Optimal Bounds for Monotonicity and Lipschitz Testing over Hypercubes and Hypergrids

Deeparnab Chakrabarty
Microsoft Research India
dechakr@microsoft.com

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

ABSTRACT

The problem of monotonicity testing over the hypergrid and its special case, the hypercube, is a classic question in property testing. We are given query access to $f : [k]^n \mapsto \mathbf{R}$ (for some ordered range $\mathbf{R}$). The hypergrid/cube has a natural partial order given by coordinate-wise ordering, denoted by $\prec$. A function is *monotone* if for all pairs $x \prec y$, $f(x) \leq f(y)$. The distance to monotonicity, ε_f, is the minimum fraction of values of f that need to be changed to make f monotone.

For $k = 2$ (the boolean hypercube), the usual tester is the *edge tester*, which checks monotonicity on adjacent pairs of domain points. It is known that the edge tester using $O(\varepsilon^{-1} n \log |\mathbf{R}|)$ samples can distinguish a monotone function from one where $\varepsilon_f > \varepsilon$. On the other hand, the best lower bound for monotonicity testing over general $\mathbf{R}$ is $\Omega(n)$. We resolve this long standing open problem and prove that $O(n/\varepsilon)$ samples suffice for the edge tester. For hypergrids, known testers require $O(\varepsilon^{-1} n \log k \log |\mathbf{R}|)$ samples, while the best known (non-adaptive) lower bound is $\Omega(\varepsilon^{-1} n \log k)$. We give a (non-adaptive) monotonicity tester for hypergrids running in $O(\varepsilon^{-1} n \log k)$ time.

Our techniques lead to optimal property testers (with the same running time) for the natural *Lipschitz property* on hypercubes and hypergrids. (A c-Lipschitz function is one where $|f(x) - f(y)| \leq c\|x - y\|_1$.) In fact, we give a general unified proof for $O(\varepsilon^{-1} n \log k)$-query testers for a class of "bounded-derivative" properties, a class containing both monotonicity and Lipschitz.

Categories and Subject Descriptors

F.2.2 [**Analysis of algorithms and problem complexity**]: Nonnumerical Algorithms and Problems—*Computations on discrete structures*; G.2.1 [**Discrete Mathematics**]: Combinatorics—*Combinatorial algorithms*

General Terms

Theory

STOC '13 June 1-4, 2013, Palo Alto, California, USA.

Keywords

Property Testing, Monotonicity, Lipschitz functions

1. INTRODUCTION

Given query access to a function $f : \mathbf{D} \mapsto \mathbf{R}$ where $\mathbf{D}$ is finite, what can we learn about the properties of f without reading all of f? The field of property testing [24, 19] formalizes this question by dealing with relaxed decision problems. A *property* $\mathcal{P}$ is a subset of all functions; we say that a function f has property $\mathcal{P}$ if $f \in \mathcal{P}$. The distance between f and $\mathcal{P}$, denoted by $\varepsilon_{f,\mathcal{P}}$, is the minimum fraction of places at which f must be changed to have the property $\mathcal{P}$. Formally, $\varepsilon_{f,\mathcal{P}} = \min_{g \in \mathcal{P}} (|\{x | f(x) \neq g(x)\}|/|\mathbf{D}|)$.

Given a parameter $\varepsilon \in (0, 1)$, the classic property testing question is to design a randomized algorithm for the following problem. If $\varepsilon_{f,\mathcal{P}} = 0$ (meaning f has the property), the algorithm must accept with probability $> 2/3$, and if $\varepsilon_{f,\mathcal{P}} > \varepsilon$, it must reject with probability $> 2/3$. If $\varepsilon_{f,\mathcal{P}} \in (0, \varepsilon)$, then any answer is allowed. Such an algorithm is called a *property tester for* $\mathcal{P}$. The quality of a tester is determined by the number of queries it makes, and the running time of the tester. A *one-sided tester* never errs if the function satisfies the property. A *non-adaptive tester* decides all of its queries in advance. In other words, the queries are independent of the answers it receives.

A classic property studied in this framework is *monotonicity*. Typically, one assumes a total order on the range $\mathbf{R}$ (so $\mathbf{R}$ may be assumed to be a subset of the reals), and a partial order $\preceq$ on the domain $\mathbf{D}$. A function f is *monotone* if $f(x) \leq f(y)$ whenever $x \preceq y$.

Our results focus on n-dimensional *hypergrids*; that is, $\mathbf{D} = [k]^n$.[1] Of particular interest is the n-dimensional *hypercube* where $k = 2$ and is often denoted as $\mathbf{D} = \{0, 1\}^n$. The hypergrid/hypercube defines the natural coordinate-wise partial order: $x \preceq y$, iff $\forall i \in [n], x_i \leq y_i$.

Monotonicity has been studied extensively in the past decade [15, 18, 13, 22, 17, 2, 16, 20, 23, 3, 5, 7, 10, 8]. For the hypercube domain, Goldreich et al. [18] introduced the *edge tester*. Let $\mathbf{H}$ be the pairs corresponding to the edges of the hypercube. That is, pairs that differ in precisely one coordinate. The edge tester picks a pair in $\mathbf{H}$ uniformly at random and checks for monotonicity of this

[1] More generally, a hypergrid is defined as $\prod_{i=1}^{n} [k_i]$; all our results extend to the different k_i case but for brevity's sake we will stick to the symmetric case.

pair. When the range is boolean, a classic result of Goldreich et al. [18] is that $O(n/\varepsilon)$ samples suffice to give a bonafide monotonicity tester. Dodis et al. [13] generalize the above result to show that $O(\varepsilon^{-1} n \log |\mathbf{R}|)$ samples suffice for a general range $\mathbf{R}$. In the worst case, $|\mathbf{R}| = 2^n$, and so this gives a $O(n^2/\varepsilon)$-query tester. Briët et al. [10] give an $\Omega(n/\varepsilon)$-lower bound for non-adaptive, one-sided testers, and in a recent breakthrough, Blais, Brody, and Matulef [8] prove that $\Omega(\min(n, |\mathbf{R}|^2))$ samples are required by any tester. (For the boolean range, Fischer et al. [17] give a series of stronger lower bounds for different settings, most notably an $\Omega(\sqrt{n})$ lower bound for one-sided non-adaptive testers.)

It has been an outstanding open problem in property testing (see Question 5 in the Open Problems list from the Bertinoro Workshop [1]) to give an optimal bound for monotonicity testing over the hypercube. We resolve this by showing that the edge tester is indeed optimal (when $|\mathbf{R}| \geq \sqrt{n}$).

THEOREM 1. *The edge tester is an $O(n/\varepsilon)$-query (non-adaptive, one-sided) monotonicity tester for functions $f : \{0,1\}^n \mapsto \mathbf{R}$.*

When the domain is the hypergrid $[k]^n$, Dodis et al [13] give a $O(n \log k \log |\mathbf{R}|/\varepsilon)$-query monotonicity tester. Since $|\mathbf{R}|$ can be as large as k^n, this gives an $O(\varepsilon^{-1} n^2 \log^2 k)$-query tester. In a recent, unpublished result, Blais et al. [9] prove a lower bound of $\Omega(n \log k)$ queries for non-adaptive monotonicity testers (for sufficiently large $\mathbf{R}$).

In this paper, we give a $O(\varepsilon^{-1} n \log k)$-query monotonicity tester on hypergrids that generalizes the edge tester. This tester is also a uniform pair tester, in the sense it defines a set $\mathbf{H}$ of pairs, picks a pair uniformly at random from it, and checks for monotonicity among this pair. The pairs in $\mathbf{H}$ also differ in exactly one coordinate, as in the edge tester. This difference in the coordinate is fixed to be a power of 2. (This is the same structure as previous testers in [13, 7].) Observe that this reduces to the edge tester when $k = 2$.

THEOREM 2. *There exists a non-adaptive, one-sided $O(\varepsilon^{-1} n \log k)$-query monotonicity tester for functions $f : [k]^n \mapsto \mathbf{R}$.*

We discuss some other previous work on monotonicity testers for hypergrids. For the total order (the case $n = 1$), which has been called the monotonicity testing problem on the *line*, Ergün et al [15] give an $O(\varepsilon^{-1} \log k)$-query tester, and this is optimal [15, 16]. Results for general posets were first obtained by Fischer et al [17]. The elegant concept of 2-TC spanners introduced by Bhattacharyya et al [7] give a general class of monotonicity testers for various posets. It is known that such constructions give testers with polynomial dependence of n for the hypergrid [6]. For constant n, Halevy and Kushilevitz [20] give a $O(\varepsilon^{-1} \log k)$-query tester (although the dependency on n is exponential).

Another property that has been studied recently is that of a function being *Lipschitz*: a function $f : [k]^n \mapsto \mathbf{R}$ is called c-Lipschitz if for all $x, y, |f(x) - f(y)| \leq c\|x - y\|_1$. The Lipschitz testing question was introduced by Jha and Raskhodnikova [21], who show that for the range $\mathbf{R} = \delta\mathbb{Z}$, $O(n^2/(\delta\varepsilon))$ queries suffice[2] for Lipschitz testing. They also give a $O(\varepsilon^{-1} \log k)$-query tester for the line. For general hypergrids, Awasthi et al. [4] recently give an $O((\delta\varepsilon)^{-1} n^2 k \log k)$-query tester[3] when the range is $\mathbf{R} = \delta\mathbb{Z}$. As for lower bounds, Jha and Raskhodnikova [21] give a general $\Omega(n)$-query lower bound for the Lipschitz testing question on the hypercube, and an $\Omega(\log k)$-query non-adaptive 1-sided lower bound for that on the line. The recent manuscript by Blais et al. [9] mentioned above also gives an $\Omega(n \log k)$-query lower bound for non-adaptive Lipschitz testers.

Testing the Lipschitz property is a natural and important question that arises in many applications. For instance, given a computer program, one may like to test the sensitivity of the program's output to the input. This has been studied before, for instance in [12], however, the solution provided looks into the code to detect if the program satisfies Lipschitz or not. The property testing setting is a black-box approach to the problem. Jha and Raskhodnikova [21] also provide an application to privacy; a class of mechanisms known as Laplace mechanisms proposed by Dwork et al. [14] achieve privacy in the process of outputting a function by adding a noise proportional to the Lipschitz constant of the function. To find the Lipschitz constant, one typically needs to guess c and test whether the function is c-Lipschitz.

We give a unified tester for the Lipschitz property that improves all known results and matches existing lower bounds. In fact, the testers are the same as that of monotonicity; the pairs are chosen at random from the same set $\mathbf{H}$, and checked for the Lipschitz condition instead of monotonicity. (The tester of Awasthi et al. [4] follows the same settings, but have a different set $\mathbf{H}$.) No non-trivial result was known for general ranges with arbitrarily small δ.

THEOREM 3. *There exists a non-adaptive, one-sided $O(\varepsilon^{-1} n \log k)$-query c-Lipschitz tester for functions $f : [k]^n \mapsto \mathbf{R}$.*

Our techniques apply to property testing of a much larger class of properties that contains monotonicity and Lipschitz. We call it the bounded derivative property, or more technically, the (α, β)-Lipschitz property. Given parameters α, β, with $\alpha < \beta$, we say that a function $f : [k]^n \mapsto \mathbf{R}$ has the (α, β)-Lipschitz property if for any $x \in [k]^n$, and y obtained by increasing exactly one coordinate of x by exactly 1, we have $\alpha \leq f(y) - f(x) \leq \beta$. Note that when $(\alpha = 0, \beta = \infty)$[4], we get monotonicity. When $(\alpha = -c, \beta = +c)$, we get c-Lipschitz. Our tester above can be generalized for the (α, β)-Lipschitz property.

THEOREM 4. *There exists a non-adaptive, one-sided $O(\varepsilon^{-1} n \log k)$-query (α, β)-Lipschitz tester for functions $f : [k]^n \mapsto \mathbf{R}$, for any $\alpha < \beta$. (There is no dependence in the running time on α and β.)*

Although Theorem 4 implies all the other theorems stated above, we will only give a complete proof of Theorem 1 and Theorem 2 in this extended abstract. These are interesting in their own right and illustrate most of the techniques invented. We give a brief outline of the proof for Theorem 4 in §6 and §7, and a full version is available as [11].

[2] They get a tighter bound of $O(nD/(\delta\varepsilon))$, where D is a bound on range of values that f takes.

[3] They get a tighter bound of $O(nD \log D/(\delta\varepsilon))$, where D is a bound on range of values that f takes. [4] also give a 2-sider tester making $O\left(\frac{nk\sqrt{n}\log k}{\delta\varepsilon}\right)$ queries.

[4] If the reader is uncomfortable with the choice of β as ∞, β can be thought of as much larger than any value in f.

2. THE PROOF ROADMAP

The challenge of property testing is to relate the tester behavior to the distance to the property. Consider monotonicity over the hypercube. To argue about the edge tester, we want to show that a large distance to monotonicity implies many violated edges. Most current analyses of the edge tester go via what we could call the *contrapositive route*. If there are few violated edges in f, then they show the distance to monotonicity is small. This is done by modifying f to make it monotone, and bounding the number of changes as a function of the number of violated edges. There is an inherently "constructive" viewpoint to this: it specifies a method to convert non-monotone functions to monotone ones.

Implementing this becomes difficult when the range becomes large, and bounds degrade with $\mathbf{R}$. For the Lipschitz property, this route becomes incredibly complex. A non-constructive approach may give more power, but how does one get a handle on the distance? The *violation graph* provides a method. The violation graph has an edge between any pair of comparable domain vertices (x, y), that is $x \prec y$, if $f(x) > f(y)$. The following theorem can be found as Corollary 2 in [17].

THEOREM 5 ([17]). *The size of the minimum vertex cover of the violation graph is exactly $\varepsilon_f|\mathbf{D}|$. As a corollary, the size of any maximal matching in the violation graph is at least $\frac{1}{2}\varepsilon_f|\mathbf{D}|$.*

Can a large matching in the violated graph imply there are many violated edges? Lehman and Ron [22] give an approach by reducing the monotonicity testing on the hypercube problem to certain routing problems on the hypercube. In particular, they show that if for any k source-sink pairs (corresponding to the endpoints of the maximal matching in the violated graph) on the directed hypercube, at least $k\mu(k)$ edges need to be deleted in order to pairwise separate them, then $O(n/\varepsilon\mu(n))$ queries suffice for the edge tester. Therefore, if $\mu(n)$ is at least a constant, one gets a linear query monotonicity tester on the cube. Lehman and Ron [22] explicitly ask for bounds on $\mu(n)$. Briët et al. [10] showing that $\mu(n)$ could be as small as $\frac{1}{\sqrt{n}}$, thereby putting an $\Omega(n^{3/2}/\varepsilon)$ bottleneck to the above approach.

In the reduction above, however, the function values are altogether ignored. Once one moves to the combinatorial routing question on source-sink pairs, the fact that they are related by actual function values is lost. Our analysis crucially uses the value of the functions to argue about the structure of the maximal matching in the violation graph.

2.1 It's all about matchings

Our proof is intimately connected with the actual function values and is non-constructive. The key insight is to move to a *weighted* violation graph. The weight of violation (x, y) depends on the property at hand; for now it suffices to know that for monotonicity, the weight of (x, y) with $x \prec y$ is $f(x) - f(y)$. This can be thought of as a measure of the magnitude of the violation. (Violation weights were also used for Lipschitz testers [21].) We now look at a maximum *weighted* matching $\mathbf{M}$ in the violation graph. Naturally, this is maximal as well, so we know $|\mathbf{M}| \geq \frac{1}{2}\varepsilon_f|\mathbf{D}|$.

Our testers are uniform pair testers, that is, all our algorithms pick a pair uniformly at random from a predefined set $\mathbf{H}$ of pairs, and check the property on that pair. Our whole analysis is based on the construction of a one-to-one mapping (not quite, but not far from the truth) from pairs in $\mathbf{M}$ to *violating* pairs in $\mathbf{H}$. This mapping implies $|\mathbf{H}| \geq |\mathbf{M}|$, and thus the uniform pair tester succeeds with probability $\Omega(\varepsilon_f|\mathbf{D}|/|\mathbf{H}|)$, implying $O(|\mathbf{H}|/\varepsilon_f|\mathbf{D}|)$ queries suffice.

To obtain this mapping, we first decompose $\mathbf{M}$ into sets $M_1, M_2, \ldots, M_t$ such that each pair in $\mathbf{M}$ is in at least one M_i. Furthermore, we *partition* $\mathbf{H}$ into sets $H_1, H_2, \ldots, H_t$, respectively. M_i's are clearly matchings since $\mathbf{M}$ was one. $\mathbf{H}$ is partitioned so that each of the H_i's are perfect matchings of the domain $\mathbf{D}$. For instance, in the hypercube case, M_i is the collection of pairs in $\mathbf{M}$ whose ith coordinates differ, and H_i is the collection of pairs differing *only* in the ith coordinate; for the hypergrid case, the partitions are more involved.

We map each pair in M_i to a unique violating pair in H_i. For simplicity, let us forget the subscripts and call the matchings M and H. We will assume in this discussion that $M \cap H = \emptyset$. Pairs in the intersection can be easily dealt with. Let us denote the endpoints of M by the set X. We now consider the *alternating paths and cycles* generated by the symmetric difference of $\mathbf{M}$ and H. Note we use $\mathbf{M}$ to generate these, not M. Starting from a point $x \in X$, we *walk* along the alternating objects, beginning with the H-edge. This gives a sequence of vertices, which we call $\mathbf{S}_x$, for each $x \in X$. We terminate this sequence if we ever reach a vertex which is $\mathbf{M}$-unmatched (recall H is a perfect matching), or if we encounter another vertex of X. In this way we get a collection of sequences, and it is not hard to see they are disjoint. A detailed description is given in §3. Our main technical lemma shows that there must exist at least one violating H-pair in each $\mathbf{S}_x$. The mapping is now complete.

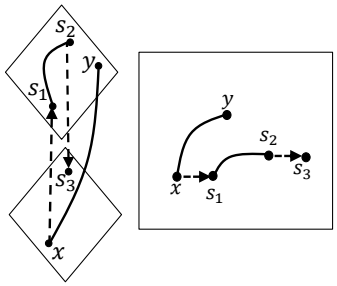

Figure 1: The alternating path: the curved lines connect pairs of M, and the dashed lines are edges of H.

2.2 Getting the violating H-pairs

But why should each alternating path have a violating H-pair? As mentioned earlier, let us focus on monotonicity on the boolean hypercube, so an H-pair is just an edge. Consider M, the pairs of $\mathbf{M}$ which differ on the first coordinate, and H is the set of edges in the dimension cut along this coordinate. Let $(x, y) \in M$, and say $(x)_1 = 0$ giving us $x \prec y$. (We denote the ath coordinate of x by $(x)_a$.) Recall that the weight of this violation is $f(x) - f(y)$. The first step from x (in $\mathbf{S}_x$) leads to s_1. Note that $s_1 \prec y$. Suppose we stopped here because s_1 was $\mathbf{M}$-unmatched. Now for the crucial observation. Delete (x, y) from M and add (s_1, y). If (x, s_1) was not a violation, so $f(x) < f(s_1)$[5], then

[5]We are assuming here that all function values are distinct;

$f(s_1) - f(y) > f(x) - f(y)$. We obtain a new matching with larger weight, contradicting the choice of $\mathbf{M}$. Maybe s_1 was not $\mathbf{M}$-unmatched, but was in X. That is, the matched pair (s_1, s_2) is in M. Observe, however, that if $(s_1, s_2) \in M$, we get $s_1 \succ s_2$. This is because $(s_1)_1 = 1$ (since $(s_1)_1 \neq x_1$) and (s_1, s_2) must differ on the 1st coordinate implying $(s_2)_1 = 0$. Note that $s_2 \prec y$, so we could replace pairs (x, y) and (s_2, s_1) in $\mathbf{M}$ with (s_2, y). Again, if (x, s_1) is not a violation, then $f(s_2) - f(y) > [f(s_2) - f(s_1)] + [f(x) - f(y)]$, contradicting the maximality of $\mathbf{M}$.

With care, this argument can be carried over for longer chains and a description of this is given in §4. Let us demonstrate it a little further. Again, we start with $(x, y) \in M$, and $(x)_1 = 0$. Following the sequence $\mathbf{S}_x$, the first term s_1 is x projected "up" dimension cut H. The second term is obtained by following the M-pair incident to s_1. Suppose it exists, and the other end is s_2. In the next step, s_2 is projected "down" along H to get s_3. Suppose $s_2 \prec s_1$. Then, one can remove (x, y) and (s_1, s_2) and add (x, s_1) and (s_2, y) to increase the matching weight. (We just made the argument earlier; the interested reader may wish to verify.) Hence, $s_2 \succ s_1$, and we get the left part of Fig. 1. Observe that $x \prec y$ and $s_1 \prec s_2$. *By the nature of the dimension cut* H, $x \prec s_3$ *and* $s_1 \prec y$. So, if s_3 is unmatched and (s_2, s_3) is not a violation, we can again rearrange the matching to improve the weight. We alternately go "up" and "down" H_1 in traversing $\mathbf{S}_x$, because of which we can modify the pairs in $\mathbf{M}$ and get other matchings in the violation graph. The maximality of $\mathbf{M}$ imposes additional structure, which leads to violating edges in H.

In general, the spirit of all our arguments is as follows. Take an endpoint of M and start walking along the sequence given by the alternating paths generated by $\mathbf{M}$ and H. Naturally, this sequence must terminate somewhere. If we never encounter a violating pair of H during the entire sequence, then we can rewire the matching $\mathbf{M}$ and increase the weight. Contradiction!

Observe the crucial nature of alternating up and down movements along H. This happens because the first coordinate of the points in $\mathbf{S}_x$ switches between the two values of 0 and 1 (for $k = 2$). Such a reasoning does not hold water in the hypergrid domain. The structure of $\mathbf{H}$ needs to be more complex, and is not as simple as a partition of the edges of the hypergrid. Consider the extreme case of the line $[k]$. Let 2^r be less than k. We break $[k]$ into contiguous pieces of length 2^r. We can now match the first part to the second, the third to the fourth, etc. In other words, the pairs look like $(1, 2^r + 1)$, $(2, 2^r + 2)$, ..., $(2^r, 2^{r+1})$, then $(2^{r+1} + 1, 2^{r+1} + 2^r + 1)$, $(2^{r+1} + 2, 2^{r+1} + 2^r + 2)$, etc. We can construct such matchings for all powers of 2 less than k, and these will be our H_i's. Those familiar with existing proofs for monotonicity on $[k]$ will not be surprised by this set of matchings. All methods need to cover all "scales" from 1 to k (achieved by making them all powers of 2 up to k). It can also be easily generalized to $[k]^n$.

What about the choice of $\mathbf{M}$? Simply choosing $\mathbf{M}$ to be a maximum weight matching and setting up the sequences $\mathbf{S}_x$ does not seem to work. It suffices to look at $[k]^2$ and the matching H along the first coordinate where $r = 0$, so the pairs are $\{(x, x') | (x)_1 = 2i - 1, (x')_1 = 2i, (x)_2 = (x')_2\}$. A good candidate for the corresponding M is the set of pairs in $\mathbf{M}$ that connect lower endpoints of H to higher endpoints of H. Let us now follow $\mathbf{S}_x$ as before. Refer to the right part of Fig. 1. Take $(x, y) \in M$ and let $x \prec y$. We get s_1 by following the H-edge on x, so $s_1 \succ x$. We follow the M-pair incident to s_1 (suppose it exists) to get s_2. We could get $s_2 \succ s_1$. It is in s_3 that we see a change from the hypercube. We could get $s_3 \succ s_2$, because there is no guarantee that s_2 is at the higher end of an H-pair. This could not happen in the hypercube. We could have a situation where s_3 is unmatched, we have not encountered a violation in H, and yet we cannot rearrange $\mathbf{M}$ to increase the weight. For a concrete example, consider $x = (1, 1)$, $y = (4, 3)$, $s_1 = (2, 1)$, $s_2 = (5, 2)$, $s_3 = (6, 2)$ (as in Fig. 1) and $f(x) = f(s_1) = f(s_3) = 1$, $f(y) = f(s_2) = 0$. Some thought leads to the conclusion that s_3 must be less than s_2 for any such rearrangement argument to work.

The road out of this impasse is suggested by the two observations. First, the difference in 1-coordinates between s_1 and s_2 must be odd. Next, we could rearrange and match (x, s_2) and (s_1, y) instead. The weight may not increase, but this matching might be more amenable to the alternating path approach. We could start from a maximum weight matching that also maximizes the number of pairs where coordinate differences are even. Indeed, the major insight for hypergrids is the definition of a *potential* Φ for $\mathbf{M}$, that is a generalization of this idea. The potential Φ is obtained by summing for every pair $(x, y) \in \mathbf{M}$ and every coordinate a, the largest power of 2 dividing the difference $|(x)_a - (y)_a|$. We can show that a maximum weight matching that *also maximizes* Φ does not end up in the bad situation above. With some addition arguments, we can generalize the hypercube proof. We describe this in §5. Observe that the potential with alternating paths give a *unified and optimal* proof for two very "different" hypergrids: the hypercube and the line.

2.3 Attacking the generalized Lipschitz property

One of the challenges in dealing with the Lipschitz property is the lack of direction. The Lipschitz property, defined as $\forall x, y, |f(x) - f(y)| \leq \|x - y\|_1$, is fundamentally an *undirected property*, while monotonicity is directed in nature. In monotonicity, a point x only "interacts" with the subcube above and below x, while in Lipschitz, constraints are defined between all pairs of points. Previous results for Lipschitz testing require very technical and clever machinery to deal with this issue, since arguments analogous to monotonicity just do not work. The alternating paths argument given above for monotonicity also exploits this directionality, as can be seen by heavy use of inequalities in the informal calculations. Observe that in the monotonicity example for hypergrids in Fig. 1, the fact that $s_3 \succ s_2$ (as opposed to $s_3 \prec s_2$) required the potential Φ (and a whole new proof). The (α, β)-Lipschitz property creates even more problems, since constraints are not symmetric between x and y.

A subtle point is that while the property of Lipschitz is undirected, violations to Lipschitz are "directed". If $|f(x) - f(y)| \leq \|x - y\|_1$, then either $f(x) - f(y) > \|x - y\|_1$ or $f(y) - f(x) > \|x - y\|_1$, but never both. This can be interpreted as a direction for violations. In the alternating paths for monotonicity (especially for the hypercube), the partial order relation between successive terms follow a fixed pattern. This is crucial for performing the matching rewiring.

As might be guessed, the weight of a violation (x, y) be-

as we show in Claim 3 this is without loss of generality.

comes $\max(f(x)-f(y)-\|x-y\|_1, f(y)-f(x)-\|x-y\|_1)$. For the generalized Lipschitz problem, this is defined in terms of a pseudo-distance over the domain. We look at the maximum weight matching as before (and use the same potential function Φ). The notion of "direction" takes the place of the partial order relation in monotonicity. The main technical arguments show that these directions follow a fixed pattern in the corresponding alternating paths. Once we have this pattern, with some work we can perform the matching rewiring argument for the generalized Lipschitz problem.

3. ALTERNATING PATHS AND THE SEQUENCE $\mathbf{S}_X$

In this section we formally define the sequences as described in proof roadmap above. We will need three objects: $\mathbf{M}$, the matching of violating pairs, M, one of the parts of the classification of $\mathbf{M}$, and H, a matching of $\mathbf{D}$. Usually, for each M, we will find some appropriate H.

We require the following technical definition. This is necessary for only for the hypergrid proof. This arises because we will use matchings that are not necessarily perfect. Observe that a perfect matching H is always adequate.

DEFINITION 1. *The matching H is* adequate *if for every violation (x, y), both x and y participate in the matching H.*

We will henceforth assume that H is adequate. The symmetric difference of $\mathbf{M}$ and H is a collection of alternating paths and cycles. Let X be the endpoints of M that are present in these. (Note that if a pair in M is actually an edge in H, then the endpoints of M are *not* present in the alternating paths/cycles.) We will denote the set of alternating paths/cycles that contain some vertex of X by $\mathcal{A}$.

We define the sequence $\mathbf{S}_x$ for all $x \in X$ as follows.

1. The first term $\mathbf{S}_x(0)$ is x.
2. For even i, $\mathbf{S}_x(i+1) = H(\mathbf{S}_x(i))$.
3. For odd i: if $\mathbf{S}_x(i) \in X$, or is $\mathbf{M}$-unmatched, then $\mathbf{S}_x$ terminates.
   Otherwise, $\mathbf{S}_x(i+1) = \mathbf{M}(\mathbf{S}_x(i))$.

Note that because H is adequate for $\mathbf{M}$, this sequence is well defined. Indeed, it never terminates at an even term, since $H(\mathbf{S}_x(i))$ always exists. As described in §2, an intuitive way of understanding $\mathbf{S}_x$ is by looking at what happens in $\mathcal{A}$. All the paths/cycles of $\mathcal{A}$ containing points of X can be partitioned into contiguous sequences. Pick any vertex in $x \in X$ and start walking along the H-link incident to it. We stop when we reach a vertex in X. We keep repeating this procedure until all paths/cycles in $\mathcal{A}$ are subpartitioned into the sequences.

Observe that any cycle containing some point of $x \in X$ also contains $\mathbf{M}(x) = M(x) \in X$. Hence, this decomposition breaks the cycle into a collection of paths with the following property. The first and last vertices these paths are in X, and all internal vertices in the path are not in X. The starting and ending edges are in H. (The paths are undirected, so the label of start and end is quite arbitrary.) Every vertex in X is the start or end of some path. The sequence $\mathbf{S}_x$ is simply the ordered list of vertices (starting from x) in the path containing x. The following proposition, whose proof follows from the discussion above, captures basic properties of $\mathbf{S}_x$. We use $T(x)$ to denote the last vertex in $\mathbf{S}_x$. We refer the reader to Fig. 2 for an illustration of the procedure described.

PROPOSITION 1. *Assume that H is adequate for $\mathbf{M}$.*

- *Every $\mathbf{S}_x$ terminates.*
- *$T(x)$ is $\mathbf{M}$-unmatched, or $T(x) \in X$ in which case, $\mathbf{S}_{T(x)}$ is the reverse of $\mathbf{S}_x$ and $T(T(x)) = x$.*
- *For $x, y \in X$, either $y = T(x)$ or $\mathbf{S}_x$ and $\mathbf{S}_y$ are disjoint.*

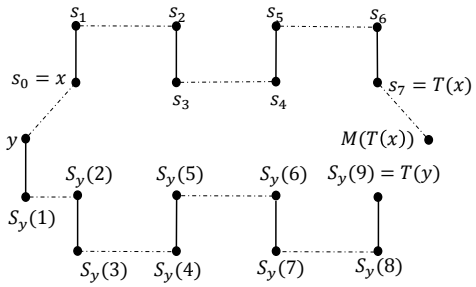

Figure 2: An illustration of the alternating paths. The dashed lines are M-edges; the solid lines are H edges. Consider the edge $(x, y) \in M$. Thus, $x, y \in X$. The top alternating path starting from x is s_i used as a shorthand for $\mathbf{S}_x(i)$. $\mathbf{S}_x$ terminates at $s_7 = T(x)$ since $T(x)$ also lies in X. The alternating path starting from $T(x)$ would be the same as $\mathbf{S}_x$ but reversed, and would end at x. The path starting from y going below is $\mathbf{S}_y$ and ends at $\mathbf{S}_y(9) = T(y)$. This could be because $T(y)$ is M-unmatched. Note that $T(y)$ could also be $M(T(x))$ (in which case, this would be an alternating cycle).

As stated in the introduction, the main part of the proofs will be to show that there exists a violated H-pair in $\mathbf{S}_x$, for all $x \in X$. Note that this would imply $|M| \leq |H|$ since the $\mathbf{S}_x$'s are disjoint. For the sake of contradiction, we assume there is some $\mathbf{S}_c$ without a violated H-pair. With this, the following two lemmas contradict the termination condition of Prop. 1.

LEMMA 1 (PROGRESS LEMMA.). *For odd i, $\mathbf{S}_x(i)$ is not $\mathbf{M}$-unmatched.*

LEMMA 2 (DISJOINTEDNESS LEMMA.). *For odd i, $\mathbf{S}_x(i) \notin X$.*

It is clear from Prop. 1, that the two lemmas imply non-termination. The proofs of both lemmas essentially show that if there are no H-violating pairs in $\mathbf{S}_x$ thus far, and the condition of the lemma didn't hold, then a better $\mathbf{M}$ can be found.

4. MONOTONICITY ON BOOLEAN HYPERCUBE

We prove Theorem 1. The weight of a pair (x, y) is defined to be $f(x) - f(y)$ if $x \prec y$, and is $-\infty$ otherwise. Thus violating pairs have positive weight. We choose a maximum weight matching $\mathbf{M}$ of pairs. Note that every pair in $\mathbf{M}$ is a violating pair. Furthermore $\mathbf{M}$ is also is a maximal family of disjoint violating pairs, and therefore, $|\mathbf{M}| \geq \frac{1}{2}\varepsilon_f \cdot 2^n$. We distribute the pairs in $\mathbf{M}$ into n classes $M_1, \ldots, M_n$. M_i contains the pairs which differ in coordinate i. Note that each pair in $\mathbf{M}$ is in some class.

We denote the set of all edges of the hypercube as $\mathbf{H}$. We partition $\mathbf{H}$ into $H_1, \ldots, H_n$ where H_i is the collection of hypercube edges which differ *only* in the ith coordinate. Note that each H_i is a perfect matching and is hence always adequate. We let $C_i \subseteq H_i$ denote the violating pairs of H_i. The following is the main charging lemma.

LEMMA 3. *For all* $1 \leq r \leq n$, $|M_r| \leq |C_r|$.

The above lemma proves Theorem 1; the probability that the edge-tester succeeds is precisely

$$\frac{1}{|\mathbf{H}|}\sum_{r=1}^{n}|C_r| \geq |\mathbf{M}|/(n2^n) \geq \varepsilon/2n.$$

Therefore, $O(n/\varepsilon)$ queries suffice.

Henceforth, we lose the subscript r. Recall X is the set of endpoints of M. We feed $M, H, \mathbf{M}$ into the machinery of §3 to obtain the sequences $\mathbf{S}_x$ for $x \in X$. Fix $x \in X$, and for brevity's sake we let s_ℓ denote $\mathbf{S}_x(\ell)$ for $\ell \geq 0$. We also let s_{-1} denote $M(x)$. Without loss of generality, assume $x[r] = 0$ (it could be that or 1; the argument in that case is analogous). Recall that for even i, $(s_i, s_{i+1}) \in H$ and for odd i, $(s_i, s_{i+1}) \in \mathbf{M}$. The reader may find it useful to refer to Fig. 2. We start off with the following observation.

PROPOSITION 2. *For* $i \equiv 0, 3 \pmod 4$, $s_i[r] = 0$. *For* $i \equiv 1, 2 \pmod 4$, $s_i[r] = 1$.

PROOF. If i is odd, then $(s_{i-1}, s_i) \in H$. Therefore, $s_i[r] \neq s_{i-1}[r]$. If $i \equiv 1 \pmod 4$, then by induction $s_{i-1}[r] = 0$ and thus $s_i[r] = 1$. $i \equiv 3 \pmod 4$ case is similar. If i is even, then $(s_{i-1}, s_i) \in \mathbf{M}$. Furthermore, $s_{i-1} \notin X$. So, $s_{i-1}[r] = s_i[r]$. A similar argument as above finishes the proof. □

For contradiction's sake, we assume $(s_{i-1}, s_i) \notin C$ for any odd i. Using the above proposition, this implies

$$\begin{aligned} f(s_{i-1}) - f(s_i) > 0, \ \forall i \equiv 3 \pmod 4 \\ f(s_i) - f(s_{i-1}) > 0, \ \forall i \equiv 1 \pmod 4 \end{aligned} \qquad (*)$$

Note the strict inequalities used; this is without loss of generality although we prove it formally later in Claim 3. The above property will now be assumed to hold in the remainder of the proof. In the end we will get a contradiction.

For odd i, the pair (s_i, s_{i+1}) lies in $\mathbf{M}$, but a priori we do not know which of these two is the ancestor and which is the descendant. The following lemma characterizes this.

LEMMA 4. *Let* i *be odd. Then,*

$$\forall i \equiv 1 \pmod 4, \ s_{i+1} \succ s_i; \quad \forall i \equiv 3 \pmod 4, \ s_i \succ s_{i+1}$$

PROOF. The proof is by induction on i. Assume the claim is true for all odd $j < i$, for some $i \equiv 1 \pmod 4$. The proof for the other case is similar. By construction, the base case ($i = -1$) can be checked to hold true. Suppose for contradiction, $s_i \succ s_{i+1}$. We now construct a matching $\mathbf{M}'$ of larger weight than $\mathbf{M}$ as follows. Delete the set of $\mathbf{M}$-edges $E_- := \{(s_j, s_{j+1}) : j \text{ odd }, -1 \leq j \leq i\}$, and add the set of edges $E_+ :=$

$$(s_{-1}, s_1) \cup \{(s_{j-1}, s_{j+2}) : j \text{ odd }, 1 \leq j \leq i-4\} \cup (s_{i-3}, s_{i+1})$$

Check that $\mathbf{M} - E_- + E_+$ is a valid matching which leaves s_i, s_{i-1} unmatched. Now we consider the weights. The weight of E_-, by induction, is

$$\begin{aligned} W_- = \ & [f(s_0) - f(s_{-1})] + [f(s_1) - f(s_2)] + [f(s_4) - f(s_3)] \\ & + \cdots + [f(s_{i-1}) - f(s_{i-2})] + [f(s_{i+1}) - f(s_i)] \end{aligned}$$

Observe the signs changing from term to term due to induction hypothesis, except for the last term which is assumed for the sake of contradiction. Also by induction, note that $(s_{j-1}, s_{j+2}) = (s_j \oplus e_r, s_{j+1} \oplus e_r)$, where e_r is a vector with either $+1$ or -1 on the rth coordinate and 0 everywhere else, and $\oplus$ is the coordinate wise sum operator. This is because (s_{j-1}, s_j) and (s_{j+1}, s_{j+2}) are both in H, and it suffices to show that the rth coordinates of s_{j-1} and s_{j+1} are different. This follows from Prop. 2. Therefore, when $j \equiv 3 \pmod 4$, and therefore by induction, $s_j \succ s_{j+1}$, we have $s_{j-1} \succ s_{j+2}$ Similarly, when $j \equiv 1 \pmod 4$, $s_{j-1} \prec s_{j+2}$. We get that whenever $1 \leq j \equiv 1 \pmod 4$, $s_{j+2} \succ s_{j-1}$ and whenever $(i-2) \geq j \equiv 3 \pmod 4$, $s_{j-1} \succ s_{j+2}$. By the assumption, we get $s_{i-3} \succ s_i \succ s_{i+1}$. In particular, $s_{i-3} \succ s_i$ which in turn, for the sake of contradiction, we have assumed $\succ s_{i-1}$. Using this, we get the weight of E_+ is precisely

$$\begin{aligned} W_+ \ = \ & [f(s_1) - f(s_{-1})] + [f(s_0) - f(s_3)] + \cdots \\ & + [f(s_{i-5}) - f(s_{i-2})] + [f(s_i) - f(s_{i-3})] \end{aligned}$$

Thus, we get the weight of the new matching is precisely $w(\mathbf{M}) - W_- + W_+ = w(\mathbf{M}) + f(s_i) - f(s_{i-1})$. By $(*)$, we get that $f(s_i) > f(s_{i-1})$ contradicting the maximality of $\mathbf{M}$. □

Armed with this handle on the ancestor-descendant relationships, we prove the progress and disjointedness lemmas alluded to in §3.

LEMMA 5. *For odd* i, s_i *is not* $\mathbf{M}$*-unmatched.*

PROOF. Suppose it is. Then as in the proof of the previous lemma we can find a better matching. Once again, assume $i \equiv 1 \pmod 4$. We delete the set of edges $E_- := \{(s_j, s_{j+1}) : j \text{ odd }, -1 \leq j \leq i-2\}$ and add the set of edges $E_+ = (s_{-1}, s_1) \cup \{(s_{j-1}, s_{j+2}) : j \text{ odd }, 1 \leq j \leq i-3\}$. Lemma 4 shows that $\mathbf{M} - E_- + E_+$ is a valid matching whose weight is, as before, $w(\mathbf{M}) + f(s_i) - f(s_{i-1}) > w(\mathbf{M})$ by $(*)$. □

LEMMA 6. *For odd* i, $s_i \notin X$.

PROOF. This is really just a corollary of Lemma 4. Suppose $i \equiv 1 \pmod 4$. Then, by Prop. 2, $s_i[r] = 1$. By Lemma 4, $\mathbf{M}(s_i) = s_{i+1} \succ s_i$, and so $s_{i+1}[r] = 1$. Hence, $(s_i, s_{i+1}) \notin M$, and thus $s_i \notin X_r$. If $i \equiv 3 \pmod 4$, then $s_i[r] = 0$ and $\mathbf{M}(s_i) \prec s_i$. Again, $s_i \notin X_r$. □

We conclude that for any $x \in X$, if Condition $(*)$ holds, then $\mathbf{S}_x$ can never terminate. The non-termination contradictions Prop. 1, and therefore $(*)$ must be violated. Therfore, we can find a violated edge in each $\mathbf{S}_x$. The number of such sequences is at least $|M|$; each endpoint leads to a sequence, and at most two sequences collide (Prop. 1). The number of endpoints is $2|M|$. This ends the proof of Lemma 3 , and thus, the proof of Theorem 1.

5. MONOTONICITY ON HYPERGRIDS

In this section, we prove Theorem 2. Let's recall the tester. We define $\mathbf{H}$ to be pairs that differ in exactly one coordinate, and furthermore, the difference is a power of 2. The tester chooses a pair in $\mathbf{H}$ uniformly at random, and checks for monotonicity among this pair. We describe the partition of $\mathbf{H}$ and make some (unimportant) technical arguments allowing for a more convenient setting.

5.1 The partition of H and the issue of adequacy

Suppose we had a domain $[k]^n$, where $\ell = \lceil \lg k \rceil$. For this domain, we partition $\mathbf{H}$ into $2n(\ell+1)$ sets $H^0_{a,b}$ and $H^1_{a,b}$, $1 \le a \le n$, $0 \le b \le \ell$. $H_{a,b}$ consists of pairs (x, y) which differ only in the ath coordinate, and furthermore $|y[a] - x[a]| = 2^b$.

For a pair $(x, y) \in H_{a,b}$, exactly one among $x[a] \pmod{2^{b+1}}$[6] and $y[a] \pmod{2^{b+1}}$ is $> 2^b$ and one is $\le 2^b$. This is because $|y[a] - x[a]| \pmod{2^{b+1}} = 2^b$ (since 2^b divides the difference but 2^{b+1} does not). We put $(x, y) \in H_{a,b}$ (with $x \prec y$) in the set $H^0_{a,b}$ if $y[a] \pmod{2^{b+1}} > 2^b$, and in the set $H^1_{a,b}$ if $1 \le y[a] \pmod{2^{b+1}} \le 2^b$.

Note that each $H^0_{a,b}$ and $H^1_{a,b}$ are matchings. However, some may not be perfect matchings (consider the matching $H^1_{1,0}$). This technicality forced us to introduce the notion of adequacy of matchings. We will eventually prove the following theorem.

THEOREM 6. *Let k be a power of 2. Suppose for every violation (x, y) and every coordinate a, $|y[a] - x[a]| \le 2^c$ (for some c). Furthermore, suppose that for $b \le c$, all matchings $H^0_{a,b}, H^1_{a,b}$ are adequate. Then there exists a maximal matching $\mathbf{M}$ of the violation graph such that the number of violating pairs in $\mathbf{H}$ is at least $|\mathbf{M}|$.*

First, we reduce the general case to this special case using a simple padding argument. Note that the following theorem implies Theorem 2. Proof given in full version. Henceforth, we will assume that $k = 2^\ell$ and that all matchings $H^0_{a,b}, H^1_{a,b}$ are adequate (for $b \le c$, where 2^c is an upper bound on the coordinate difference for any violation).

THEOREM 7. *Consider any function $f : [k]^n \mapsto R$. At least an $\varepsilon_f/(2n(\lceil \log k \rceil + 1))$-fraction of pairs in $\mathbf{H}$ are violations.*

5.2 The potential Φ

As in the hypercube case, the weight of a pair (x, y) is defined to be $f(x) - f(y)$ if $x \prec y$, and $-\infty$ otherwise. We will now pick $\mathbf{M}$ to be a maximum weight matching, as in the hypercube case, however, we will need this matching to have certain other properties as well. To that end, let's define $\mathtt{msd}(a)$ of a non-negative integer a to be the largest power of 2 which divides a. That is, $\mathtt{msd}(a) = p$ implies $2^p \mid a$ but $2^{p+1} \nmid a$. We define $\mathtt{msd}(0) := \ell + 1$. Now given a matching $\mathbf{M}$, define the following potential.

$$\Phi(\mathbf{M}) := \sum_{(x,y)\in\mathbf{M}} \sum_{i=1}^{n} \mathtt{msd}\left(|y_i - x_i|\right). \tag{1}$$

Now choose $\mathbf{M}$ to be the maximum weighted matching which maximizes $\Phi(\mathbf{M})$. As before, since $\mathbf{M}$ is a maximal set of violated pairs, we get $|\mathbf{M}| \ge \frac{1}{2}\varepsilon_f k^n$. To give some intuition for the potential, note that it is aligned towards picking pairs which differ in as few coordinates as possible (since $\mathtt{msd}(0)$ is large). Furthermore, divisibility by power of 2 is favored because the tester queries pairs which are 'powers of 2 apart'.

We distribute $\mathbf{M}$ into $2n(\ell+1)$ classes as we did for $\mathbf{H}$: $M^0_{a,b}$ and $M^1_{a,b}$, for $1 \le a \le n$ and $0 \le b \le \ell$. $M_{a,b} := M^0_{a,b} \cup M^1_{a,b}$ contains pairs $(x, y) \in \mathbf{M}$ which differ in the ath coordinate, and furthermore $\mathtt{msd}(|y[a] - x[a]|) = b$. Note that every pair in $\mathbf{M}$ lies in at least one of the classes $M_{a,b}$. We put $(x, y) \in M^0_{a,b}$ if $y[a] \pmod{2^{b+1}} > 2^b$, and in $M^1_{a,b}$ if $1 \le y[a] \pmod{2^{b+1}} \le 2^b$. Note that if $(x, y) \in M^0_{a,b}$, then $x \prec y$. In summary, we divide $\mathbf{M}$ into classes based on which dimensions they differ in, the $\mathtt{msd}$ of the length, and the 'parity' of the endpoints.

We let $C^r_{a,b}$ be the violated pairs in $H^r_{a,b}$ for $r \in \{0, 1\}$, $1 \le a \le n$, $0 \le b \le \ell$. The following lemma is key.

LEMMA 7. *For all r, a, b, we have $|M^r_{a,b}| \le |C^r_{a,b}|$.*

By our assumptions, if $|M^r_{a,b}| > 0$, then $H^r_{a,b}$ for $r \in \{0, 1\}$ is adequate. This lemma directly implies Theorem 6.

5.3 The proof of Lemma 7

Let's fix a, b. Suppose $r = 0$ (the other case is analogous and omitted). Keeping this in mind, we now lose all superscripts and subscripts. As before, X is the set of endpoints of M. Since H is adequate, we can feed $M, H, \mathbf{M}$ to the machinery of §3 to obtain the sequences $\mathbf{S}_x$ for all $x \in X$. Fix $x \in X$. Wlog, assume $x \prec y$ in the pair $(x, y) \in M^0_{a,b}$. Note that by the assumption $r = 0$, $y[a] \pmod{2^{b+1}} > 2^b$. Since $\mathtt{msd}(y[a] - x[a]) = b$, $(y[a] - x[a]) \pmod{2^{b+1}} = 2^b$. Therefore, $x[a] \pmod{2^{b+1}} \le 2^b$.

we will use s_t to denote $\mathbf{S}_x(t)$. we will use s_{-1} to denote y. we will also abuse notation to let s_i to sometimes denote $s_i[a]$; we hope the context will disabuse. Recall that for even i, $(s_i, s_{i+1}) \in H$ while for odd i, $(s_i, s_{i+1}) \in \mathbf{M}$.

Henceforth, for contradiction's sake we will assume that $(s_i, s_{i+1}) \notin C$ for any even i. We will show that $\mathbf{S}_x$ cannot terminate in that case. The following lemma is captures the structure of the neighboring pairs in $\mathbf{S}_x$ if there are no violating pairs. We would like to point out that the lemma below is more involved than Lemma 4. The reason is that there is no easy analog to Prop. 2. This relates to what we mentioned in the introduction, that is, in the hypercube, if (x, x') is an edge across the rth dimension, then $x_r = 0$ implies $x'_r = 1$. For hypergrids that is not true. In fact, we will need the extra "Φ-maximizing" property of the matchings for the lemma to go through.

LEMMA 8. *If $i \equiv 1 \pmod 4$: **(i)** $s_i \succ s_{i-1}$, **(ii)** $s_{i+1} \succ s_i$, **(iii)** $s_i \pmod{2^{b+1}} > 2^b$, **(iv)** $s_{i+1} \pmod{2^{b+1}} > 2^b$.*

*If $i \equiv 3 \pmod 4$: **(v)** $s_i \prec s_{i-1}$, **(vi)** $s_{i+1} \prec s_i$, **(vii)** $s_i \pmod{2^{b+1}} \le 2^b$, **(viii)** $s_{i+1} \pmod{2^{b+1}} \le 2^b$. s*

PROOF. The proof is by induction, and is similar to Lemma 4 with some crucial differences in part **(i)**. For parts **(iv)** and **(viii)** we will assume $s_{i+1} = M(s_i)$ exists.

(i) The base case of $i = 1$ follows since we have assumed $r = 0$, and therefore as argued above, $s_0 = x[a] \pmod{2^{b+1}} \le 2^b$. Suppose for some $i \equiv 1 \pmod 4$ we get $s_i \prec s_{i-1}$ and for all $j < i$ the lemma is indeed true. Since $(s_{i-1}, s_i) \in H$, $s_{i-1} \succ s_i$ implies $s_{i-1} \pmod{2^{b+1}} > 2^b$ (recall $r = 0$). We now exhibit a different matching $\mathbf{M}'$ with larger $\Phi()$ value, contradicting the choice of $\mathbf{M}$.

Define the set of edges: $E_- := \{(s_j, s_{j+1}) : j \text{ odd}, -1 \le j < i\}$, and $E_+ := (s_{-1}, s_1) \cup \{(s_{j-1}, s_{j+2}) : j \text{ odd}, 1 \le j \le i-4\} \cup (s_{i-1}, s_{i-3})$. By induction, for $1 \le j \le i-4$, the pair (s_{j-1}, s_{j+2}) is precisely $(s_j \oplus e_a, s_{j+1} \oplus e_a)$. Here, $\oplus$ is the

[6] We abuse notation and define $p \pmod{2^{b+1}}$ to be 2^{b+1} (instead of 0) if $2^{b+1} \mid p$.

coordinate-wise sum, and e_a is a vector with 0's on all coordinates but a, and either $+2^b$ or -2^b on the ath coordinate depending on whether $j \equiv 3 \pmod 4$ or $1 \pmod 4$, respectively. Note that this argument requires s_j and s_{j+1} to have the same $\pmod{2^{b+1}}$, which is guaranteed by the induction hypothesis. Since (s_j, s_{j+1}) was in $\mathbf{M}$, the pair (s_{j-1}, s_{j+2}) is a valid pair as well. Furthermore, if $s_j \succ s_{j+1}$, then $s_{j-1} \succ s_{j+2}$, and a similar statement is true with $\prec$ replacing $\succ$.

The above shows that we can swap E_- by E_+ from $\mathbf{M}$ to get $\mathbf{M}'$ without changing the matched end points. Now for the weights. The weight of E_-, by induction, is $W_- = [f(s_0) - f(s_{-1})] + [f(s_1) - f(s_2)] + [f(s_4) - f(s_3)] + \cdots + [f(s_{i-1}) - f(s_{i-2})]$. Observe the signs changing from term to term due to induction hypothesis. Similarly, we get $w(E_+) = [f(s_1) - f(s_{-1})] + [f(s_0) - f(s_3)] + \cdots + [f(s_{i-5}) - f(s_{i-2})] + [f(s_{i-1}) - f(s_{i-3})]$. Therefore, $w(E_-) = w(E_+)$ and $w(\mathbf{M}') = w(\mathbf{M})$.

Note, for odd $1 \le j \le i-4$, we have $|s_{j+1} - s_j| = |s_{j+2} - s_{j-1}|$. Thus, the only pairs which affect Φ are the pairs $(s_{-1}, s_0), (s_{i-1}, s_{i-2})$ in $\mathbf{M}$ and $(s_{-1}, s_1), (s_{i-1}, s_{i-3})$ in $\mathbf{M}'$. The following claim proves that if $s_i \prec s_{i-1}$, then $\Phi(\mathbf{M}') > \Phi(\mathbf{M})$.

Claim 1. $\mathtt{msd}(|s_{-1} - s_1|) + \mathtt{msd}(|s_{i-1} - s_{i-3}|) > \mathtt{msd}(|s_{-1} - s_0|) + \mathtt{msd}(|s_{i-1} - s_{i-2}|)$.

Proof. Note that $\mathtt{msd}(s_{-1} - s_0) = b$, by definition, since it lies in $M^0_{a,b}$. Furthermore, $s_1 = s_0 + 2^b$. The following easy observation implies that $\mathtt{msd}(s_{-1} - s_1) \ge b + 1$.

Observation 1. *For integers b, p, if $2^b \mid p$ and $2^{b+1} \nmid p$, then $2^{b+1} \mid p \pm 2^b$.*

Now we show that $\mathtt{msd}(|s_{i-1} - s_{i-3}|) \ge \mathtt{msd}(|s_{i-1} - s_{i-2}|)$. Let the RHS be b'. Note that $s_{i-3} - s_{i-1} = s_{i-2} - s_{i-1} + 2^b$, since by induction $s_{i-2} \pmod{2^{b+1}} \le 2^b$, and $(s_{i-3}, s_{i-2}) \in H$. Thus, if $b' \le b$, then $2^{b'} \mid |s_{i-2} - s_{i-1}|$ implies $2^{b'} \mid |s_{i-3} - s_{i-1}|$ as well. Thus, it suffices to show $b' \le b$. Suppose not, and $b' \ge b + 1$. This implies $2^{b+1} \mid (s_{i-2} - s_{i-1})$. By induction, we get that $s_{i-2} \pmod{2^{b+1}} \le 2^b$. By supposition, we have $s_{i-1} \pmod{2^{b+1}} > 2^b$. Contradiction. □

(ii) Suppose for some $i \equiv 1 \pmod 4$ we get $s_i \succ s_{i+1}$ and for all $j < i$ the lemma is indeed true. Delete the set of $\mathbf{M}$-edges $E_- := \{(s_j, s_{j+1}) : j \text{ odd }, -1 \le j \le i\}$, and add the set of edges $E_+ := (s_{-1}, s_1) \cup \{(s_{j-1}, s_{j+2}) : j \text{ odd }, 1 \le j \le i-4\} \cup (s_{i-3}, s_{i+1})$. As in the above case, check that $\mathbf{M} - E_- + E_+$ is a valid matching (this uses the induction hypothesis, as above) which leaves s_i, s_{i-1} unmatched. Now we consider the weights. The weight of E_-, by induction, is

$$W_- = [f(s_0) - f(s_{-1})] + [f(s_1) - f(s_2)] + [f(s_4) - f(s_3)] + \cdots + [f(s_{i-1}) - f(s_{i-2})] + [f(s_{i+1}) - f(s_i)]$$

Observe the signs changing from term to term due to induction hypothesis, except for the last term which is assumed for the sake of contradiction. Similar to the previous case, we get $w(E_+) =: W_+ = [f(s_1) - f(s_{-1})] + [f(s_0) - f(s_3)] + \cdots + [f(s_{i-5}) - f(s_{i-2})] + [f(s_i) - f(s_{i-3})]$. Thus, we get the weight of the new matching is precisely $w(\mathbf{M}) - W_- + W_+ = w(\mathbf{M}) + f(s_i) - f(s_{i-1})$. We proved in **(i)** that $s_i \succ s_{i-1}$, and since (s_{i-1}, s_i) is not a violation, we get $f(s_i) > f(s_{i-1})$ contradicting the maximality of $\mathbf{M}$.

(iii) For $i \equiv 1 \pmod 4$, we have $(s_{i-1}, s_i) \in H$. We have proved in **(i)** that $s_i \succ s_{i-1}$. Therefore (since $r = 0$), $s_i \pmod{2^{b+1}} > 2^b$.

We now note that parts **(v)**, **(vi)**, and **(vii)** can be proved similarly as the three cases above. We do not repeat them here. We now show **(iv)** and **(viii)** follow as easy corollaries.

(iv),**(viii)** For $i \equiv 1 \pmod 4$, if $s_{i+1} \pmod{2^{b+1}} \le 2^b$, then $s_{i+2} \succ s_{i+1}$, which contradicts **(v)** for $i + 2 \equiv 3 \pmod 4$. For $i \equiv 3 \pmod 4$, if $s_{i+1} \pmod{2^{b+1}} > 2^b$, then $s_{i+1} \succ s_{i+2}$ which contradicts **(i)** for $i + 2 \equiv 1 \pmod 4$.

Armed with this handle on the ancestor-descendant relationships, we can prove the progress and disjointedness lemmas alluded to in §3.

Lemma 9. *For odd i, s_i is not $\mathbf{M}$-unmatched.*

Proof. Suppose it is. Then as in the proof of the previous lemma we can find a better matching. Once again, assume $i \equiv 1 \pmod 4$ (leaving the other case out since it is analogous). We delete the set of edges $E_- := \{(s_j, s_{j+1}) : j \text{ odd }, -1 \le j \le i-2\}$ and add the set of edges $E_+ = (s_{-1}, s_1) \cup \{(s_{j-1}, s_{j+2}) : j \text{ odd }, 1 \le j \le i-3\}$. As in the above lemma, we get that $\mathbf{M} - E_- + E_+$ is a valid matching whose weight is, as before, $w(\mathbf{M}) + f(s_i) - f(s_{i-1}) > w(\mathbf{M})$ since $s_i \succ s_{i-1}$ and we have assumed there are violated pairs (s_{i-1}, s_i). □

Lemma 10. *For odd i, $s_i \notin X$.*

Proof. We need to show that $(s_i, s_{i+1}) \notin M$. Suppose $i \equiv 1 \pmod 4$ (the other case has the same argument). Lemma 8 **(iii)**, **(iv)** shows that both $s_i \pmod{2^{b+1}}$ and $s_{i+1} \pmod{2^{b+1}}$ are $> 2^b$. Now, if the remainders are same then $2^{b+1} \mid |s_i - s_{i+1}|$ implying $\mathtt{msd}(|s_i - s_{i+1}|) > b$ which in turn implies $(s_i, s_{i+1}) \notin M$. If the remainders are not same, then $|s_i - s_{i+1}| < 2^b$. This implies that $2^b \nmid |s_i - s_{i+1}|$ which again implies $(s_i, s_{i+1}) \notin M$. □

We conclude that for any $x \in X$, if no $(s_i, s_{i+1}) \in C$ for even i, then $\mathbf{S}_x$ can never terminate. The non-termination contradictions Prop. 1, and therefore our supposition must be wrong. This ends the proof of Lemma 7 , and thus, the proof of Theorem 2.

6. A PSEUDO-DISTANCE FOR $\mathcal{L}_{\alpha,\beta}$

A key concept that leads to the unification of Lipschitz and monotonicity is a pseudo-distance defined on $\mathbf{D}$. This distance provides a lot of power in manipulating the alternating paths for more general properties. The monotonicity proof requires no distance, so generalizing it for Lipschitz properties is quite non-trivial. An important feature of the distance is the triangle inequality. The challenge faced in the final proof is tweezing out all the places in the previous argument where the distance function is "hidden". This involves replacing many equalities in the monotonicity argument with inequalities (going the "right way") based on the triangle inequality of this distance.

We begin by defining a weighted directed graph $\mathsf{G} = (\mathbf{D}, E)$ where $\mathbf{D}$ in this section is the hypergrid $[k]^n$. E contains directed edges of the form (x, y), where $\|x - y\|_1 = 1$. The length of edge (x, y) is gives as follows. If $x \prec y$, the length is $-\alpha$. If $x \succ y$, the length is β.

DEFINITION 2. *The function* $\mathsf{d}(x,y)$ *between* $x,y \in \mathbf{D}$ *is the shortest path length from* x *to* y *in* G.

This function is asymmetric, meaning that $\mathsf{d}(x,y)$ and $\mathsf{d}(y,x)$ are possibly different. Furthermore, $\mathsf{d}(x,y)$ can be negative, so this does not truly qualify to be a distance (in the usual parlance of metrics). Nonetheless, $\mathsf{d}(x,y)$ has many useful properties, which can be proven by expressing it in a more convenient form. Given any $x,y \in \mathbf{D}$, we define $\mathsf{hcd}(x,y)$ to be the $z \in \mathbf{D}$ maximizing $||z||_1$ such that $x \succ z$ and $y \succ z$. That is, z is the highest common descendant of x and y. Note that if $x \succ y$ then $\mathsf{hcd}(x,y) = y$.

The proofs for this section are deferred to the full version. They are mostly mechanical and do not provide much insight.

CLAIM 2. *For any* $x,y \in \mathbf{D}$, $\mathsf{d}(x,y) = \beta||x - \mathsf{hcd}(x,y)||_1 - \alpha||y - \mathsf{hcd}(x,y)||_1$.

It is instructive to keep in mind what this distance translates to in the case of monotonicity and Lipschitz. In the case of monotonicity (when $\alpha = 0, \beta = \infty$), we get $\mathsf{d}(x,y) = \infty$ unless $x \prec y$ in which case $\mathsf{d}(x,y) = 0$. In the case of Lipschitz, the distance $d(x,y)$ is precisely the Hamming distance $\mathsf{d}(x,y) = ||x-y||_1$.

The following lemma connects the distance to the property $\mathcal{L}_{\alpha,\beta}$.

LEMMA 11. *A function is* (α,β)*-Lipschitz iff for all* $x,y \in \mathbf{D}$, $f(x) - f(y) - \mathsf{d}(x,y) \le 0$.

The next lemma is a generalization of Theorem 5 which argued that the size of a minimum vertex cover is exactly $\varepsilon_f 2^n$. A similar statement is also known for the Lipschitz property, and we prove this for generalized Lipschitz functions. We crucially use the triangle inequality for $\mathsf{d}(x,y)$.

We define an undirected weighted clique on $\mathbf{D}$. Given a function f, we define the weight $w(x,y)$ (for any $x,y \in \mathbf{D}$) as follows:

$$w(x,y) := \max\Big(f(x)-f(y)-\mathsf{d}(x,y),\ f(y)-f(x)-\mathsf{d}(y,x)\Big) \tag{2}$$

Note that although the distance d is asymmetric, the weights are defined on an undirected graph. Lemma 11 shows that a function is (α,β)-Lipschitz iff all $w(x,y) \le 0$. Once again, it is instructive to understand the special cases of monotonicity and Lipschitz. For monotonicity, we get that $w(x,y) = f(x) - f(y)$ when $x \prec y$ and $-\infty$ otherwise. For Lipschitz, we get $w(x,y) = |f(x) - f(y)| - ||x-y||_1$.

We define the *violation graph* as $VG_f = (\mathbf{D}, V_f)$ where $V_f = \{(x,y) : w(x,y) > 0\}$. The violation graph is unweighted. The following lemma generalizes Theorem 5 from [17].

LEMMA 12. *The size of a minimum vertex cover in* VG_f *is exactly* $\varepsilon_f|\mathbf{D}|$.

Below, we make a technical claim that allows for easier arguments about w. Essentially, by a perturbation argument, we can assume that $w(x,y)$ is never exactly zero. Note that this justifies the strict inequalities we have encountered so far.

CLAIM 3. *For any function* f*, there exists a function* f' *with the following properties. Both* f *and* f' *have the same number of violated edges,* $\varepsilon_f = \varepsilon_{f'}$*, and for all* $x,y \in \mathbf{D}$, $w_{f'}(x,y) \neq 0$.

7. GENERALIZED LIPSCHITZ TESTING ON HYPERGRIDS

Because of space considerations, we only provide a general outline. Intuitively, with the distance $\mathsf{d}(x,y)$ in place, the basic spirit of the monotonicity proofs can be carried over. The final proof, however, is much more complex and requires many algebraic manipulations. We do not explicitly have the "directed" behavior of monotonicity that allows for many of rewiring arguments to be performed. The properties of the distance provide the tools to rewire the matchings.

We borrow the definition of $\mathbf{H}$ from §5; the generalized Lipschitz tester picks a pair $(x,y) \in \mathbf{H}$ at random, with say $x \succ y$, and checks if $\alpha||x-y||_1 \le f(x) - f(y) \le \beta||x-y||_1$. The weight of an edge (x,y) is as defined (2). As in §5, we choose $\mathbf{M}$ to be the maximum weight matching which maximizes $\Phi(M)$ as defined by (1). The classification of $\mathbf{M}$, and partition of $\mathbf{H}$ into $2n(\ell+1)$ classes, are also borrowed from §5, and in the remainder of the section we will prove Lemma 7 in the context of generalized Lipschitz.

As in the proof of Lemma 7, we focus on the case of $r = 0$ and fix a,b. We lose the sub/superscripts and feed $M, H, \mathbf{M}$ into the machinery of §3 to obtain the sequences $\mathbf{S}_x$ for all $x \in X$, the endpoints of M. We fix $x \in X$ and assume $x[a] \pmod{2^{b+1}} \le 2^b$. We let s_t denote $\mathbf{S}_x(t)$, and also at times abusing notation, let it denote $s_t[a]$. We restate that for even i, $(s_i, s_{i+1}) \in H$; for odd i, $(s_i, s_{i+1}) \in \mathbf{M}$. For contradiction's sake, we assume for all even i, (s_i, s_{i+1}) satisfies the generalized Lipschitz property. That is, if $s_i \succ s_{i+1}$, then $\alpha 2^b < f(s_i) - f(s_{i+1}) < \beta 2^b$, else the inequalities are reversed. Note that we have strict inequalities; this follows from Claim 3. Till now, we have just mimicked what we had done in §5. However, the inherent directionality in the monotonicity case led to simple weights and therefore (in comparison what is to follow) simpler (to read, at least) proofs. The proof of Lemma 7 for generalized Lipschitz is quite involved, and needs some notation.

Notation. Let $y = M(x)$. We denote y by s_{-1} as well. The weight $w(y,x)$ is given by $\max(f(x) - f(y) - \mathsf{d}(x,y), f(y) - f(x) - \mathsf{d}(y,x))$. To abstract out these two cases cleanly, we define the following.

- Order dependent functions d_{-1} and d_1: $\mathsf{d}_1(x,y) = \mathsf{d}(x,y)$ and $\mathsf{d}_{-1}(x,y) = \mathsf{d}(y,x)$.
- The marker bit $\mathfrak{b}$: If $w(y,x) = f(y) - f(x) - \mathsf{d}(y,x)$, then $\mathfrak{b} = 1$. Otherwise, it is -1.
- The function $\sigma(y,x,\mathfrak{b}) := \mathfrak{b}(f(y) - f(x)) - \mathsf{d}_{\mathfrak{b}}(y,x)$.
- Indicator μ_i which is 1 if $i \equiv 1 \pmod 4$ and 0 if $i \equiv 3 \pmod 4$.

Note that for any two points $u,v \in [k]^n$, $w(u,v) \ge \sigma(u,v,b)$ for any $b \in \{-1,+1\}$. Also note that $w(y,x) = \mathfrak{b}(f(y) - f(x)) + \mathsf{d}_{\mathfrak{b}}(y,x) = \sigma(y,x,\mathfrak{b})$. The marker bit introduces the "direction" in the pairs on $\mathbf{M}$.

As in the monotonicity case, we need to understand the weights of the pairs of $\mathbf{M}$ in $\mathbf{S}_x$. For odd i, we know that $w(s_i, s_{i+1})$ is $\max(\sigma(s_i, s_{i+1}, -1), \sigma(s_i, s_{i+1}, 1))$, but which value does it take? To execute the argument described above, we need to know this. It turns out that this is exactly decided by μ_i, and therefore has a very consistent behavior. This is the real workhorse of the proof, and brings out the directionality required for our rewiring. The following lemma is analogous to Lemma 4 in §4 and Lemma 8 in §5.

LEMMA 13. *For odd i, suppose $s_{i+1} = M(s_i)$ is defined. Then,*

i. $w(s_i, s_{i+1}) = \sigma(s_i, s_{i+1}, (-1)^{\mu_i}\mathfrak{b})$.
ii. *If* $i \equiv 1 \pmod 4$, $s_{i+1} \pmod{2^{b+1}} > 2^b$.
iii. *If* $i \equiv 3 \pmod 4$, $s_{i+1} \pmod{2^{b+1}} \leq 2^b$.

Armed with this handle on the ancestor-descendant relationships, we can prove the progress and disjointedness lemmas alluded to in §3.

LEMMA 14. *For odd i, s_i is not* **M***-unmatched.*

LEMMA 15. *For odd i, $s_i \notin X$.*

8. ACKNOWLEDGEMENTS

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. CS is grateful for the support received from the Early Career LDRD program at Sandia National Laboratories.

9. REFERENCES

[1] *Open problems in data streams, property testing, and related topics*, 2011, http://sublinear.info/files/bertinoro2011_kanpur2009.pdf.

[2] 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.

[3] 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.

[4] P. Awasthi, M. Jha, M. Molinaro, and S. Raskhodnikova, *Testing Lipschitz functions on hypergrid domains*, Proceedings of 16th. International Workshop on Randomization and Computation (RANDOM), 2012.

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

[6] A. Bhattacharyya, E. Grigorescu, M. Jha, K. Jung, S. Raskhodnikova, and D. Woodruff, *Lower bounds for local monotonicity reconstruction from transitive-closure spanners*, SIAM Journal of Discrete Math **26** (2012), no. 2, 618–646, Conference version in RANDOM 2010.

[7] 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.

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

[9] E. Blais, S. Raskhodnikova, and G. Yaroslavtsev, *Lower bounds for testing properties of functions on hypergrid domains*, Tech. Report TR13-036, ECCC, March 2013.

[10] 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.

[11] D. Chakrabarty and C. Seshadhri, *Optimal bounds for monotonicity and Lipschitz testing over the hypercube and hypergrids*, Tech. Report Revision #1, TR12-030, ECCC, November 2012.

[12] S. Chaudhuri, S. Gulwani, R. Lublinerman, and S. Navidpour, *Proving programs robust.*, Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE), 2011.

[13] 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.

[14] C. Dwork, F. McSherry, K. Nissim, and A. Smith, *Calibrating noise to sensitivity in private data analysis*, Proceedings of the Theory of Cryptography Conference (TCC), 2006.

[15] 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.

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

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

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

[19] O. Goldreich, S. Goldwasser, and D. Ron, *Property testing and its connection to learning and approximation*, Journal of the ACM **45** (1998), no. 4, 653–750.

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

[21] M. Jha and S. Raskhodnikova, *Testing and reconstruction of Lipschitz functions with applications to data privacy*, Proceedings of the 52nd Annual Symposium on Foundations of Computer Science (FOCS), 2011, pp. 433–442.

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

[23] 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.

[24] R. Rubinfeld and M. Sudan, *Robust characterization of polynomials with applications to program testing*, SIAM Journal of Computing **25** (1996), 647–668.