcal{M}} d(f, g)$.

A function is called ε-far from monotone if the distance to monotonicity is at least ε. A *property tester for monotonicity* is a, possibly randomized, algorithm that takes as input a distance parameter $\varepsilon \in (0, 1)$, error parameter $\delta \in [0, 1]$,

[⋆] Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

and query access to an arbitrary f. If f is monotone, then the tester must accept with probability $> 1-\delta$. If it is ε-far from monotone, then the tester rejects with probability $> 1-\delta$. (If neither, then the tester is allowed to do anything.) The aim is to design a property tester using as few queries as possible. A tester is called *one-sided* if it always accepts a monotone function. A tester is called *non-adaptive* if the queries made do not depend on the function values. The most general tester is an adaptive two-sided tester.

Monotonicity testing has a rich history and the hypergrid domain, $[n]^d$, has received special attention. The boolean hypercube ($n = 2$) and the total order ($d = 1$) are special instances of hypergrids. Following a long line of work [4,3,5,6,7,8,9,10,11,12,13,14], previous work of the authors [15] shows the existence of $O(\varepsilon^{-1} d \log n)$-query monotonicity testers. Our result is a matching adaptive lower bound that is optimal in all parameters (for unbounded range functions). This closes the question of monotonicity testing for unbounded ranges on hypergrids. This is also the first adaptive bound for monotonicity testing on general hypergrids.

Theorem 1. *Any (adaptive, two-sided) monotonicity tester for functions* $f : [n]^d \to \mathbb{N}$ *requires* $\Omega(\varepsilon^{-1} d \log n - \varepsilon^{-1} \log \varepsilon^{-1})$ *queries.*

1.1 Previous work

The problem of monotonicity testing was introduced by Goldreich et al. [3], with an $O(n/\varepsilon)$ tester for functions $f : \{0,1\}^n \to \{0,1\}$. The first tester for general hypergrids was given by Dodis et al. [5]. The upper bound of $O(\varepsilon^{-1} d \log n)$ for monotonicity testing was recently proven in [15]. We refer the interested reader to the introduction of [15] for a more detailed history of previous upper bounds.

There have been numerous lower bounds for monotonicity testing. We begin by summarizing the state of the art. The known adaptive lower bounds are $\Omega(\log n)$ for the total order $[n]$ by Fischer [9], and $\Omega(d/\varepsilon)$ for the boolean hypercube $\{0,1\}^d$ by Brody [16]. For general hypergrids, Blais, Raskhodnikova, and Yaroslavtsev [17] recently proved the first result, a non-adaptive lower bound of $\Omega(d \log n)$. Theorem 1 is the first adaptive bound for monotonicity testing on hypergrids and is optimal (for arbitrary ranges) in all parameters.

Now for the chronological documentation. The first lower bound was the non-adaptive bound of $\Omega(\log n)$ for the total order $[n]$ by Ergun et al. [4]. This was extended by Fischer [9] to an (optimal) adaptive bound. For the hypercube domain $\{0,1\}^d$, Fischer et al. [7] proved the first non-adaptive lower bound of $\Omega(\sqrt{d})$. (This was proven even for the range $\{0,1\}$.) This was improved to $\Omega(d/\varepsilon)$ by Brïet et al. [18]. Blais, Brody, and Matulef [14] gave an ingenious reduction from communication complexity to prove an adaptive, two-sided bound of $\Omega(d)$. (Honing this reduction, Brody [16] improved this bound to $\Omega(d/\varepsilon)$.) The non-adaptive lower bounds of Blais, Raskhodnikova, and Yaroslavtsev [17] were also achieved through communication complexity reductions.

We note that our theorem only holds when the range is $\mathbb{N}$, while some previous results hold for restricted ranges. The results of [14,16] provide lower bounds for range $[\sqrt{d}]$. The non-adaptive bound of [17] holds even when the range is $[nd]$. In that sense, the communication complexity reductions provide stronger lower bounds than our result.

1.2 Main ideas

The starting point of this work is the result of Fischer [9], an adaptive lower bound for monotonicity testing for functions $f : [n] \to \mathbb{N}$. He shows that adaptive testers can be converted to comparison-based testers, using Ramsey theory arguments. A comparison-based tester for $[n]$ can be easily converted to a non-adaptive tester, for which an $\Omega(\log n)$ bound was previously known. We make a fairly simple observation. The main part of Fischer's proof actually goes through for functions over *any partial order*, so it suffices to prove lower bounds for comparison-based testers. (The reduction to non-adaptive testers only holds for $[n]$.)

We then prove a comparison-based lower bound of $\Omega(\varepsilon^{-1} d \log n - \varepsilon^{-1} \log \varepsilon^{-1})$ for the domain $[n]^d$. As usual, Yao's minimax lemma allows us to focus on determinstic lower bounds over some distribution of functions. The major challenge in proving (even non-adaptive) lower bounds for monotonicity is that the tester might make decisions based on the actual values that it sees. Great care is required to construct a distribution over functions whose monotonicity status cannot be decided by simply looking at the values. But a comparison-based tester has no such power, and optimal lower bounds over all parameters can be obtained with a fairly clean distribution.

2 The reduction to comparison based testers

Consider the family of functions $f : \mathbf{D} \to \mathbf{R}$, where $\mathbf{D}$ is some partial order, and $\mathbf{R} \subseteq \mathbb{N}$. We will assume that f always takes distinct values, so $\forall x, y, f(x) \neq f(y)$. Since we are proving lower bounds, this is no loss of generality.

Definition 1. *An algorithm $\mathcal{A}$ is a (t, ε, δ)-*monotonicity tester *if $\mathcal{A}$ has the following properties. For any $f : \mathbf{D} \to \mathbf{R}$, the algorithm $\mathcal{A}$ makes t (possibly randomized) queries to f and then outputs either "accept" or "reject". If f is monotone, then $\mathcal{A}$ accepts with probability $> 1 - \delta$. If f is ε-far from monotone, then $\mathcal{A}$ rejects with probability $> 1 - \delta$.*

Given a positive integer s, let $\mathbf{D}^s$ denote the collection of *ordered*, s-tupled vectors with each entry in $\mathbf{D}$. We define two symbols `acc` and `rej`, and denote $\mathbf{D}' = \mathbf{D} \cup \{\texttt{acc}, \texttt{rej}\}$. Any (t, ε, δ)-tester can be completely specified by the following family of functions. For all $s \leq t$, $\mathbf{x} \in \mathbf{D}^s$, $y \in \mathbf{D}'$, we consider a function $p^y_{\mathbf{x}} : \mathbf{R}^s \to [0, 1]$, with the semantic that for any $\mathbf{a} \in \mathbf{R}^s$, $p^y_{\mathbf{x}}(\mathbf{a})$ denotes the probability the tester queries y as the $(s+1)$th query, given that the first s queries are $\mathbf{x}_1, \ldots, \mathbf{x}_s$ and $f(\mathbf{x}_i) = \mathbf{a}_i$ for $1 \leq i \leq s$. By querying `acc`, `rej` we imply returning accept or reject. These functions satisfy the following properties.

$$\forall s \le t,\ \forall \mathbf{x} \in \mathbf{D}^s,\ \forall \mathbf{a} \in \mathbf{R}^s,\ \sum_{y \in \mathbf{D}'} p^y_{\mathbf{x}}(\mathbf{a}) = 1 \tag{1}$$

$$\forall \mathbf{x} \in \mathbf{D}^t,\ \forall y \in \mathbf{D},\ \forall \mathbf{a} \in \mathbf{R}^t,\ p^y_{\mathbf{x}}(\mathbf{a}) = 0 \tag{2}$$

Eq. (1) ensures the decisions of the tester at step $(s+1)$ must form a probability distribution. Eq. (2) implies that the tester makes at most t queries.

For any positive integer s, let $\mathbf{R}^{(s)}$ denote the set of *unordered* subets of $\mathbf{R}$ of cardinality s. For reasons that will soon become clear, we introduce new functions as follows. For each s, $\mathbf{x} \in \mathbf{D}^s$, $y \in \mathbf{D}'$, and *each permutation* $\sigma : [s] \to [s]$, we associate functions $q^y_{\mathbf{x},\sigma} : \mathbf{R}^{(s)} \to [0,1]$, with the semantic

$$\text{For any set } S = (a_1 < a_2 < \cdots < a_s) \in \mathbf{R}^{(s)}, \qquad q^y_{\mathbf{x},\sigma}(S) := p^y_{\mathbf{x}}(a_{\sigma(1)}, \ldots, a_{\sigma(s)})$$

That is, $q^y_{\mathbf{x},\sigma_s}(S)$ sorts the answers in S in increasing order, permutes them according to σ, and passes the permuted ordered tuple to $p^y_{\mathbf{x}}$. Any adaptive tester can be specified by these functions. The important point to note is that they are finitely many such functions; their number is upper bounded by $(t|\mathbf{D}|)^{t+1}$. These q-functions allow us to define comparison based testers.

Definition 2. *A monotonicity tester $\mathcal{A}$ is* comparison-based *if for all s,$\mathbf{x} \in \mathbf{D}^s, y \in \mathbf{D}'$, and permutations $\sigma : [s] \to [s]$, the function $q^y_{\mathbf{x},\sigma}$ is a constant function on $\mathbf{R}^{(s)}$. In other words, the $(s+1)$th decision of the tester given that the first s questions is $\mathbf{x}$, depends only on the* ordering *of the answers received, and not on the values of the answers.*

The following theorem is implicit in the work of Fischer [9].

Theorem 2. *Suppose there exists a (t, ε, δ)-monotonicity tester for functions $f : \mathbf{D} \to \mathbb{N}$. Then there exists a comparison-based $(t, \varepsilon, 2\delta)$-monotonicity tester for functions $f : \mathbf{D} \to \mathbb{N}$.*

This implies that a comparison-based lower bound suffices for proving a general lower bound on monotonicity testing. We provide a proof of the above theorem in the next section for completeness.

2.1 Performing the reduction

We basically present Fischer's argument, observing that $\mathbf{D}$ can be any partial order. A monotonicity tester is called *discrete* if the corresponding functions $p^y_{\mathbf{x}}$ can only take values in $\{i/K \ : \ 0 \le i \le K\}$ for some finite K. Note that this implies the functions $q^y_{\mathbf{x},\sigma}$ also take discrete values.

Lemma 1. *Suppose there exists a (t, ε, δ)-monotonicity tester $\mathcal{A}$ for functions $f : \mathbf{D} \to \mathbb{N}$. Then there exists a discrete $(t, \varepsilon, 2\delta)$-monotonicity tester for these functions.*

Proof. We do a rounding on the p-functions. Let $K = 100t|\mathbf{D}|^t/\delta^2$. Start with the p-functions of the (t, ε, δ)-tester $\mathcal{A}$. For $y \in \mathbf{D} \cup \texttt{acc}$, $\mathbf{x} \in \mathbf{D}^s$, $\mathbf{a} \in \mathbf{R}^s$, let $\hat{p}^y_{\mathbf{x}}(\mathbf{a})$ be the largest value in $\{i/K \mid 0 \le i \le K\}$ at most $p^y_{\mathbf{x}}(\mathbf{a})$. Set $\hat{p}^{\texttt{rej}}_{\mathbf{x}}(\mathbf{a})$ so that Eq. (1) is maintained.

Note that for $y \in \mathbf{D} \cup \texttt{acc}$, if $p^y_{\mathbf{x}}(\mathbf{a}) > 10t/(\delta K)$, then

$$\left(1 - \frac{\delta}{10t}\right) p^y_{\mathbf{x}}(\mathbf{a}) \ \le\ \hat{p}^y_{\mathbf{x}}(\mathbf{a}) \ \le\ p^y_{\mathbf{x}}(\mathbf{a}).$$

Furthermore, $\hat{p}^{\texttt{rej}}_{\mathbf{x}}(\mathbf{a}) \ge p^{\texttt{rej}}_{\mathbf{x}}(\mathbf{a})$.

The $\hat{p}$-functions describe a new discrete tester $\mathcal{A}'$ that makes at most t queries. We argue that $\mathcal{A}'$ is a $(t, \varepsilon, 2\delta)$-tester. Given a function f that is either monotone or ε-far from monotone, consider a sequence of queries $x_1, \ldots, x_s$ after which $\mathcal{A}$ returns a *correct* decision $\aleph$. Call such a sequence good, and let α denote the probability this occurs. We know that the sum of probabilities over all good query sequences is at least $(1 - \delta)$. Now,

$$\alpha := p^{x_1} \cdot\ p^{x_2}_{x_1}(f(x_1)) \cdot\ p^{x_3}_{(x_1,x_2)}(f(x_1), f(x_2)) \cdots\ p^{\aleph}_{(x_1,\ldots,x_s)}(f(x_1), \ldots, f(x_s))$$

Two cases arise. Suppose all of the probabilities in the RHS are $\ge 10t/\delta K$. Then, the probability of this good sequence arising in $\mathcal{A}'$ is at least $(1-\delta/10t)^t\alpha \ge \alpha(1-\delta/2)$. Otherwise, suppose some probability in the RHS is $< 10t/\delta K$. Then the total probability mass on such good sequences in $\mathcal{A}$ is at most $10t/\delta K \cdot |\mathbf{D}|^t \le \delta/2$. Therefore, the probability of good sequences in $\mathcal{A}'$ is at least $(1 - 3\delta/2)(1 - \delta/2) \ge 1 - 2\delta$. That is, $\mathcal{A}'$ is a $(t, \varepsilon, 2\delta)$ tester.

We introduce some Ramsey theory terminology. For any positive integer i, a *finite* coloring of $\mathbb{N}^{(i)}$ is a function $\texttt{col}_i : \mathbb{N}^{(i)} \to \{1, \ldots, C\}$ for some finite number C. An infinite set $X \subseteq \mathbb{N}$ is called *monochromatic* w.r.t $\texttt{col}_i$ if for all sets $A, B \in X^{(i)}$, $\texttt{col}_i(A) = \texttt{col}_i(B)$. A *$k$-wise* finite coloring of $\mathbb{N}$ is a collection of k colorings $\texttt{col}_1, \ldots, \texttt{col}_k$. (Note that each coloring is over different sized tuples.) An infinite set $X \subseteq \mathbb{N}$ is k-wise monochromatic if X is monochromatic w.r.t. all the $\texttt{col}_i$s.

The following is a simple variant of Ramsey's original theorem. (We closely follow the proof of Ramsey's theorem as given in Chap VI, Theorem 4 of [19].)

Theorem 3. *For any k-wise finite coloring of $\mathbb{N}$, there is an infinite k-wise monochromatic set $X \subseteq \mathbb{N}$.*

Proof. We proceed by induction on k. If $k = 1$, then this is trivially true; let X be the maximum color class. Since the coloring is finite, X is infinite. We will now iteratively construct an infinite set of $\mathbb{N}$ via induction.

Start with a_0 being the minimum element in $\mathbb{N}$. Consider a $(k-1)$-wise coloring of $(\mathbb{N} \setminus \{a_0\})$ $\texttt{col}'_1, \ldots, \texttt{col}'_{k-1}$, where $\texttt{col}'_i(S) := \texttt{col}_{i+1}(S \cup a_0)$. By the induction hypothesis, there exists an infinite $(k-1)$-wise monochromatic set $A_0 \subseteq \mathbb{N} \setminus \{a_0\}$ with respect to coloring $\texttt{col}'_i$s. That is, for $1 \le i \le k$, and any set $S, T \subseteq A_0$ with $|S| = |T| = i - 1$, we have $\texttt{col}_i(a_0 \cup S) = \texttt{col}_i(a_0 \cup T) = C^0_i$, say. Denote the collection of these colors as a vector $\mathbf{C}_0 = (C^0_1, C^0_2, \ldots, C^0_k)$.

Subsequently, let a_1 be the minimum element in A_0, and consider the $(k-1)$-wise coloring $\texttt{col}'$ of $(A_0 \setminus \{a_1\})$ where $\texttt{col}'_i(S) = \texttt{col}_{i+1}(S \cup \{a_1\})$ for $S \subseteq A_0 \setminus \{a_1\}$. Again, the induction hypothesis yields an infinite $(k-1)$-wise monochromatic set A_1 as before, and similarly the vector $\mathbf{C}_1$. Continuing this procedure, we get an infinite sequence $a_0, a_1, a_2, \ldots$ of natural numbers, an infinite sequence of vectors of k colors $\mathbf{C}_0, \mathbf{C}_1, \ldots$, and an infinite nested sequence of infinite sets $A_0 \supset A_1 \supset A_2 \ldots$. Every A_r contains $a_s, \forall s > r$ and by construction, any set $(\{a_r\} \cup S)$, $S \subseteq A_r$, $|S| = i-1$, has color C^i_r. Since there are only finitely many colors, some vector of colors occurs infinitely often as $\mathbf{C}_{r_1}, \mathbf{C}_{r_2}, \ldots$. The corresponding infinite sequence of elements $a_{r_1}, a_{r_2}, \ldots$ is k-wise monochromatic.

Proof. (of Theorem 2) Suppose there exists a (t, ε, δ)-tester for functions $f : \mathbf{D} \to \mathbb{N}$. We need to show there is a comparison-based $(t, \varepsilon, 2\delta)$-tester for such functions.

By Lemma 1, there is a discrete $(t, \varepsilon, 2\delta)$-tester $\mathcal{A}$. Equivalently, we have the functions $q^y_{\mathbf{x},\sigma}$ as described in the previous section. We now describe a t-wise finite coloring of $\mathbb{N}$. Consider $s \in [t]$. Given a set $A \subseteq \mathbb{N}^{(s)}$, $\texttt{col}_s(A)$ is a vector indexed by $(y, \mathbf{x}, \sigma)$, where $y \in D'$, $\mathbf{x} \in D^s$, and σ is a s-permutation, whose entry is $q^y_{\mathbf{x},\sigma}(A)$. The domain is finite, so the number of dimensions is finite. Since the tester is discrete, the number of possible colors entries is finite. Applying Theorem 3, we know the existence of a t-wise monochromatic infinite set $\mathbf{R} \subseteq \mathbb{N}$. We have the property that for any $y, \mathbf{x}, \sigma$, and any two sets $A, B \in \mathbf{R}^{(s)}$, we have $q^y_{\mathbf{x},\sigma}(A) = q^y_{\mathbf{x},\sigma}(B)$. That is, the algorithm $\mathcal{A}$ is a comparison based tester for functions with range $\mathbf{R}$.

Consider the strictly monotone map $\phi : \mathbb{N} \to \mathbf{R}$, where $\phi(b)$ is the bth element of $\mathbf{R}$ in sorted order. Now given any function $f : \mathbf{D} \to \mathbb{N}$, consider the function $\phi \circ f : \mathbf{D} \to \mathbf{R}$. Consider an algorithm $\mathcal{A}'$ which on input f runs $\mathcal{A}$ on $\phi \circ f$. More precisely, whenever $\mathcal{A}$ queries a point x, it gets answer $\phi \circ f(x)$. Observe that if f is monotone (or ε-far from monotone), then so is $\phi \circ f$, and therefore, the algorithm $\mathcal{A}'$ is a $(t, \varepsilon, 2\delta)$-tester of $\phi \circ f$. Since the range of $\phi \circ f$ is $\mathbf{R}$, $\mathcal{A}'$ is comparison-based.

3 Lower bounds

We assume that n is a power of 2, set $\ell := \log_2 n$, and think of $[n]$ as $\{0, 1, \ldots, n-1\}$. For any number $0 \le z < n$, we think of the binary representation of z as an ℓ-bit vector $(z_1, z_2, \ldots, z_\ell)$, where z_1 is the least significant bit.

Consider the following canonical, one-to-one mapping $\phi : [n]^d \to \{0,1\}^{d\ell}$. For any $\boldsymbol{y} = (y_1, y_2, \ldots, y_d) \in [n]^d$, we concatenate binary representations of the y_is in order to get a $d\ell$-bit vector $\phi(\boldsymbol{y})$. Hence, we can transform a function $f : \{0,1\}^{d\ell} \to \mathbb{N}$ into a function $\widetilde{f} : [n]^d \to \mathbb{N}$ by defining $\widetilde{f}(\boldsymbol{y}) := f(\phi(\boldsymbol{y}))$.

We will now describe a distribution of functions over the boolean hypercube with equal mass on monotone and ε-far from monotone functions. The key property is that for a function drawn from this distribution, any deterministic comparison based algorithm errs in classifying it with non-trivial probability.

This property will be used in conjunction with the above mapping to get our final lower bound.

3.1 The hard distribution

We focus on functions $f : \{0,1\}^m \to \mathbb{N}$. (Eventually, we set $m = d\ell$.) Given any $x \in \{0,1\}^m$, we let $\texttt{val}(x) := \sum_{i=1}^{m} 2^{i-1} x_i$ denote the number for which x is the binary representation. Here, x_1 denotes the least significant bit of x.

For convenience, we let ε be a power of 1/2. For $k \in \{1, \ldots, \frac{1}{2\varepsilon}\}$, we let

$$S_k := \{x : \texttt{val}(x) \in [2(k-1)\varepsilon 2^m, 2k\varepsilon 2^m - 1) \ \ \}.$$

Note that S_ks partition the hypercube, with each $|S_k| = \varepsilon 2^{m+1}$. In fact, each S_k is a subhypercube of dimension $m' := m + 1 - \log(1/\varepsilon)$, with the minimal element having all zeros in the m' least significant bits, and the maximal element having all ones in those.

We describe a distribution $\mathcal{F}_{m,\varepsilon}$ on functions. The support of $\mathcal{F}_{m,\varepsilon}$ consists of $f(x) = 2\texttt{val}(x)$ and $\frac{m'}{2\varepsilon}$ functions indexed as $g_{j,k}$ with $j \in [m']$ and $k \in [\frac{1}{2\varepsilon}]$, defined as follows.

$$g_{j,k}(x) = \begin{cases} 2\texttt{val}(x) - 2^j - 1 & \text{if } x_j = 1 \text{ and } x \in S_k \\ 2\texttt{val}(x) & \text{otherwise} \end{cases}$$

The distribution $\mathcal{F}_{m,\varepsilon}$ puts probability mass 1/2 on the function $f = 2\texttt{val}$ and $\frac{\varepsilon}{m'}$ on each of the $g_{j,k}$s. All these functions take distinct values on their domain. Note that $2\texttt{val}$ induces a total order on $\{0,1\}^m$.

The distinguishing problem: Given query access to a random function f from $\mathcal{F}_{m,\varepsilon}$, we want a deterministic comparison-based algorithm that declares that $f = 2\texttt{val}$ or $f \neq 2\texttt{val}$. We refer to any such algorithm as a *distinguisher*. Naturally, we say that the distinguisher errs on f if its declaration is wrong. Our main lemma is the following.

Lemma 2. *Any deterministic comparison-based distinguisher that makes less than $\frac{m'}{8\varepsilon}$ queries errs with probability at least* 1/8.

The following proposition allows us to focus on *non-adaptive* comparison based testers.

Proposition 1. *Given any deterministic comparison-based distinguisher $\mathcal{A}$ for $\mathcal{F}_{m,\varepsilon}$ that makes at most t queries, there exists a deterministic* non-adaptive *comparison-based distinguisher $\mathcal{A}'$ making at most t queries whose probability of error on $\mathcal{F}_{m,\varepsilon}$ is at most that of $\mathcal{A}$.*

Proof. We represent $\mathcal{A}$ as a comparison tree. For any path in $\mathcal{A}$, the total number of distinct domain points involved in comparisons is at most t. Note that $2\texttt{val}(x)$ is a total order, since for any x, y either $\texttt{val}(x) < \texttt{val}(y)$ or vice versa. For any comparison in $\mathcal{A}$, there is an outcome inconsistent with this ordering. (An outcome "$f(x) < f(y)$" where $\texttt{val}(x) > \texttt{val}(y)$ is inconsistent with the total order.) We construct a comparison tree $\mathcal{A}'$ where we simply reject whenever a comparison is inconsistent with the total order, and otherwise mimics $\mathcal{A}$. The comparison tree of $\mathcal{A}'$ has an error probability at most that of $\mathcal{A}$ (since it may reject a few $f \neq 2\texttt{val}$), and is just a path. Hence, it can be modeled as a non-adaptive distinguisher. We query upfront all the points involving points on this path, and make the relevant comparisons for the output.

Combined with Proposition 1, the following lemma completes the proof of Lemma 2.

Lemma 3. *Any deterministic, non-adaptive, comparison-based distinguisher $\mathcal{A}$ making fewer than $t \leq \frac{m'}{8\varepsilon}$ queries, errs with probability at least $1/8$.*

Proof. Let X be the set of points queried by the distinguisher. Set $X_k =: X \cap S_k$; these form a partition of X. We say that a pair of points (x, y) *captures* the (unique) coordinate j, if j is the largest coordinate where $x_j \neq y_j$. (By largest coordinate, we refer to most significant bit.) For a set Y of points, we say Y captures coordinate j if there is a pair in Y that captures j.

Claim. For any j, k, if the algorithm distinguishes between $\texttt{val}$ and $g_{j,k}$, then X_k captures j.

Proof. If the algorithm distinguishes between $\texttt{val}$ and $g_{j,k}$, there must exist $(x, y) \in X$ such that $\texttt{val}(x) < \texttt{val}(y)$ and $g_{j,k}(x) > g_{j,k}(y)$. We claim that x and y capture j; this will also imply they lie in the same $S_{k'}$ since the $m - j$ most significant bit of x and y are the same.

Firstly, observe that we must have $y_j = 1$ and $x_j = 0$; otherwise, $g_{j,k}(y) - g_{j,k}(x) \geq 2(\texttt{val}(y) - \texttt{val}(x)) > 0$ contradicting the supposition. Now suppose (x, y) don't capture j implying there exists $i > j$ which is the largest coordinate at which they differ. Since $\texttt{val}(y) > \texttt{val}(x)$ we have $y_i = 1$ and $x_j = 0$. Therefore, we have

$$g_{j,k}(y) - g_{j,k}(x) \geq 2(\texttt{val}(y) - \texttt{val}(x)) - 2^j - 1 \geq (2^i + 2^j) - \sum_{1 \leq r < i} 2^r - 2^j - 1 > 0.$$

So, x, y capture j and lie in the same $S_{k'}$. If $k' \neq k$, then again $g_{j,k}(y) - g_{j,k}(x) = 2(\texttt{val}(y) - \texttt{val}(x)) > 0$. Therefore, X_k captures j.

The following claim allows us to complete the proof of the lemma.

Claim. A set Y captures at most $|Y| - 1$ coordinates.

Proof. We prove this by induction on $|Y|$. When $|Y| = 2$, this is trivially true. Otherwise, pick the largest coordinate j captured by Y and let $Y_0 = \{y : y_j = 0\}$ and $Y_1 = \{y : y_j = 1\}$. By induction, Y_0 captures at most $|Y_0| - 1$ coordinates, and Y_1 captures at most $|Y_1| - 1$ coordinates. Pairs $(x, y) \in Y_0 \times Y_1$ only capture coordinate j. Therefore, the total number of captured coordinates is at most $|Y_0| - 1 + |Y_1| - 1 + 1 = |Y| - 1$.

If $|X| \leq m'/8\varepsilon$, then there exist at least $1/4\varepsilon$ values of k such that $|X_k| \leq m'/2$. By the previous claim, each such X_k captures at most $m'/2$ coordinates. Therefore, there exist at least $\frac{1}{4\varepsilon} \cdot \frac{m'}{2} = \frac{m'}{8\varepsilon}$ functions $g_{j,k}$ that are indistinguishable from the monotone function $\texttt{2val}$ to a comparison-based procedure that queries X. This implies the distinguisher must err (make a mistake on either these $g_{j,k}$s or $\texttt{2val}$) with probability at least $\min(\frac{\varepsilon}{m'} \cdot \frac{m'}{8\varepsilon}, 1/2) = 1/8$.

3.2 The final bound

Recall, given function $f : \{0,1\}^{d\ell} \to \mathbb{N}$, we have the function $\widetilde{f} : [n]^d \to \mathbb{N}$ by defining $\widetilde{f}(\boldsymbol{y}) := f(\phi(\boldsymbol{y}))$. We start with the following observation.

Proposition 2. *The function $\widetilde{\texttt{2val}}$ is monotone and every $\widetilde{g_{j,k}}$ is $\varepsilon/2$-far from being monotone.*

Proof. Let $\boldsymbol{u}$ and $\boldsymbol{v}$ be elements in $[n]^d$ such that $\boldsymbol{u} \prec \boldsymbol{v}$. We have $\texttt{val}(\phi(\boldsymbol{u})) < \texttt{val}(\phi(\boldsymbol{v}))$, so $\widetilde{\texttt{2val}}$ is monotone. For the latter, it suffices to exhibit a matching of violated pairs of cardinality $\varepsilon 2^{d\ell}$ for $\widetilde{g_{j,k}}$. This is given by pairs $(\boldsymbol{u}, \boldsymbol{v})$ where $\phi(\boldsymbol{u})$ and $\phi(\boldsymbol{v})$ only differ in their jth coordinate, and are both contained in S_k. Note that these pairs are comparable in $[n]^d$ and are violations.

Theorem 4. *Any $(t, \varepsilon/2, 1/16)$-monotonicity tester for $f : [n]^d \to \mathbb{N}$, must have $t \geq \frac{d \log n - \log(1/\varepsilon)}{8\varepsilon}$.*

Proof. By Theorem 2, it suffices to show this for comparison-based $(t, \varepsilon/2, 1/8)$ testers. By Yao's minimax lemma, it suffices to produce a distribution $\mathcal{D}$ over functions $f : [n]^d \to \mathbb{N}$ such that any deterministic comparison-based $(t, \varepsilon/2, 1/8)$-monotonicity tester for $\mathcal{D}$ must have $t \geq s$, where $s := \frac{d \log n - \log(1/\varepsilon)}{8\varepsilon}$.

Consider the distribution $\mathcal{D}$ where we generate f from $\mathcal{F}_{m,\varepsilon}$ and output $\widetilde{f}$. Suppose $t < s$. By Proposition 2, the deterministic comparison based monotonicity tester acts as a determinisitic comparison-based distinguisher for $\mathcal{F}_{m,\varepsilon}$ making fewer than s queries, contradicting Lemma 3.

4 Conclusion

In this paper, we exhibit a lower bound of $\Omega(\varepsilon^{-1} d \log n - \varepsilon^{-1} \log \varepsilon^{-1})$ queries on adaptive, two-sided monotonicity testers for functions $f : [n]^d \to \mathbb{N}$, matching the upper bound of $O(\varepsilon^{-1} d \log n)$ queries of [15]. Our proof hinged on two things: that for monotonicity on any partial order one can focus on comparison-based testers, and a lower bound on comparison-based testers for the hypercube domain. Some natural questions are left open. Can one focus on some restricted class of testers for the Lipschitz property, and more generally, can one prove adaptive, two-sided lower bounds for the Lipschitz property testing on the hypergrid/cube? Currently, a $\Omega(d \log n)$-query non-adaptive lower bound is known for the problem [17]. Can one prove comparison-based lower bounds for monotonicity testing on a general N-vertex poset? For the latter problem, there is a $O(\sqrt{N/\varepsilon})$-query non-adaptive tester, and a $\Omega(N^{\frac{1}{\log\log N}})$-query non-adaptive, two-sided error lower bound [7]. Our methods do not yield any results for bounded ranges, but there are significant gaps in our understanding for that regime. For monotonicity testing of boolean functions $f : \{0,1\}^n \to \{0,1\}$, the best adaptive lower bound of $\Omega(\log n)$, while the best non-adaptive bound is $\Omega(\sqrt{n})$ [7].

References

1. Rubinfeld, R., Sudan, M.: Robust characterization of polynomials with applications to program testing. SIAM Journal of Computing **25** (1996) 647–668 1
2. Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. Journal of the ACM **45**(4) (1998) 653–750 1
3. Goldreich, O., Goldwasser, S., Lehman, E., Ron, D., Samorodnitsky, A.: Testing monotonicity. Combinatorica **20** (2000) 301–337 1, 2
4. Ergun, F., Kannan, S., Kumar, R., Rubinfeld, R., Viswanathan, M.: Spot-checkers. Journal of Computer Systems and Sciences (JCSS) **60**(3) (2000) 717–751 2
5. Dodis, Y., Goldreich, O., Lehman, E., Raskhodnikova, S., Ron, D., Samorodnitsky, A.: Improved testing algorithms for monotonicity. Proceedings of the 3rd International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM) (1999) 97–108 2
6. Lehman, E., Ron, D.: On disjoint chains of subsets. Journal of Combinatorial Theory, Series A **94**(2) (2001) 399–404 2
7. Fischer, E., Lehman, E., Newman, I., Raskhodnikova, S., Rubinfeld, R., Samorodnitsky, A.: Monotonicity testing over general poset domains. In: Proceedings of the 34th Annual ACM Symposium on the Theory of Computing (STOC). (2002) 474–483 2, 10
8. Ailon, N., Chazelle, B.: Information theory in property testing and monotonicity testing in higher dimension. Information and Computation **204**(11) (2006) 1704–1717 2
9. Fischer, E.: On the strength of comparisons in property testing. Information and Computation **189**(1) (2004) 107–116 2, 3, 4
10. Halevy, S., Kushilevitz, E.: Testing monotonicity over graph products. Random Structures and Algorithms **33**(1) (2008) 44–67 2
11. Parnas, M., Ron, D., Rubinfeld, R.: Tolerant property testing and distance approximation. Journal of Computer and System Sciences **6**(72) (2006) 1012–1042 2
12. Ailon, N., Chazelle, B., Comandur, S., Liu, D.: Estimating the distance to a monotone function. Random Structures and Algorithms **31**(3) (2006) 1704–1711 2
13. Batu, T., Rubinfeld, R., White, P.: Fast approximate PCPs for multidimensional bin-packing problems. Information and Computation **196**(1) (2005) 42–56 2
14. Blais, E., Brody, J., Matulef, K.: Property testing lower bounds via communication complexity. Computational Complexity **21**(2) (2012) 311–358 2
15. Chakrabarty, D., Seshadhri, C.: Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In: Proceedings of Symposium on Theory of Computing (STOC). (2013) 2, 9

16. Brody, J.: Personal communication (2013) 2
17. Blais, E., Raskhodnikova, S., Yaroslavtsev, G.: Lower bounds for testing properties of functions on hypergrid domains. Technical Report TR13-036, ECCC (March 2013) 2, 3, 9
18. Briët, J., Chakraborty, S., García-Soriano, D., Matsliah, A.: Monotonicity testing and shortest-path routing on the cube. Combinatorica **32**(1) (2012) 35–53 2
19. Bollobás, B.: Modern Graph Theory. Springer (2000) 5