SAND2012-9447P

Sublogarithmic monotonicity testers for boolean functions

Deepamab Chakrabarty
Microsoft Research India
dechakr@microsoft.com

C. Seshadhri*
Sandia National Labs, Livermore
scomand@sandia.gov

ABSTRACT
Given oracle access to a Boolean function $f : \{0,1\}^n \mapsto \{0,1\}$, we design a randomized tester which takes as input a parameter $\varepsilon > 0$, and outputs Yes if the function is monotone, and outputs No with probability $> 2/3$, if the function is ε-far from monotone. That is, f needs to be modified at ε-fraction of the points to make it monotone. Our algorithms makes $O(n^{0.95}\varepsilon^{1.45})$ queries to the oracle.

This answers a more than decade old question of Goldreich et. al. [12], who exhibitied a $O(n/\varepsilon)$ tester, and asked whether testers exist with a significantly lower dependence on n. In fact, for a large class of functions, namely functions with low average sensitivity, we can show that $O(\sqrt{n} \cdot \text{poly}(1/\varepsilon))$ queries suffice. This is optimal for non-adaptive testers, since the $\Omega(\sqrt{n})$ lower bound examples of [11] have low average sensitivity.

Our tester is based on random walks on the directed hypercube. We show the following rather curious result: if S be a subset of points on the boolean hypercube of size $\varepsilon 2^n$; then the probability that a particular random walk on the directed hypercube starts and ends in S, is at least $\Omega(\varepsilon^2/\ln(1/\varepsilon))$. Observe that the probability two *independent* samples lie in S is ε^2; the start and end points of a random walk are highly correlated.

The above result forms the first piece of our analysis. Along with this idea, we need to use some of the techniques developed in an earlier paper [7] by us, and a non-trivial result of Lehman and Ron [14], to complete the analysis.

General Terms
Theory

*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.

Keywords
Property Testing, Monotonicity, Random walks

1. INTRODUCTION
Testing monotonicity of boolean functions is one of the fundamental questions in the area of property testing. The boolean hypercube $\{0,1\}^n$ defines the natural partial order $\prec$ where $x \prec y$ iff $x_i \leq y_i$ for all $i \in [n]$. A boolean function $f : \{0,1\}^n \mapsto \{0,1\}$ is *monotone* if $f(x) \leq f(y)$ whenever $x \prec y$. We are interested in testing whether a boolean function, given oracle access, is monotone, making as few queries as possible.

The *distance* of a boolean function to monotonicity is the minimum fraction of points at which it needs to be modified to make it monotone. This parameter is denoted as $\varepsilon_M(f)$. In the property testing framework, a monotonicity tester is given a parameter $\varepsilon > 0$, and it is supposed to (a) accept, if the function is monotone, and, (b) reject, if the function is ε-far from monotone, that is, $\varepsilon_M(f) \geq \varepsilon$. The algorithm is allowed to be randomized, and in that case, one expects (a) and (b) to occur with non-trivial probability (say $> 2/3$). An algorithm is called a *one-sided tester* if case (a) occurs with probability 1. An algorithm is *non-adaptive* if the queries made by the algorithm doesn't depend on the answers given by the oracle.

The quality of a monotonicity tester is governed by both the number of queries it makes, and the running time of the algorithm. Goldreich et al. [12] suggested the following, rather simple, algorithm: query the function value on a pair of points which differ on exactly a single coordinate and reject if monotonicity is violated. In other words, the algorithm samples a random edge of the hypercube and checks for monotonicity between the two endpoints. This is called the *edge tester* for monotonicity; it is clear the running time is of the same order as the query time.

Goldreich et al. [12] show that $O(n/\varepsilon)$-queries by the edge tester suffice to test monotonicity. They also show that their analysis is tight; the edge tester can do no better. They explicitly ask the obvious question: does there exist some other tester with an improved query complexity in terms of n? Fischer et al. [11] show that any non-adaptive, one-sided tester[1] for monotonicity must make $\Omega(\sqrt{n})$-queries for constant $\varepsilon > 0$. Since then, no significant[2] progress has been

[1] [11] also show a $\Omega(\log n)$ lower bound for 2-sided testers.
[2] However, see §1.2

made on this decade long question. In this paper, we answer the question of [12] in the affirmative.

THEOREM 1. *There exists an $O(n^{0.95}\varepsilon^{-1.45})$-query tester for monotonicity of boolean functions $f : \{0,1\}^n \mapsto \{0,1\}$.*

Our tester is one-sided and non-adaptive. In fact, our tester falls in the class of what are known as pair testers. We describe a distribution on pairs ($x \prec y$) independent of the function, and make independent queries on pairs drawn from this distribution. We reject if and only if some drawn pair violates monotonicity. Observe that the edge tester is a special pair tester.

At a high level, the distribution is defined via random walks on the directed hypercube with an arc from x to y if $x \prec y$ and differ in exactly one coordinate. We consider a random path from 0^n to 1^n and sample a pair of points on this path. (We do need some more technical conditions, but this should suffice for now.) We call this tester the random path tester. Our final tester which proves the above theorem is obtained by running either the random path tester or the edge tester with probability 1/2.

Although the exponent of n in Theorem 1 seems quite close to 1, we believe the analysis of the random path tester may be improved. In particular, we do not know whether the tester needs to make $\omega(\sqrt{n})$-queries when ε is a constant. We leave this as an open question. What we can show is that for a large class of functions, the random path tester does indeed work with $O(\sqrt{n})$-queries. More precisely, if a function is ε-far from monotone and has *low average sensitivity* (smaller than any polynomial in n), then the random path tester detects a violation with $\tilde{O}(1/\sqrt{n})$ probability.

Given a boolean function f, the influence of dimension i, denoted as Inf_i, is defined as the fraction of edges on the hypercube crossing the ith dimension whose endpoints have different function values. In other words, Inf_i measures how sensitive the function is to the flip of dimension i. The average sensitivity, also called the total influence, is the sum of all the n influences. The average sensitivity of a function f is denoted as $\mathbf{I}(f)$. Clearly, $\mathbf{I}(f) \leq n$, for any boolean function f. It is not too hard to show that for monotone functions, $\mathbf{I}(f) \leq \sqrt{n}$. In fact, one can show that if $\mathbf{I}(f)$ is 'high', say $\geq 1.1\sqrt{n}$, then in fact $O(\sqrt{n})$-queries of the edge tester can detect non-monotonicity. Our next theorem shows that the random path tester is good for 'low' average sensitivity functions.

THEOREM 2. *For any boolean function which is ε-far from monotone, the random path tester detects a violation with probability $\Omega\left(\frac{1}{\sqrt{n}} \cdot \mathrm{poly}\left(\frac{\varepsilon}{\mathbf{I}(f)}\right)\right)$. As a corollary, the random path tester is an $O(n^{\frac{1}{2}+o(1)}\mathrm{poly}(1/\varepsilon))$-query monotonicity tester for functions with average sensitivity $n^{o(1)}$.*

The class of functions having average sensitivity $n^{o(1)}$ is non-trivial. For instance, it contains boolean functions captured by constant depth circuits [15]. Furthermore, it includes the example of Fischer et al. [11]. In fact, the function of [11] has constant average sensitivity implying that our tester is optimal, as far as the dependence in n is concerned, for constant average sensitivity functions.

We end the introduction by putting our results in perspective. Monotonicity is a fundamental property of functions, and has been extensively studied [9, 12, 8, 14, 11, 1, 10, 13, 16, 2, 3, 4, 6, 5, 7]. In this paper, we focus on boolean functions defined over the hypercube. However, one could ask the same question in richer domains and ranges. Improving on a line of work, the authors of this paper show [7] that the edge tester of [12] works in $O(n/\varepsilon)$-time for any range for functions over the hypercube. This is optimal for non-adaptive, one-sided testers [6], and close to optimal for general testers [5]. [7] also give the best testers known when the domain is the hypergrid $[k]^n$. One of the main techniques of that paper is a charging scheme based on matchings and alternating paths; in this paper we make use of (a simplified version) of that scheme as one part of our argument.

1.1 Main Ideas.

As stated above, our algorithm samples a random path and picks two vertices randomly from it. The inspiration for our algorithm is a recent paper by Ron et al. [17] who provide an $O(\sqrt{n})$-query randomized algorithm to estimate the average sensitivity of a *monotone* function. Loosely, the algorithm described in [17] performs a random walk of around $\Theta(\sqrt{n})$ steps from a random point and checks if the endpoints are different. In fact, Ron et al. [17] explicitly ask whether an algorithm "in the spirit" above can be used for monotonicity. The answer is yes.

The key observation that [17] use is the following: for monotone functions, a path from 0^n to 1^n cannot contain more than one influential edge. Given this, the analysis reduces to calculating the probability that a random walk crosses a given edge, and then adding over all the influential edges. The problem with general non-monotone functions is that a random walk could cross two influential edges, one violating and the other not, and discover nothing from the values at the endpoints. Therefore, this idea doesn't seem to directly pan out for monotonicity testing.

Nonetheless, the idea of taking a random walk and testing "far away" points is useful. To see this, consider a toy example. Consider a function $f : \{0,1\}^{n+1} \mapsto \{0,1\}$ such that $f(0,x) = 0$ if $|x| \leq n/2 - \sqrt{n}$ and 1 if $|x| > n/2 - \sqrt{n}$. However, $f(1,x) = 0$ if $|x| \leq n/2 + \sqrt{n}$ and 1 otherwise. Note that the violations to monotonicity are across the first dimension (which, by the way, the tester isn't aware of) on the edges $((0,x),(1,x))$ for $n/2 - \sqrt{n} \leq |x| \leq n/2 + \sqrt{n}$. The function is ε-far from monotone for some constant ε. Also observe the edge tester needs $\Theta(n)$ queries.

How does the random walk tester fare? With constant probability it picks a random point $(0,x)$ with $n/2 - \sqrt{n} < |x| \leq n/2$. So, $f(0,x) = 1$. Then suppose it makes $\sqrt{n}$ steps. If in any of these steps it flips coordinate 1, then observe that it will catch a violation. This is because it'll end up at $(1,x')$ with $n/2 < |x'| \leq n/2 + \sqrt{n}$ which evaluates to 0. What is the probability that this dimension 1 is flipped in $\sqrt{n}$ steps? It is $\Theta(1/\sqrt{n})$; and thus the random walk tester works with

$O(\sqrt{n})$ queries for this function.

Consider, now, the case of what we call almost monotone functions. Such functions can be thought of as two monotone functions defined on two hypercubes, with all the violations between these hypercubes. That is, all violations to monotonicity are along one unknown coordinate. Note that the above 'flip once in $\sqrt{n}$ steps' argument can be carried over if we could lower bound the probability that we start at an endpoint of a violated edge and end at the endpoint of another violated edge. Since the function is ε-far, we know that the number of such endpoints is $\Omega(\varepsilon 2^n)$, but unlike in the example of the previous paragraph, these could be peppered throughout the cube.

Our first result is precisely bounding such a probability. In particular, we show that if a constant ε fraction of the points in the hypercube are, say, colored blue, then the probability a random walk starts and ends at blue is $\Omega(\varepsilon^2/\ln(1/\varepsilon))$. Note that if we sample two points independently, the probability that they are both blue would be ε^2; our result shows that the correlation caused by the random walk doesn't degrade too much.

The above idea can be extended to the case where instead of all violations being along one coordinate, we rather have a large, that is of size $\Omega(\varepsilon 2^n)$, *matching* of violated edges. In almost monotone functions, this was indeed the case. The idea is to look at the endpoints of this matching which evaluate to 1, and use the above argument to show that the tester will pick a pair of these with poly(ε) probability. After that, we can *charge* these events to events which pick a 1-endpoint and a ancestor 0-endpoint, taking a loss of $\frac{\varepsilon}{\sqrt{n}}$. The matching property is crucially used in the charging. However, to make this idea go through, we need to modify the random walk a bit which leads to a slight degradation in the parameter of ε in the 'blue' result mentioned above. Basically, we need to make sure the walk doesn't end "too fast".

What if there is no large matching of violated edges? Since f is ε-far, we know that any *maximal* matching M of violated *pairs* has $|M| \geq \varepsilon 2^{n-1}$ [8]. If the *average length* of these pairs is not too large, then we show via a simple counting argument that there must lie a large enough matching of violated edges reducing it to the previous case. Of course, there is a loss in the size, and this is why the exponent of n takes a hit. What if the average length of *every* maximal matching is large? This is where we use the machinery developed in our previous paper [7]. We show that if the matching which minimizes the average length is still "too long", then there must be lots of violated edges. And therefore, the edge tester suffices.

1.2 Comparison with a result of Briët et. al.

In a recent paper, Briët et al. [6] prove a lower bound of $\Omega(n/\varepsilon)$ on non-adaptive, 1-sided monotonicity testing for general ranges. The ranges of their bad examples are of size $\Theta(\sqrt{n})$. Using a 'range reduction' result of Dodis et al. [8], they deduce that there can be no $o(\frac{n}{\varepsilon \log n})$-query *pair testers* for monotonicity of Boolean functions. In particular, Dodis et al. [8] prove that if there is a distribution on pairs such that for any Boolean function f, the probability a violation is caught is at least $\varepsilon_M(f)/C(n)$, then for any function g with range $\mathbf{R}$, the probability (w.r.t. the same distribution) a violation is caught is at least $\varepsilon_M(g)/(C(n)\log|\mathbf{R}|)$. Thus, if pair testers could detect violations with probability $\geq \varepsilon_M(f)\log n/n$, then it would violate the lower bound for general monotone functions as proven by Briët et al. [6].

Our tester is a pair tester, and we claim a much better dependency on n than $n/\log n$. The reconciliation lies in the exponent of ε. Note that in Theorem 1, the exponent of ε is smaller than -1. Briët et al. [6]'s result shows that this degradation is necessary. We remark that the reduction of Dodis et al. [8] doesn't work if the success probability of the Boolean tester is guaranteed to be, say $\varepsilon_M^2(f)/C(n)$. In particular, nothing, as far as we know, rules out a pair tester running in time $O(\sqrt{n}\cdot \text{poly}(1/\varepsilon))$.

2. THE TESTER AND ITS ANALYSIS

Recall, we are given a parameter $\varepsilon > 0$; we want to accept if the function is monotone, and reject if $\varepsilon_M(f) \geq \varepsilon$. We may assume $\varepsilon \leq 1/2$ since any function can be made monotone by changing at most $1/2$ of its values. Set parameter $\ell := 2C_\varepsilon\sqrt{n}$, where $C_\varepsilon = \sqrt{10\ln(1/\varepsilon)}$. We use $|x|$ to denote the number of 1's in a binary vector $x \in \{0,1\}^n$. The directed hypercube is the directed graph with vertex set $\{0,1\}^n$, and an arc from x to y if $x \prec y$ and $|y| - |x| = 1$.

Random Path Tester.

1. Let $\mathcal{P}$ be the collection of paths in the directed hypercube from 0^n to 1^n. Pick a path $\mathbf{p} \in \mathcal{P}$ u.a.r. Let $X_\mathbf{p} := \{z \in \mathbf{p} : |z| \in [\frac{n}{2} - C_\varepsilon\sqrt{n}, \frac{n}{2} + C_\varepsilon\sqrt{n}]\}$, $|X_\mathbf{p}| = \ell$.
2. Sample $x \in X_\mathbf{p}$ uniformly at random.
3. Let $Y_\mathbf{p}(x) := \{z \in X_p : ||z-x||_1 \geq \lfloor \frac{\varepsilon\ell}{32C_\varepsilon} \rfloor\}$. Sample y uniformly at random from $Y_\mathbf{p}(x)$.
4. Reject if (x, y) violate monotonicity; that is, $f(x) < f(y), x \succ y$ or $f(x) > f(y), x \prec y$.

We also have the edge tester which picks a random edge (x, y) of the hypercube and checks for violations among this pair. Our final tester flips a coin and with probability $1/2$ performs either the edge test or the path test.

Observe that both the edge tester and the path tester always accept if the function f is monotone. Therefore, our tester is a one-sided tester. To prove Theorem 1, it suffices to show that if f is ε-far from monotone, then the probability our tester rejects is at least $\Omega(n^{-0.95}\varepsilon^{1.45})$. Henceforth, we assume the function f is ε-far, and we call the rejection event a success. Recall, since f is ε-far from monotonicity, any maximal set M of disjoint, violating pairs, which we henceforth call a *matching* of violated pairs, satisfies $|M| \geq \varepsilon 2^{n-1}$ [8]. The lower bound on the success probability is derived from the following four easy pieces.

- Suppose one colors *any* ε-fraction of the points on the hypercube blue. Our first lemma (Lemma 1) shows that the probability that both the points x, y sampled by the path tester is blue, is at least $\Omega(\varepsilon^{4.5})$.

- Our second lemma (Lemma 2) shows that if there is a collection of hypercube edges, E, which from a matching *and* each edge in E is a violation, then the probability that the path tester succeeds is at least
$$\frac{\varepsilon}{\sqrt{n}} \cdot \Omega\left(\left(\frac{|E|}{2^n}\right)^{4.5}\right).$$
- Our third step argues that if the *average length* of pairs in a maximal matching M of violated pairs (where length of (x, y) is just $|y| - |x|$) is small, say $\le r$, then there exists a matching of violated edges, E, with $|E| \ge |M|/16r^2$. This, along with the second point above, proves Theorem 1 when the average length, r, is small, that is $r \le n^{0.05}\varepsilon^{0.45}$.
- We choose the maximum cardinality matching $M$of violated pairs which minimizes the average length. Using a technique from our earlier paper [7], we prove that if the average length of M is r, then the number of violated edges is $\ge \frac{r}{\varepsilon}2^n$. This implies that the edge tester succeeds with probability $\Omega(\frac{r\varepsilon}{n})$. This proves Theorem 1 when $r \ge n^{0.05}\varepsilon^{0.45}$.

2.1 Piece 1.

Assume ε-fraction of the hypercube is colored blue. Let x and y be a pair sampled by the path tester. We are interested in the event
$$\mathcal{E}: \ x \text{ and } y \text{ are both blue}$$

Let's $b(\mathbf{p})$ denote the number of blue points in the set $X_{\mathbf{p}}$ corresponding to a path $\mathbf{p}$. At times, we abuse notation, and let it be a random variable corresponding to the path $\mathbf{p}$ chosen u.a.r from $\mathcal{P}$. The probability that the first point sampled by the random path tester is blue, conditioned on the probability that $\mathbf{p}$ is sampled, is precisely $\frac{b(\mathbf{p})}{\ell}$.

The probability that the second point is blue is at least
$$\frac{|\text{blue points in } Y_{\mathbf{p}}(x)|}{|Y_{\mathbf{p}}(x)|} \ge \frac{b(\mathbf{p}) - \varepsilon\ell/16C_\varepsilon}{\ell}.$$
This is because there are at least $(b(\mathbf{p}) - \varepsilon\ell/16C_\varepsilon)$ blue points in $Y_{\mathbf{p}}(x)$. Therefore, the probability that both points sampled by the random path tester are blue, is
$$\Pr[\mathcal{E}] \ \ge \ \frac{1}{|\mathcal{P}|}\sum_{\mathbf{p}\in\mathcal{P}}\left(\frac{b(\mathbf{p})}{\ell} \cdot \frac{b(\mathbf{p}) - \varepsilon\ell/16C_\varepsilon}{\ell}\right). \tag{1}$$

We now lower bound the RHS of the above. To do so, we perform the sum only over paths $\mathbf{p}$ with $b(\mathbf{p}) \ge \ell\varepsilon/8C_\varepsilon$. Let us use $\mathcal{Q} := \{\mathbf{p} : b(\mathbf{p}) \ge \varepsilon\ell/8C_\varepsilon\}$. In that case, the numerator of the second fraction becomes at least $b(\mathbf{p})/2$. Therefore we get,
$$\Pr[\mathcal{E}] \ \ge \ \frac{1}{2|\mathcal{P}|}\sum_{\mathbf{p}\in\mathcal{Q}}\left(\frac{b(\mathbf{p})}{\ell}\right)^2. \tag{2}$$

We now investigate how $b(\mathbf{p})$ looks like. Let L_i denote the set of points in the hypercube having i ones. That is, $L_i := \{x \in \{0,1\}^n : |x| = i\}$. Note that
$$|L_i| \le \binom{n}{n/2} \le \frac{2^n}{\sqrt{n}}.$$
Let n_i be the number of blue nodes in layer L_i. If we let X_i be the indicator variable for the event whether the ith layer vertex in $\mathbf{p}$ is blue or not, then we get
$$b(\mathbf{p}) = X_1 + \cdots + X_\ell.$$
Now, a $\mathbf{p}$ chosen u.a.r from $\mathcal{P}$ contains a vertex in layer L_i u.a.r. as well, for all i. Therefore,
$$\mathbf{Exp}[X_i] = \frac{n_i}{|L_i|} \ge \frac{\sqrt{n}}{2^n}\cdot n_i.$$
Furthermore, by a standard application of Chernoff bounds, we get that the number of points of the hypercube not lying in any of the L_i's is at most $2\varepsilon^5 \cdot 2^n$. Even if all of them are colored blue, we get
$$\sum_i n_i \ge (\varepsilon - 2\varepsilon^5)2^n \ge (\varepsilon/2)2^n.$$
since $\varepsilon \le 1/2$. Thus,
$$\mathbf{Exp}[b(\mathbf{p})] \ge \frac{\sqrt{n}}{2^n}\sum_{i=1}^{\ell} n_i \ge \frac{\sqrt{n}}{2}\varepsilon = \frac{\ell\varepsilon}{4C_\varepsilon}.$$
Taking the expectation now path-by-path, we get that
$$\frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}} b(\mathbf{p}) \ge \frac{\ell\varepsilon}{4C_\varepsilon}. \tag{3}$$

From (3), we get
$$\Pr[\mathbf{p} \in \mathcal{Q}] := \Pr[b(\mathbf{p}) \ge \varepsilon\ell/8C_\varepsilon] \ge \frac{\varepsilon}{8C_\varepsilon} \tag{4}$$
This is because the maximum value of $b(\mathbf{p})$ is ℓ. If (4) didn't hold we would get,
$$\mathbf{Exp}[b(\mathbf{p})] \le \ell \cdot \Pr[b(\mathbf{p}) \ge \varepsilon\ell/8C_\varepsilon] + \ell\varepsilon/8C_\varepsilon \cdot 1 < \ell\varepsilon/4C_\varepsilon$$
contradicting (3).

Now we are ready lower bound $\Pr[\mathcal{E}]$.

LEMMA 1.
$$\Pr[\mathcal{E}] \ \ge \ \frac{1}{2|\mathcal{P}|}\sum_{\mathbf{p}\in\mathcal{Q}}\left(\frac{b(\mathbf{p})}{\ell}\right)^2 \ \ge \ \frac{\varepsilon^3}{1024\ C_\varepsilon^3} \ = \Omega(\varepsilon^{4.5}).$$

PROOF. The first inequality is the same as (2). Note that this can be restated as
$$\Pr[\mathcal{E}] \ge \frac{1}{2}\frac{|\mathcal{Q}|}{|\mathcal{P}|}\cdot\frac{1}{|\mathcal{Q}|}\sum_{\mathbf{p}\in\mathcal{Q}}\left(\frac{b(\mathbf{p})}{\ell}\right)^2. \tag{5}$$
From (4), we get that
$$\frac{|\mathcal{Q}|}{|\mathcal{P}|} \ge \frac{\varepsilon}{8C_\varepsilon}. \tag{6}$$
By Jensen's inequality, we get that
$$\frac{1}{|\mathcal{Q}|}\sum_{\mathbf{p}\in\mathcal{Q}}\left(\frac{b(\mathbf{p})}{\ell}\right)^2 \ge \left(\frac{1}{|\mathcal{Q}|}\sum_{\mathbf{p}\in\mathcal{Q}}\frac{b(\mathbf{p})}{\ell}\right)^2. \tag{7}$$

Since for each $\mathbf{p} \in \mathcal{Q}$, $b(\mathbf{p}) \geq \varepsilon\ell/8C_\varepsilon$, we get that

$$\left(\frac{1}{|\mathcal{Q}|}\sum_{\mathbf{p}\in\mathcal{Q}}\frac{b(\mathbf{p})}{\ell}\right)^2 \geq \frac{\varepsilon^2}{64C_\varepsilon^2}. \tag{8}$$

Plugging (6),(7),(8) in (5) gives the lemma. The last inequality in the lemma follows from the definition of C_ε, and the rather weak inequality, $\ln(1/\varepsilon) \leq 1/\varepsilon$. □

2.1.1 An Aside

We think that the ideas presented in the previous subsection are worth distilling in this separate subsection. Consider the following simplification of our random path tester: pick the path $\mathbf{p}$ randomly from $\mathcal{P}$, and then sample two points x, y independently from $\mathbf{p}$. What's the probability of the event $\mathcal{E}$ that x and y are both blue?

The calculation becomes slightly simpler. We get

$$\Pr[\mathcal{E}] = \frac{1}{|\mathcal{P}|}\sum_{\mathbf{p}\in\mathcal{P}}\left(\frac{b(\mathbf{p})}{\ell}\right)^2 \geq \left(\frac{1}{|\mathcal{P}|}\sum_{\mathbf{p}\in\mathcal{P}}\frac{b(\mathbf{p})}{\ell}\right)^2 = \Omega\left(\frac{\varepsilon^2}{\ln(1/\varepsilon)}\right) \tag{9}$$

where the second inequality follows from Jensen's inequality, and the last inequality follows from (3). Finally note that the above random process can be thought of as the following random walk.

<u>DirRandWalk:</u> Pick $i, j \in [\frac{n}{2} - C_\varepsilon\sqrt{n}, \frac{n}{2} + C_\varepsilon\sqrt{n}]$ u.a.r. Pick $x \in L_i$ u.a.r. and perform a random walk starting from x up the directed hypercube for j steps. End at vertex y.

It is not hard to see that the distribution on pairs (x, y) is the same as produced by the random process described above: for any $x \prec y$ with number of ones in $[\frac{n}{2} - C_\varepsilon\sqrt{n}, \frac{n}{2} + C_\varepsilon\sqrt{n}]$, the probability we get (x, y) is precisely $\frac{1}{\ell^2} \cdot \frac{t!s!(n-s-t)!}{n!}$, where $|x| = t$ and $|y| - |x| = s$. Therefore, we have the following theorem.

THEOREM 3. *Given a subset S of points in the hypercube of size $|S| \geq \varepsilon 2^n$, the probability that the* DirRandWalk *starts and ends at S is $\Omega(\varepsilon^2/\ln(1/\varepsilon))$.*

Observe that if x, y were sampled *independently*, then the probability both would be in S is ε^2. The theorem above shows that one gets a qualitatively same probability even with the highly correlated pairs generated by the directed random walk.

2.2 Piece 2.

We now show that if there is a large matching E of violating edges of the hypercube, then the path tester succeeds with large probability. We assume that all edges of E lie between the layers $(n/2 - C_\varepsilon\sqrt{n}, n/2 + C_\varepsilon\sqrt{n})$. That is, for all $(x, y) \in E$, we have $\frac{n}{2} - C_\varepsilon\sqrt{n} < |x|, |y| < \frac{n}{2} + C_\varepsilon\sqrt{n}$. We call such edges to be in the *middle layer*.

LEMMA 2. *Suppose there exists a collection E of matching, violated edges in the middle layer of the hypercube. Then the random path tester succeeds with probability*

$$\Omega\left(\frac{\varepsilon}{\sqrt{n}}\right) \cdot \left(\frac{|E|}{2^n}\right)^{4.5}.$$

PROOF. Given the matching E, let's denote the set of endpoints of edges in E, as B. We partition B into B_0 and B_1, to indicate the points where the function evaluates to 0 and those where the function evaluates to 1, respectively. Note that $|B_0| = |B_1| = |E|$. Let η denote the fraction $|E|/2^n$.

Let us focus on pairs $(x, y) \in B_1 \times B_1$. Let $\mathcal{E}_{xy}$ denote the event that the path tester chooses x and y. If we think of the vertices in B_1 to be colored bluw, then Lemma 1 implies,

$$\sum_{(x,y)\in B_1\times B_1}\Pr[\mathcal{E}_{xy}] = \Omega(\eta^{4.5}). \tag{10}$$

We can also think of the event $\mathcal{E}_{xy}$ as picking a path containing both x and y, and conditioned on this, sampling x and y. In other words,

$$\Pr[\mathcal{E}_{xy}] = \sum_{\mathbf{p}:x,y\in\mathbf{p}}\Pr[\mathbf{p} \text{ sampled }] \cdot \Pr[x, y \text{ sampled } \mid \mathbf{p} \text{ sampled }].$$

The crucial observation is that the probability x and y are sampled conditioned on the path $\mathbf{p}$ is the *same* irrespective of $\mathbf{p}$, and depends only on x and y, as long as $\mathbf{p}$ contains both of them.

Say x is the first point to be sampled. If $y \notin Y_p(x)$, then the probability y is the second point sampled is 0. Observe that if $y \notin Y_\mathbf{p}(x)$, then $x \notin Y_\mathbf{p}(y)$ as well. If $y \in Y_\mathbf{p}(x)$, then the probability y is the second point sampled is $\frac{1}{|Y_\mathbf{p}(x)|}$. Note that $|Y_\mathbf{p}(x)|$ is *independent* of $\mathbf{p}$ and depends only on $|x|$. Therefore, we get that for $x, y \in X_\mathbf{p}$,

$$\Pr[x, y \text{ sampled } \mid \mathbf{p} \text{ sampled }] = \theta_{xy} = \begin{cases} 0 & \text{if } ||y - x||_1 < \lfloor\frac{\varepsilon\ell}{32C_\varepsilon}\rfloor \\ \frac{1}{\ell}\left(\frac{1}{|Y_\mathbf{p}(x)|} + \frac{1}{|Y_\mathbf{p}(y)|}\right) & \text{otherwise.}\end{cases} \tag{11}$$

Thus, we can write

$$\Pr[\mathcal{E}_{xy}] = \theta_{xy}\frac{|\mathcal{P}_{xy}|}{|\mathcal{P}|} \tag{12}$$

where $\mathcal{P}_{xy}$ are paths from 0^n to 1^n containing both x and y.

Till now, we have been looking at the probability of picking two vertices in B_1. What we really need to sample is a vertex x in B_1 and ancestor y' in B_0. Here's where the matching will help us. We can map every event $\mathcal{E}_{xy'}$ to the event $\mathcal{E}_{xy}$, where $y = E(y')$, the matched pair of y'. Since we have a matching, the mapping is one-to-one. Now, we can lower bound our success probability as

$$\begin{aligned}\Pr[\text{ success }] &\geq \sum_{(x,y')\in B_1\times B_0} \Pr[\mathcal{E}_{xy'}] \\ &= \sum_{(x,y)\in B_1\times B_1} \Pr[\mathcal{E}_{xy}]\cdot\frac{\Pr[\mathcal{E}_{xy'}]}{\Pr[\mathcal{E}_{xy}]}\end{aligned}$$

We claim that $\theta_{xy'}$ is almost as large as θ_{xy}.

CLAIM 1. $\theta_{xy'} \geq \left(1-\frac{1}{\sqrt{n}}\right)\theta_{xy}$

PROOF. Note that since $|y'| = |y|+1$, we have $||y'-x||_1 > ||y-x||_1$ implying if $\theta_{xy'} = 0$, so is θ_{xy}. Furthermore, since $|y'| < n/2 - C_\varepsilon\sqrt{n}$, for all paths $\mathbf{p}$ containing both x and y', we have $y' \in X_{\mathbf{p}}$.

Also note that $|Y_{\mathbf{p}}(y')| \leq |Y_{\mathbf{p}}(y)|+1$ since the number of 1's in y and y' differ by at most 1. Therefore, we get

$$\frac{\theta_{xy'}}{\theta_{xy}} \geq \frac{|Y_{\mathbf{p}}(y)|}{|Y_{\mathbf{p}}(y)|+1}.$$

The claim follows by noting that $|Y_{\mathbf{p}}(y)| \geq \ell - \frac{\varepsilon\ell}{16C_\varepsilon} \geq \sqrt{n}$, since $\varepsilon < 1/2$ and $\ell = 2C_\varepsilon\sqrt{n}, C_\varepsilon = \sqrt{10\ln(1/\varepsilon)}$. $\square$

Using the above claim in (12)

$$\frac{\Pr[\mathcal{E}_{xy'}]}{\Pr[\mathcal{E}_{xy}]} \geq \left(1-\frac{1}{\sqrt{n}}\right)\frac{|\mathcal{P}_{xy'}|}{|\mathcal{P}_{xy}|} \tag{13}$$

The good thing is that we know exactly what both the numbers in the RHS are. Say $|x| = t$ and $|y| = t+s$. Note $s \geq \varepsilon\ell/32C_\varepsilon$. Also note $|y'| = |y|+1$. Then,

$$|\mathcal{P}_{xy}| = t!s!(n-s-t)! \quad \text{and} \quad |\mathcal{P}_{xy'}| = t!(s+1)!(n-s-t-1)!$$

Plugging in (13), gives that

$$\Pr[\mathcal{E}_{xy'}] \geq \left(1-\frac{1}{\sqrt{n}}\right)\frac{s+1}{n-s-t}.$$

The denominator is $\Theta(n)$ since $|y| \leq n/2 + C_\varepsilon\sqrt{n}$. The numerator is at least $\varepsilon\ell/32C_\varepsilon = \varepsilon\sqrt{n}/16$. Thus we get

$$\Pr[\mathcal{E}_{xy'}] = \Omega\left(\frac{\varepsilon}{\sqrt{n}}\right)\cdot\Pr[\mathcal{E}_{xy}].$$

Therefore the probability of success is at least

$$\begin{aligned}\sum_{(x,y')\in B_1\times B_0} \Pr[\mathcal{E}_{xy'}] &= \Omega\left(\frac{\varepsilon}{\sqrt{n}}\right)\cdot\sum_{(x,y)\in B_1\times B_1}\Pr[\mathcal{E}_{xy}] \\ &= \Omega\left(\frac{\varepsilon}{\sqrt{n}}\right)\eta^{4.5}\end{aligned}$$

where the last inequality follows from (10). This completes the proof of the lemma.

2.3 Piece 3.

Let M be a maximal matching of violated pairs. Suppose the average length of the pairs in M is r. That is,

$$\sum_{(x,y)\in M}\Big||y|-|x|\Big| = r|M|$$

Since f is ε-far, we know that $|M| = \Omega(\varepsilon 2^n)$ [8]; we will remove all pairs which have either endpoint x with $|x| \notin (\frac{n}{2} - C_\varepsilon\sqrt{n}, \frac{n}{2} + C_\varepsilon\sqrt{n})$. From a Chernoff bound we know we have removed at most $2\varepsilon^5 2^n$ pairs, and thus $|M|$ remains $\Omega(\varepsilon 2^n)$. Therefore, we may assume that all pairs of M have both endpoints in the middle layers. The main result in this section states that there is a comparable collection of matching, violated edges in the middle layer of the hypercube.

LEMMA 3. *If the average length of M is r, then there exists set E of matching, violated edges in the middle layers of the hypercube with $|E| \geq |M|/16r^2$.*

PROOF. Since the average length of M is $\leq r$, by Markov's inequality at least $|M|/2$ pairs have length $\leq 2r$. Therefore, there exists a length $\ell \in [1, 2r]$ such that at least $|M|/4r$ pairs have length exactly ℓ. Let these pairs be called M'. Partition M' into $2\ell \leq 4r$ classes as follows: (x,y) falls in class C_i if $|x| \pmod{2\ell} \equiv i$. One of the classes, say C_i, consists of at least $|M|/16r^2$ pairs.

The class C_i consists of matched pairs whose endpoints are respectively in the hypercube levels $(i, \ell+i), (2\ell+i, 3\ell+i), (4\ell+i, 5\ell+i)$ and so on. Suppose there are n_k matched pairs with one end point in layer $k\ell+i$ and other in layer $(k+1)\ell+i$; note that $\sum_{\text{even } k} n_k \geq |M|/16r^2$. We now use the following theorem of Lehman and Ron [14].

THEOREM 4 (LEHMAN-RON). *Let $\mathcal{S}, \mathcal{R}$ be two subsets of points of the hypercube such that $|\mathcal{S}| = |\mathcal{R}| = m$ and all points in $\mathcal{S}$ (respectively, $\mathcal{R}$) have s (respectively, r) ones, with $r < s$. Furthermore, suppose there is a mapping $\phi : \mathcal{S} \mapsto \mathcal{R}$ such that $\phi(r) \succ r$. Then there exists m vertex disjoint chains that contain all of $\mathcal{S}$ and $\mathcal{R}$.*

For our purposes, $\mathcal{S}$ is the set of endpoints lying on the $(k+1)\ell+1$ layer and $\mathcal{R}$ is those on the $k\ell+1$ layer. The mapping ϕ is defined by the matching. Therefore, we get n_k disjoint paths, each starting at $\mathcal{R}$ and ending at $\mathcal{S}$. Note that all vertices in $\mathcal{R}$ have function value 1, while all vertices in $\mathcal{S}$ have function value 0. Therefore, in each of these paths there must lie a violating edge. Since the paths are disjoint, the edges form a matching. Finally, edges between layer $k\ell+i$ and $(k+1)\ell+i$ cannot intersect edges found between the layers $(k+2)\ell+i$ and $(k+3)\ell+i$. Thus, we have demonstrated a matching E of size $|E| \geq \sum_{\text{even } k} n_k \geq |M|/16r^2$. Observe that the set E lies in the middle layers of the hypercube, since M has all pairs lying in the middle layers of the hypercube. $\square$

2.4 Piece 4.

For the final piece, we use the flexibility on choosing the maximal matching M. In particular, we choose M as the maximum cardinality matching of violated pairs which minimizes the average length. Clearly, M is maximal. Let M_i be the set of pairs which cross dimension i, that is, $M_i := \{(x,y) \in M : x_i = 0, y_i = 1\}$. The following theorem falls in the framework developed in the paper [7]; we provide a proof here for completeness.

THEOREM 5. *The number of violated edges across dimension i is at least $|M_i|$.*

PROOF. Let H be the *perfect* matching formed by the *edges* crossing the dimension i. Let X be the endpoints of M_i. For all $x \in X$, we define a sequence $\mathbf{S}(x)$ as follows. The first term $\mathbf{S}_x(0)$ is x. For even i, $\mathbf{S}_x(i+1) = H(\mathbf{S}_x(i))$. For odd i, if $\mathbf{S}_x(i) \in X$, or is M-unmatched, then $\mathbf{S}_x$ terminates. Otherwise, $\mathbf{S}_x(i+1) = M(\mathbf{S}_x(i))$. Above, we have used the shorthand $M(v)$ and $H(v)$ to denote the partners of v in the matchings M and H, respectively.

The best way to think about $\mathbf{S}_x$ is via alternating paths and cycles formed by the matchings M and H. We start at x and take the H-edge along the alternating path. We keep on moving till we reach an endpoint or another vertex in X. Thus, each $\mathbf{S}_x$ terminates. It's not too hard to see that if $\mathbf{S}_x$ ends at $y \in X$, then $\mathbf{S}_y$ is just $\mathbf{S}_x$ in reverse. Furthermore, $\mathbf{S}_x$ and $\mathbf{S}_y$ are disjoint unless y terminates $\mathbf{S}_x$. Therefore the number of sequences is at least $|X|/2 = |M_i|$. Now we claim that for all x, $\mathbf{S}_x$ contains a violated edge in H. This will prove the theorem.

Suppose this is not true for some vertex x. We now show that $\mathbf{S}_x$ can't terminate which will end the proof. For brevity, let's use s_i to denote $\mathbf{S}_x(i)$. Also let (x, y) be the i-crossing pair in M_i. We use s_{-1} to denote y. Wlog, assume $x \succ y$, thus $x_i = 1$ and $y_i = 0$. Also $f(y) = 1$ and $f(x) = 0$ since the pair is a violation. Note that $s_1 = H(x)$ has ith coordinate 0. Since there is no violating edge, $f(s_1) = 0$ as well. Note that $s_1 \succ y$, as well, and therefore (y, s_1) forms a violating pair. If s_1 were not matched in M, then $M-(x,y)+(y,s_1)$ would be a matching which would decrease the average length. Therefore s_1 must be matched. Since $f(s_1) = 0$, $M(s_1) = s_2 \prec s_1$. In particular, the ith coordinate of s_2 and s_1 are both 0, and therefore the edge doesn't lie in M_i. So $s_2 \notin X$. The same argument shows no other vertex can be in X. Once again, check that $s_3 = H(s_2) \prec s_0 = x$. If s_3 weren't matched, then we can replace the edges (x, y) and (s_1, s_2) in M by (y, s_1) and (s_0, s_3) to get a matching with smaller average length. Therefore, anytime we reach a vertex which is not M-matched, we get a better matching. Therefore, we never encounter a vertex which is in X or is M-unmatched. This contradicts the termination of $\mathbf{S}_x$, and therefore, there must exist a violating edge in $\mathbf{S}_x$. This proves the theorem. $\square$

This following lemma is a corollary to the above theorem.

LEMMA 4. *If the average length of the matching M is r, then there are $\geq \varepsilon r 2^{n-1}$ violated edges.*

PROOF. The proof follows by noting that $\sum_i |M_i| = \sum_{(x,y)\in M} \Big||y| - |x|\Big| = r|M|$, since a pair (x, y) appears in precisely $|y| - |x|$ different M_i's. Theorem 5 implies that the number of violated edges is at least $\sum_i |M_i|$, and therefore, since $|M| \geq \varepsilon 2^{n-1}$, the lemma follows. $\square$

2.5 Putting it all together.

In this section we prove Theorem 1 and Theorem 2.

Proof of Theorem 1: We start off with the matching described in Piece 4. Let r be the average length. Then the edge tester succeeds with with probability $\Omega(\varepsilon r/n)$. Let's calculate the success probability of the path tester. We find matching violated edges of size $|E| = |M|/16r^2$. Since $|M| \geq \varepsilon 2^{n-1}$, this gives $|E| \geq \frac{\varepsilon}{32r^2} \cdot 2^n$. Thus, we have $\eta = \frac{\varepsilon}{32r^2}$. Plugging in Lemma 2, we get that the random path tester succeeds with probability

$$\Omega\left(\frac{\varepsilon^{5.5}}{r^9\sqrt{n}}\right)$$

Therefore, the hybrid tester succeeds with probability at least

$$\Omega\left(\max\left(\frac{r\varepsilon}{n}, \frac{\varepsilon^{5.5}}{r^9\sqrt{n}}\right)\right) = \Omega\left(n^{-19/20}\varepsilon^{1.45}\right)$$

The equality follows by setting $r = n^{1/20}\varepsilon^{0.45}$. $\square$

Proof of Theorem 2: Let M be the maximum cardinality matching which minimizes the average length, as in Piece 4, and let r be the average length. Lemma 4 gives us that the number of violated edges is at least $\varepsilon r 2^{n-1}$. This in turn implies that $\mathbf{I}(f) \geq \varepsilon r/2$; every violated edge is a 'sensitive' edge. In other words, $r = O(\mathbf{I}(f)/\varepsilon)$. Theorem 2 now follows from Lemma 3 and Lemma 2. $\square$

3. CONCLUSION

In this paper, we make progress on a decade old question of testing monotonicity of boolean functions over the hypercube. In particular, we exhibit a $o(n)$-query tester. Although our current exponent is far from the known lower bound of $\sqrt{n}$, for a large class of functions we show that our analysis suffices to give optimal results.

It should be clear to the reader that we haven't optimized the exponent on n, ε in the previous analysis. In particular, the exponent 4.5 of ε in Lemma 1 is suboptimal and we chose it so that we don't bear logs around. How far can our current analysis hope to bring down the exponent? The exponent of ε in Lemma 1 can't be smaller than 2, and the exponent of r in Lemma 3 can't be smaller than 1. Therefore, if now one does a similar calculation as in the proof above, one could hope to get the exponent of n down to $5/6 \approx 0.833$, but no smaller. We haven't been able to do, as yet, but this shows our current analysis cannot come much close to the optimal $n^{1/2}$ result.

Looking ahead, we believe the random path tester may well be a $O(\sqrt{n})$-query monotonicity tester for Boolean functions. But as is with many problems in property testing, the crux lies in the analysis. We think one would need a new idea than those described in this current paper to make this go through. This is the obvious open direction. Finally, there may be a different algorithm which one may think of for 'high' average sensitivity functions. That is also an exciting direction to explore.

4. REFERENCES

[1] N. Ailon and B. Chazelle, *Information theory in property testing and monotonicity testing in higher dimension*, Information and Computation **204** (2006), no. 11, 1704–1717.

[2] N. Ailon, B. Chazelle, S. Comandur, and D. Liu, *Estimating the distance to a monotone function*, Random Structures and Algorithms **31** (2006), no. 3, 1704–1711.

[3] T. Batu, R. Rubinfeld, and P. White, *Fast approximate PCPs for multidimensional bin-packing problems*, Information and Computation **196** (2005), no. 1, 42–56.

[4] A. Bhattacharyya, E. Grigorescu, K. Jung, S. Raskhodnikova, and D. Woodruff, *Transitive-closure spanners*, Proceedings of the 18th Annual Symposium on Discrete Algorithms (SODA), 2009, pp. 531–540.

[5] E. Blais, J. Brody, and K. Matulef, *Property testing lower bounds via communication complexity*, Computational Complexity **21** (2012), no. 2, 311–358.

[6] J. Briët, S. Chakraborty, D. García-Soriano, and A. Matsliah, *Monotonicity testing and shortest-path routing on the cube*, Combinatorica **32** (2012), no. 1, 35–53.

[7] D. Chakrabarty and C. Seshadhri, *Optimal bounds for monotonicity and lipschitz testing over hypercubes and hypergrids.*, Unpublished manuscript. (2012).

[8] Y. Dodis, O. Goldreich, E. Lehman, S. Raskhodnikova, D. Ron, and A. Samorodnitsky, *Improved testing algorithms for monotonicity*, Proceedings of the 3rd International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM) (1999), 97–108.

[9] F. Ergun, S. Kannan, R. Kumar, R. Rubinfeld, and M. Viswanathan, *Spot-checkers*, Journal of Computer Systems and Sciences (JCSS) **60** (2000), no. 3, 717–751.

[10] E. Fischer, *On the strength of comparisons in property testing*, Information and Computation **189** (2004), no. 1, 107–116.

[11] E. Fischer, E. Lehman, I. Newman, S. Raskhodnikova, and R. Rubinfeld, *Monotonicity testing over general poset domains*, Proceedings of the 34th Annual ACM Symposium on the Theory of Computing (STOC) (2002), 474–483.

[12] O. Goldreich, S. Goldwasser, E. Lehman, D. Ron, and A. Samordinsky, *Testing monotonicity*, Combinatorica **20** (2000), 301–337.

[13] S. Halevy and E. Kushilevitz, *Testing monotonicity over graph products*, Random Structures and Algorithms **33** (2008), no. 1, 44–67.

[14] E. Lehman and D. Ron, *On disjoint chains of subsets*, Journal of Combinatorial Theory, Series A **94** (2001), no. 2, 399–404.

[15] N. Linial, Y. Mansour, and N. Nisan, *Constant depth circuits, fourier transform, and learnability*, Journal of the ACM **40** (1993), no. 3.

[16] M. Parnas, D. Ron, and R. Rubinfeld, *Tolerant property testing and distance approximation*, Journal of Computer and System Sciences **6** (2006), no. 72, 1012–1042.

[17] D. Ron, R. Rubinfeld, S. Safra, and O. Weinstein, *Approximating the Influence of Monotone Boolean Functions in $O(\sqrt{n})$ Query Complexity.*, Proceedings of the 15th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM), 2011.