Isoperimetric Inequalities for Real-Valued Functions with Applications to Monotonicity Testing

Hadley Black
Department of Computer Science, University of California at Los Angeles, CA, USA

Iden Kalemaj
Department of Computer Science, Boston University, MA, USA

Sofya Raskhodnikova
Department of Computer Science, Boston University, MA, USA

Abstract

We generalize the celebrated isoperimetric inequality of Khot, Minzer, and Safra (SICOMP 2018) for Boolean functions to the case of real-valued functions $f : \{0,1\}^d \to \mathbb{R}$. Our main tool in the proof of the generalized inequality is a new Boolean decomposition that represents every real-valued function f over an arbitrary partially ordered domain as a collection of Boolean functions over the same domain, roughly capturing the distance of f to monotonicity and the structure of violations of f to monotonicity.

We apply our generalized isoperimetric inequality to improve algorithms for testing monotonicity and approximating the distance to monotonicity for real-valued functions. Our tester for monotonicity has query complexity $\widetilde{O}(\min(r\sqrt{d}, d))$, where r is the size of the image of the input function. (The best previously known tester makes $O(d)$ queries, as shown by Chakrabarty and Seshadhri (STOC 2013).) Our tester is nonadaptive and has 1-sided error. We prove a matching lower bound for nonadaptive, 1-sided error testers for monotonicity. We also show that the distance to monotonicity of real-valued functions that are α-far from monotone can be approximated nonadaptively within a factor of $O(\sqrt{d \log d})$ with query complexity polynomial in $1/\alpha$ and the dimension d. This query complexity is known to be nearly optimal for nonadaptive algorithms even for the special case of Boolean functions. (The best previously known distance approximation algorithm for real-valued functions, by Fattal and Ron (TALG 2010) achieves $O(d \log r)$-approximation.)

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms; Mathematics of computing → Probabilistic algorithms

Keywords and phrases Isoperimetric inequalities, property testing, monotonicity testing

Digital Object Identifier 10.4230/LIPIcs.ICALP.2023.25

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/abs/2011.09441`

Funding *Hadley Black*: This work was supported by NSF Grant CCF-1553605 and Boston University's Data Science Initiative.
Iden Kalemaj: This work was supported by NSF award CCF-1909612 and Boston University's Dean's Fellowship.
Sofya Raskhodnikova: This work was supported by NSF award CCF-1909612.

Acknowledgements We thank Ramesh Krishnan Pallavoor Suresh for useful discussions.

1 Introduction

We investigate the structure of real-valued functions over the domain $\{0,1\}^d$, the d-dimensional hypercube. Our main contribution is a generalization of a powerful tool from the analysis of Boolean functions, specifically, isoperimetric inequalities[1], to the case of real-valued functions. Isoperimetric inequalities for the undirected hypercube were studied by Margulis [34] and Talagrand [40]. Chakrabarty and Seshadhri [19] had a remarkable insight to develop a directed analogue of the Margulis inequality. This beautiful line of work culminated in the directed analogue of the Talagrand inequality proved by Khot, Minzer, and Safra [32]. We refer to this as the KMS inequality. As Khot, Minzer, and Safra explain in their celebrated work, the Margulis inequality follows from the Talagrand inequality and, more generally, the directed analogue of the Talagrand inequality implies all the other inequalities we mentioned. We generalize all these inequalities to the case of real-valued functions[2].

For the directed case, we prove a generalization of the KMS inequality for functions $f\colon \{0,1\}^d \to \mathbb{R}$. To generalize the *undirected* isoperimetric inequalities, we give a property testing interpretation of the Talagrand inequality. With this interpretation, it is easy to show a generalization of the undirected Talagrand inequality to the case of real-valued functions.

Our proofs of the new isoperimetric inequalities reduce the general case to the Boolean case. Our main tool for generalizing the KMS inequality is a new Boolean decomposition theorem that represents every real-valued function f over an arbitrary partially ordered domain as a collection of Boolean functions over the same domain, roughly capturing the distance of f to monotonicity and the structure of violations of f to monotonicity.

We apply our generalized isoperimetric inequality to improve algorithms for testing monotonicity and approximating the distance to monotonicity for real-valued functions. Our algorithm for testing monotonicity is nonadaptive and has 1-sided error. An algorithm is *nonadaptive* if its input queries do not depend on answers to previous queries. A property testing algorithm has *1-sided error* if it always accepts all inputs with the property it is testing. We show that our algorithm for testing monotonicity is optimal among nonadaptive, 1-sided error testers. Our distance approximation algorithm is nonadaptive. Its query complexity is nearly optimal for nonadaptive algorithms, even for the special case of Boolean functions.

1.1 Isoperimetric Inequalities for Real-Valued Functions

We view the domain of functions $f\colon \{0,1\}^d \to \mathbb{R}$ as the vertices of a d-dimensional hypercube. For the directed isoperimetric inequalities, the edges of the hypercube are ordered pairs (x,y), where $x, y \in \{0,1\}^d$ and there is a unique[3] $i \in [d]$ such that $x_i = 0, y_i = 1$, and $x_j = y_j$ for all coordinates $j \in [d] \setminus \{i\}$. This defines a natural partial order on the domain: $x \preceq y$ if $x_i \le y_i$ for all coordinates $i \in [d]$ or, equivalently, if there is a directed path from x to y in the hypercube. A function $f\colon \{0,1\}^d \to \mathbb{R}$ is *monotone* if $f(x) \le f(y)$ whenever $x \preceq y$. The distance to monotonicity of a function $f\colon \{0,1\}^d \to \mathbb{R}$, denoted $\varepsilon(f)$, is the minimum of

[1] We discuss isoperimetric inequalities that study the size of the "boundary" between the points on which the function takes value 0 and the points on which it takes value 1. The boundary size is defined in terms of the edges of the d-dimensional hypercube with vertices labeled by the values of the function. The edges of the hypercube might be directed or undirected, depending on the type of the inequality.

[2] Following our initial manuscript, [10] and [14] proved generalizations of the KMS inequality to Boolean functions over hypergrids. We remark that our techniques also extend these inequalities to real-valued functions, $f\colon [n]^d \to \mathbb{R}$. See Section 1.3 for more discussion.

[3] Given a positive integer $\ell \in \mathbb{Z}^+$, we let $[\ell]$ denote the set $\{1, 2, \ldots, \ell\}$.

$|\{x \in \{0,1\}^d : f(x) \neq g(x)\}|/2^d$ over all monotone functions $g\colon \{0,1\}^d \to \mathbb{R}$. An edge (x, y) is *violated* by f if $f(x) > f(y)$. Let $\mathcal{S}_f^-$ be the set of violated edges. For $x \in \{0,1\}^d$, let $I_f^-(x)$ be the number of *outgoing* violated edges incident on x, specifically,

$$I_f^-(x) = \left|\left\{y\colon (x,y) \in \mathcal{S}_f^-\right\}\right|.$$

Our main result is the following isoperimetric inequality.

▶ **Theorem 1.1** (Isoperimetric Inequality). *There exists a constant $C > 0$, such that for all functions $f\colon \{0,1\}^d \to \mathbb{R}$,*

$$\mathop{\mathbb{E}}_{\boldsymbol{x} \sim \{0,1\}^d}\left[\sqrt{I_f^-(\boldsymbol{x})}\right] \geq C \cdot \varepsilon(f). \tag{1}$$

Theorem 1.1 is a generalization of the celebrated inequality of Khot, Minzer, and Safra [32] that was strengthened by Pallavoor et al. [36], who proved (1) for the special case of Boolean functions $f\colon \{0,1\}^d \to \{0,1\}$. We show that the same inequality holds for real-valued functions without any dependence on the size of the image of the function. In addition, the constant C is only a factor of 2 smaller than the constant in the inequality of Pallavoor et al.

Applications to monotonicity testing and distance approximation rely on a stronger, "robust" version of Theorem 1.1. The robust version considers an arbitrary 2-coloring $\texttt{col}\colon \mathcal{S}_f^- \to \{\text{red}, \text{blue}\}$ of the violated edges. The color of an edge is used to specify whether the edge is counted towards the lower or the upper endpoint. Let $I_{f,\text{red}}^-(x)$ be the number of *outgoing* red violated edges incident on x, and $I_{f,\text{blue}}^-(x)$ be the number of *incoming* blue violated edges incident on x, specifically,

$$I_{f,\text{red}}^-(x) = \left|\left\{y\colon (x,y) \in \mathcal{S}_f^-,\ \texttt{col}(x,y) = \text{red}\right\}\right|;$$
$$I_{f,\text{blue}}^-(y) = \left|\left\{x\colon (x,y) \in \mathcal{S}_f^-,\ \texttt{col}(x,y) = \text{blue}\right\}\right|.$$

Our next theorem is a generalization of the robust isoperimetric inequality for Boolean functions established by Khot, Minzer, and Safra and strengthened by Pallavoor et al. As before, the constant C is only a factor of 2 smaller for the real-valued case than for the Boolean case.

▶ **Theorem 1.2** (Robust Isoperimetric Inequality). *There exists a constant $C > 0$, such that for all functions $f\colon \{0,1\}^d \to \mathbb{R}$ and colorings $\texttt{col}\colon \mathcal{S}_f^- \to \{\text{red}, \text{blue}\}$,*

$$\mathop{\mathbb{E}}_{\boldsymbol{x} \sim \{0,1\}^d}\left[\sqrt{I_{f,\text{red}}^-(\boldsymbol{x})}\right] + \mathop{\mathbb{E}}_{\boldsymbol{y} \sim \{0,1\}^d}\left[\sqrt{I_{f,\text{blue}}^-(\boldsymbol{y})}\right] \geq C \cdot \varepsilon(f).$$

Note that Theorem 1.2 implies Theorem 1.1 by considering the coloring where all violated edges are red. Therefore, we only present a proof of Theorem 1.2.

1.1.1 Boolean Decomposition

Our main technical contribution is the Boolean decomposition (Theorem 1.3). It allows us to prove Theorem 1.2 by reducing the general case of real-valued functions to the special case of Boolean functions. Theorem 1.3 states that every non-monotone function f can be decomposed into Boolean functions $f_1, f_2, \ldots, f_k$ that collectively preserve the distance to monotonicity of f and violate a subset of the edges violated by f. Crucially, they violate edges in vertex-disjoint subgraphs of the hypercube.

Our Boolean decomposition works for functions over any partially ordered domain. We represent such a domain by a directed acyclic graph (DAG). For a DAG $\mathcal{G}$, we denote its vertex set by $V(\mathcal{G})$ and its edge set by $E(\mathcal{G})$. A DAG $\mathcal{G}$ determines a natural partial order on its vertex set: for all $x, y \in V(\mathcal{G})$, we have $x \preceq y$ if and only if $\mathcal{G}$ contains a path from x to y. A function $f\colon V(\mathcal{G}) \to \mathbb{R}$ is *monotone* if $f(x) \leq f(y)$ whenever $x \preceq y$. An edge (x, y) of $\mathcal{G}$ is *violated* by f if $f(x) > f(y)$. The definitions of $\varepsilon(f)$, the distance of f to monotone, and $\mathcal{S}_f^-$, the set of violated edges, are the same as for the special case of the hypercube.

▶ **Theorem 1.3** (Boolean Decomposition). *Suppose $\mathcal{G}$ is a DAG and $f\colon V(\mathcal{G}) \to \mathbb{R}$ is a function over the vertices of $\mathcal{G}$ that is not monotone. Then, for some $k \geq 1$, there exist Boolean functions $f_1, \ldots, f_k\colon V(\mathcal{G}) \to \{0, 1\}$ and vertex-disjoint (induced) subgraphs $\mathcal{H}_1, \ldots, \mathcal{H}_k$ of $\mathcal{G}$ for which the following hold:*
1. $2\sum_{i=1}^{k} \varepsilon(f_i) \geq \varepsilon(f)$.
2. $\mathcal{S}_{f_i}^- \subseteq \mathcal{S}_f^- \cap E(\mathcal{H}_i)$ *for all* $i \in [k]$.

We derive Theorem 1.2 from Theorem 1.3 in Section 2 and prove Theorem 1.3 in Section 3.

A natural first attempt to proving Theorem 1.1 is to try reducing to the special case of Boolean functions (the KMS inequality) via a thresholding argument. Given $f\colon \{0,1\}^d \to \mathbb{R}$ and $t \in \mathbb{R}$, define $h_t\colon \{0,1\}^d \to \{0,1\}$ to be $h_t(x) = 1$ iff $f(x) > t$. Clearly, this can only reduce the left-hand side of (1) since the influential edges of h_t are a subset of the influential edges of f. Thus, if there exists some $t \in \mathbb{R}$ such that $\varepsilon(h_t) = \Omega(\varepsilon(f))$, then applying the KMS inequality to h_t would show that the inequality also holds for f. In fact, this technique easily allows us to reduce the *undirected* inequality for the real-valued case to the Boolean case, without any significant additional ideas (see Section 7 of the full version [11] for details). However, in the directed setting, a simple argument shows that there exists f for which $\varepsilon(h_t) \leq \varepsilon(f)/r$ for all $t \in \mathbb{R}$, where r is the size of the image of f. Thus, we use additional ideas to prove Theorem 1.1 by a reduction to the KMS inequality. The highly structured decomposition of Theorem 1.3 gives a collection of vertex-disjoint subgraphs $\mathcal{H}_1, \ldots, \mathcal{H}_k$ of the directed hypercube where, in each $\mathcal{H}_i$, an independent "variable thresholding rule" can be applied, yielding the Boolean function f_i. The "threshold" for each vertex x in $\mathcal{H}_i$ depends on the values of the function at a particular set of vertices reachable from x.

The Boolean decomposition is quite powerful: in addition to enabling us to prove the new isoperimetric inequality, it can be used to easily derive a lower bound on the number of edges violated by a real-valued function directly from the bound for the Boolean case, without relying on Theorem 1.2. This bound is used to analyze the edge tester for monotonicity whose significance is described in Section 1.2. The early works on monotonicity testing [30, 25, 39] have shown that $|\mathcal{S}_f^-| \geq \varepsilon(f) \cdot 2^d$ for every Boolean function f on the domain $\{0,1\}^d$. In other words, the number of edges violated by f is at least the number of points on which the value of the function has to change to make it monotone. This bound was generalized to the case of real-valued functions by [25, 39] who showed that $|\mathcal{S}_f^-| \geq (\varepsilon(f)/\lceil \log r \rceil) \cdot 2^d$ for every real-valued function f on the domain $\{0,1\}^d$ and with image size r. (The size of the image of f is the number of distinct values it takes.) Chakrabarty and Seshadhri [17] improved this bound by a factor of $\Theta(\log r)$, thus removing the dependence on the size of the image of the function. Our Boolean decomposition of a real-valued function f in terms of Boolean functions $f_1, \ldots, f_k$, given by Theorem 1.3, yields this result of [17] as an immediate corollary of the special case for Boolean functions:

$$|\mathcal{S}_f^-| \geq \sum_{i=1}^{k} |\mathcal{S}_{f_i}^-| \geq \sum_{i=1}^{k} \varepsilon(f_i) \cdot 2^d \geq \varepsilon(f) \cdot 2^{d-1},$$

where the inequalities follow by first applying Item 2 of Theorem 1.3, then applying the bound for the Boolean case, and, finally, applying Item 1 of Theorem 1.3.

1.1.2 Undirected Isoperimetric Inequality for Real-Valued Functions

The original isoperimetric inequality of Talagrand [40] treats the domain $\{0,1\}^d$ as an undirected hypercube. An undirected edge $\{x,y\}$ is *influential* if $f(x) \neq f(y)$. Let $I_f(x)$ be the number of influential edges $\{x,y\}$ incident on $x \in \{0,1\}^d$ for which $f(x) > f(y)$. This definition ensures that each influential edge is counted towards $I_f(x)$ for exactly one vertex x. The variance $\text{var}(f)$ of a Boolean function is defined as $p_0(1-p_0)$, where p_0 is the probability that $f(x) = 0$ for a uniformly random point x in the domain. Talagrand [40] proved the following.

▶ **Theorem 1.4** (Talagrand Inequality [40]). *For all functions $f\colon \{0,1\}^d \to \{0,1\}$,*

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\{0,1\}^d}\left[\sqrt{I_f(\boldsymbol{x})}\right] \geq \sqrt{2}\ \text{var}(f). \tag{2}$$

Before generalizing Theorem 1.4 to real-valued functions, we reinterpret it using a property testing notion. Observe that the natural definition of the variance of a real-valued function results in a quantity that depends on specific values of the function, whereas whether an edge is influential depends only on whether the values on its endpoints are different and not on the specific values themselves. So, variance is not a suitable notion for generalizing this inequality. We replace the variance of f with the distance of f to constant, denoted $\texttt{dist}(f,\textbf{const})$, i.e., the minimum of $\Pr_{\boldsymbol{x}\sim\{0,1\}^d}[f(\boldsymbol{x}) \neq g(\boldsymbol{x})]$ over all constant functions $g\colon \{0,1\}^d \to \mathbb{R}$. For a Boolean function f, the distance to constant is $\min\{p_0,(1-p_0)\}$ and, therefore, the left-hand side of (2) is at least $\texttt{dist}(f,\textbf{const})/\sqrt{2}$. Next, we state our generalization of Talagrand's inequality. See Section 7 of the full version [11] for the proof.

▶ **Theorem 1.5** (Undirected Isoperimetric Inequality). *For all functions $f\colon \{0,1\}^d \to \mathbb{R}$,*

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\{0,1\}^d}\left[\sqrt{I_f(\boldsymbol{x})}\right] \geq \frac{\texttt{dist}(f,\textbf{const})}{2\sqrt{2}}.$$

Note that natural generalizations of the Margulis inequality and the inequality of Chakrabarty and Seshadhri to the real range follow from Theorem 1.2 (for the the special case of Boolean functions, the implication is discussed in [32], and it holds for the real range for the same reasons).

1.2 Applications of Our Isoperimetric Inequality for Real-Valued Functions

We apply our generalized isoperimetric inequality (Theorem 1.2) to improve algorithms for testing monotonicity and approximating the distance to monotonicity for real-valued functions.

1.2.1 Monotonicity Testing

Monotonicity of functions, first studied in the context of property testing by Goldreich et al. [30], is one of the most widely investigated properties in this model [26, 25, 39, 33, 29, 1, 28, 31, 3, 38, 2, 7, 15, 12, 17, 18, 19, 13, 16, 21, 5, 23, 35, 8, 32, 20, 9]. A function is ε-far from monotone if its distance to monotonicity is at least ε; otherwise, it is ε-close to monotone. An ε-tester for monotonicity is a randomized algorithm that, given a parameter $\varepsilon \in (0,1)$ and oracle access to a function f, accepts with probability at least 2/3 if f is monotone and rejects with probability at least 2/3 if f is ε-far from monotone. Prior to our work, the best monotonicity tester for real-valued functions was the *edge tester*. The edge tester, introduced by [30], queries the values of f on the endpoints of uniformly random edges of the hypercube and rejects if it finds a violated edge. As we discussed in Section 1.1, a series of works [30, 25, 39, 17] proved lower bounds on $|\mathcal{S}_f^-|$, the number of violated edges, resulting in the tight analysis of the edge tester for both Boolean and real-valued functions: $O(d/\varepsilon)$ queries are sufficient (and also necessary, e.g., for $f(x) = 1 - x_1$, the anti-dictator function). For many years, it remained open whether an $o(d)$-query tester for monotonicity existed, until a sequence of breakthroughs [19, 22, 32] designed testers for Boolean functions with query complexity $\widetilde{O}(d^{7/8}), \widetilde{O}(d^{5/6})$, and finally $\widetilde{O}(\sqrt{d})$. Prior to our work, the same question remained open for functions with image size, r, greater than 2.

We show that when r is small compared to d, monotonicity can be tested with $o(d)$ queries. (Note that $r \leq 2^d$.)

▶ **Theorem 1.6.** *There exists a nonadaptive, 1-sided error ε-tester for monotonicity of functions $f\colon \{0,1\}^d \to \mathbb{R}$ that makes $\widetilde{O}\Big(\min\big(\frac{r\sqrt{d}}{\varepsilon^2}, \frac{d}{\varepsilon}\big)\Big)$ queries and works for all functions f with image size r.*

The proof of Theorem 1.6 (in Section 4) heavily relies on the generalized isoperimetric inequality of Theorem 1.2. We extend several other combinatorial properties of Boolean functions to real-valued functions. In particular, the persistence of a vertex $x \in \{0,1\}^d$ is a key combinatorial concept in the analysis. A vertex $x \in \{0,1\}^d$ is τ-persistent if, with high probability, a random walk that starts at x and takes τ steps in the d-dimensional directed hypercube ends at a vertex y for which $f(y) \leq f(x)$. As we show, the upper bound on the number of vertices which are not τ-persistent grows linearly with the distance τ *and* the image size r. For the tester analysis, one needs to carefully choose the distance parameter τ for which many vertices are τ-persistent. In particular, this value of τ also depends on the image size r, resulting in the linear dependence on r in the query complexity of the tester.

1.2.2 Our Lower Bound for Testing Monotonicity

We show that our monotonicity tester is optimal among nonadaptive, 1-sided error testers.

▶ **Theorem 1.7.** *There exists a constant $\varepsilon > 0$, such that for all $d, r \in \mathbb{N}$, every nonadaptive, 1-sided error ε-tester for monotonicity of functions $f\colon \{0,1\}^d \to [r]$ requires $\Omega(\min(r\sqrt{d}, d))$ queries.*

We prove Theorem 1.7 by generalizing a construction of Fischer et al. [29] that showed that nonadaptive, 1-sided error monotonicity testers of Boolean functions must make $\Omega(\sqrt{d})$ queries. We refer the reader to Section 6 of the full version [11] for the proof. Blais et al. [12] demonstrated that every tester for monotonicity over the d-dimensional hypercube domain requires $\Omega(\min(d, r^2))$ queries. Our lower bound is stronger when $r \in [2, \sqrt{d}]$, although it applies only to nonadaptive, 1-sided error algorithms.

1.2.3 Approximating the Distance to Monotonicity

Motivated by the desire to handle noisy inputs, Parnas et al. [38] generalized the property testing model to tolerant testing. There is a direct connection between tolerant testing of a property and approximating the distance to the property with additive and multiplicative error in the sense that these problems can be reduced to each other with the right setting of parameters and have the same query complexity up to logarithmic factors (see, e.g., [38, Claim 2] and [36, Theorem A.1]). One clean way to state distance approximation guarantees is to replace the additive error α with the promise that the input function is α-far from the property, as specified in the following definition. A randomized *c-approximation algorithm for the distance to monotonicity,* where $c > 1$, is given a parameter $\alpha \in (0, 1)$ and oracle access to a function $f\colon \{0,1\}^d \to \mathbb{R}$ that is α-far from monotone. It outputs an estimate $\hat{\varepsilon}$ that, with probability at least $2/3$, satisfies $\varepsilon(f) \le \hat{\varepsilon} \le c \cdot \varepsilon(f)$.

Fattal and Ron [27] studied the problem of approximating the distance to monotonicity for real-valued functions over the hypergrid domain $[n]^d$. For the special case of the hypercube domain, they give an $O(d \log r)$-approximation algorithm for functions with image size r that makes $\mathrm{poly}(d, 1/\alpha)$ queries. Theorem 1.2 allows us to improve on their result, by showing that the algorithm of Pallavoor et al. [36] for approximating the distance to monotonicity of Boolean functions also works for real-valued functions, without any loss in the approximation guarantee.

▶ **Theorem 1.8.** *There exists a nonadaptive $O(\sqrt{d \log d})$-approximation algorithm for the distance to monotonicity that, given a parameter $\alpha \in (0, 1)$ and oracle access to a function $f\colon \{0,1\}^d \to \mathbb{R}$ that is α-far from monotone, makes poly$(d, 1/\alpha)$ queries.*

Pallavoor et al. prove that this approximation ratio is nearly optimal for nonadaptive algorithms, even for the special case of Boolean functions. We also note that, by the connection between tolerant testing and erasure-resilient testing observed by Dixit et al. [24], our Theorem 1.8 implies the existence of an erasure-resilient ε-tester for monotonicity of functions $f : \{0,1\}^d \to \mathbb{R}$ that can handle up to $\Theta(\varepsilon/\sqrt{d \log d})$ erasures with query complexity $\mathrm{poly}(d, 1/\varepsilon)$. The tester of Dixit et al. could handle only $O(\varepsilon/d)$ erasures. For the proof of Theorem 1.8, we refer the reader to Section 5 of the full version [11].

1.3 Other Prior Work on Monotonicity Testing and Open Questions

The query complexity of monotonicity testing of Boolean functions over the hypercube has been resolved for nonadaptive testers by Chen et al. [21, 23] who proved a lower bound of $\widetilde{\Omega}(\sqrt{d})$. For adaptive testers, the best lower bound known to date is $\widetilde{\Omega}(d^{1/3})$, also shown by [23]. It is an open question whether adaptive algorithms can do better than nonadaptive ones for functions over the hypercube domain, both in the case of Boolean functions and, more generally, for functions with small image size. As we mentioned before, there is a lower bound of $\Omega(d)$ for functions with image size $\Omega(\sqrt{d})$ [12].

Monotonicity testing has also been studied for functions on other types of domains, including general partially ordered domains [29], with particular attention to the hypergrid domain $[n]^d$. (It has also been investigated in the context where the distance to monotonicity is the normalized L_p distance instead of the Hamming distance [6], but we focus our attention here on the Hamming distance.) When $d = 1$, monotonicity testing on the hypergrid $[n]$ is equivalent to testing sortedness of n-element arrays. This problem was introduced by Ergun et al. [26]. Its query complexity has been completely pinned down in terms of n and ε by [26, 28, 18, 4]: it is $\Theta(\frac{\log(\varepsilon n)}{\varepsilon})$. Pallavoor et al. [35, 37] considered the setting when the tester is given an additional parameter r, the number of distinct elements in the array, and obtained an $O((\log r)/\varepsilon)$-query algorithm. There are also lower bounds for this setting: $\Omega(\log r)$ for nonadaptive algorithms by [13] and $\Omega(\frac{\log r}{\log \log r})$ for all testers for the case when $r = n^{1/3}$ by [4].

For general d, Black et al. [8, 9] gave an $\widetilde{O}(d^{5/6})$-query tester for Boolean functions $f\colon [n]^d \to \{0,1\}$. For real-valued functions, Chakrabarty and Seshadhri [17, 18] proved basically matching upper and lower bounds of $O((d \log n)/\varepsilon)$ and $\Omega((d \log n - \log \varepsilon^{-1})/\varepsilon)$. However, their lower bound only applies for functions with a large image. Pallavoor et al. [35]

gave an $O(\frac{d}{\varepsilon} \cdot \log \frac{d}{\varepsilon} \cdot \log r)$-query tester, where r, the size of the image, is given to the tester as a parameter. It remains open whether there is an $\widetilde{O}(\sqrt{d})$-query tester for Boolean functions on the hypergrid domain.

1.3.1 Discussion of Results Published After our Initial Manuscript

Recently, in independent works, [10] and [14] showed generalizations of the isoperimetric inequality of [32] to Boolean functions on general hypergrids (see [10, Theorem 1.4] and [14, Theorem 1.3]). These works obtain $\widetilde{O}(n\sqrt{d}/\varepsilon^2)$-query and $\widetilde{O}(n^3\sqrt{d}/\varepsilon^2)$-query nonadaptive, 1-sided error monotonicity testers, respectively, for such functions. Our Boolean decomposition (Theorem 1.3) implies that these isoperimetric inequalities also hold for functions $f\colon [n]^d \to \mathbb{R}$ by the approach described in Section 2. We also believe that this should imply the existence of an $\widetilde{O}(rn\sqrt{d}/\varepsilon^2)$-query tester for functions $f\colon [n]^d \to [r]$. A possible approach to proving this could be to generalize the analysis given in Section 7 of [10] to the case of range $[r]$. Presumably, this would follow the same approach as in our Section 4 in which we prove Theorem 1.6, but adapted to hypergrids. Since [10, 14] were published well after our initial manuscript, we will refrain from going into further details on their relationship with our results.

2 Directed Talagrand Inequality for Real-Valued Functions

In this section, we use our Boolean decomposition (Theorem 1.3) to prove Theorem 1.2, which easily implies the non-robust version (Theorem 1.1) as we point out in the introduction. Let $f\colon \{0,1\}^d \to \mathbb{R}$ be a non-monotone function over the d-dimensional hypercube and let $\texttt{col}\colon \mathcal{S}_f^- \to \{\text{red}, \text{blue}\}$ be an arbitrary 2-coloring of $\mathcal{S}_f^-$. Given $x \in \{0,1\}^d$ and a subgraph $\mathcal{H}$ of the d-dimensional hypercube, we define the quantities

$$I^-_{f,\text{red},\mathcal{H}}(x) = \left|\left\{y\colon (x,y) \in \mathcal{S}_f^- \cap E(\mathcal{H}),\ \texttt{col}(x,y) = \text{red}\right\}\right|;$$
$$I^-_{f,\text{blue},\mathcal{H}}(y) = \left|\left\{x\colon (x,y) \in \mathcal{S}_f^- \cap E(\mathcal{H}),\ \texttt{col}(x,y) = \text{blue}\right\}\right|.$$

Let $f_1, \ldots, f_k\colon \{0,1\}^d \to \{0,1\}$ be the Boolean functions and $\mathcal{H}_1, \ldots, \mathcal{H}_k$ be the vertex-disjoint subgraphs of the d-dimensional hypercube that are guaranteed by Theorem 1.3. Let C' denote the constant from the robust Boolean isoperimetric inequality (Theorem 2.7 of [36]) that is hidden by Ω. We have

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\{0,1\}^d}\left[\sqrt{I^-_{f,\text{red}}(\boldsymbol{x})}\right] + \mathop{\mathbb{E}}_{\boldsymbol{y}\sim\{0,1\}^d}\left[\sqrt{I^-_{f,\text{blue}}(\boldsymbol{y})}\right]$$
$$\geq \mathop{\mathbb{E}}_{\boldsymbol{x}}\left[\sqrt{I^-_{f,\text{red},\bigcup_{i=1}^k \mathcal{H}_i}(\boldsymbol{x})}\right] + \mathop{\mathbb{E}}_{\boldsymbol{y}}\left[\sqrt{I^-_{f,\text{blue},\bigcup_{i=1}^k \mathcal{H}_i}(\boldsymbol{y})}\right] \tag{3}$$
$$= \sum_{i=1}^k \left(\mathop{\mathbb{E}}_{\boldsymbol{x}}\left[\sqrt{I^-_{f,\text{red},\mathcal{H}_i}(\boldsymbol{x})}\right] + \mathop{\mathbb{E}}_{\boldsymbol{y}}\left[\sqrt{I^-_{f,\text{blue},\mathcal{H}_i}(\boldsymbol{y})}\right]\right) \tag{4}$$
$$\geq \sum_{i=1}^k \left(\mathop{\mathbb{E}}_{\boldsymbol{x}}\left[\sqrt{I^-_{f_i,\text{red},\mathcal{H}_i}(\boldsymbol{x})}\right] + \mathop{\mathbb{E}}_{\boldsymbol{y}}\left[\sqrt{I^-_{f_i,\text{blue},\mathcal{H}_i}(\boldsymbol{y})}\right]\right) \tag{5}$$
$$= \sum_{i=1}^k \left(\mathop{\mathbb{E}}_{\boldsymbol{x}}\left[\sqrt{I^-_{f_i,\text{red}}(\boldsymbol{x})}\right] + \mathop{\mathbb{E}}_{\boldsymbol{y}}\left[\sqrt{I^-_{f_i,\text{blue}}(\boldsymbol{y})}\right]\right) \tag{6}$$

$$\geq \sum_{i=1}^{k} C' \cdot \varepsilon(f_i) \tag{7}$$

$$\geq \frac{C' \cdot \varepsilon(f)}{2}. \tag{8}$$

The inequality (3) holds because $\bigcup_{i=1}^{k} \mathcal{H}_i$ is a subgraph of the d-dimensional hypercube. The equality (4) holds because the $\mathcal{H}_i$'s are vertex-disjoint. The inequality (5) holds since $\mathcal{S}_{f_i}^- \subseteq \mathcal{S}_f^-$ and the equality (6) holds since $\mathcal{S}_{f_i}^- \subseteq E(\mathcal{H}_i)$ (these are both by item 2 of Theorem 1.3). Finally, (7) is due to [36, Theorem 2.7] and (8) is due to item 1 of Theorem 1.3.

3 Boolean Decomposition: Proof of Theorem 1.3

In this section, we prove Boolean Decomposition (Theorem 1.3). Our results consider any partially ordered domain, which we represent by a DAG $\mathcal{G}$. The *transitive closure* of $\mathcal{G}$, denoted $\mathrm{TC}(\mathcal{G})$, is the graph with vertex set $V(\mathcal{G})$ and edge set $\{(x,y) \colon x \prec y\}$. The *violation graph* of f is the graph $(V(\mathcal{G}), E')$, where E' is the set of edges of $\mathrm{TC}(\mathcal{G})$ violated by f.

In Section 3.1, we define the key notion of sweeping graphs and identify some of their important properties. In Section 3.2, we prove a general lemma that shows how to use a matching M in $\mathrm{TC}(\mathcal{G})$ to find vertex-disjoint sweeping graphs in $\mathcal{G}$ satisfying a "matching rearrangement" property. The techniques in Section 3.1 and Section 3.2 are inspired by the techniques of [8] used to analyze Boolean functions on the hypergrid domain, $[n]^d$. In Section 3.3, we apply our matching decomposition lemma to a carefully chosen matching to obtain the subgraphs $\mathcal{H}_1, \ldots, \mathcal{H}_k$. Finally, in Section 3.4, we define the Boolean functions $f_1, \ldots, f_k$ and complete the proof of Theorem 1.3.

3.1 Sweeping Graphs and Their Properties

Given a graph $\mathcal{G}$ and two subgraphs $\mathcal{H}_1$ and $\mathcal{H}_2$, we define the union $\mathcal{H}_1 \cup \mathcal{H}_2$ to be the graph with vertex set $V(\mathcal{H}_1) \cup V(\mathcal{H}_2)$ and edge set $E(\mathcal{H}_1) \cup E(\mathcal{H}_2)$.

▶ **Definition 3.1** ((S,T)-Sweeping Graphs). *Given a DAG $\mathcal{G}$ and $s,t \in V(\mathcal{G})$, define $\mathcal{H}(s,t)$ to be the subgraph of $\mathcal{G}$ formed by the union of all directed paths in $\mathcal{G}$ from s to t. Given two disjoint subsets $S,T \subseteq V(\mathcal{G})$, define the* (S,T)-sweeping graph, *denoted $\mathcal{H}(S,T)$, to be the union of directed paths in $\mathcal{G}$ that start from some $s \in S$ and end at some $t \in T$. That is,*

$$\mathcal{H}(S,T) = \bigcup_{(s,t) \in S \times T} \mathcal{H}(s,t).$$

Note that if $s \not\preceq t$ then $\mathcal{H}(s,t) = \emptyset$.

We now prove three properties of sweeping graphs which we use in Section 3.4 to analyze our functions $f_1, \ldots, f_k$. Given disjoint sets $S,T \subseteq V(\mathcal{G})$ and $z \in V(\mathcal{H}(S,T))$, define the sets

$$S(z) = \{s \in S \colon s \preceq z\} \text{ and } T(z) = \{t \in T \colon z \preceq t\}.$$

▷ Claim 3.2 (Properties of Sweeping Graphs). Let $\mathcal{G}$ be a DAG and $S,T \subseteq V(\mathcal{G})$ be disjoint sets.

1. *(Property of Nodes in a Sweeping Graph):* If $z \in V(\mathcal{H}(S,T))$ then $S(z) \neq \emptyset$ and $T(z) \neq \emptyset$.
2. *(Property of Nodes Outside of a Sweeping Graph):* If $z \in V(\mathcal{G}) \setminus V(\mathcal{H}(S,T))$ then at most one of the following is true: (a) $\exists y \in V(\mathcal{H}(S,T))$ such that $z \prec y$, (b) $\exists x \in V(\mathcal{H}(S,T))$ such that $x \prec z$.
3. *(Sweeping Graphs are Induced):* If $x,y \in V(\mathcal{H}(S,T))$ and $(x,y) \in E(\mathcal{G})$ then $(x,y) \in E(\mathcal{H}(S,T))$.

Proof. Property 1 holds by definition of the sweeping graph $\mathcal{H}(S,T)$. If $z \in V(\mathcal{H}(S,T))$, then, by definition of $\mathcal{H}(S,T)$, there exist $s \in S$ and $t \in T$ for which z belongs to some directed path from s to t. That is, $z \in V(\mathcal{H}(s,t))$. Thus, $s \in S(z)$ and $t \in T(z)$, and property 1 holds.

We now prove property 2. Suppose, for the sake of contradiction, that there exist $x, y, z \in V(\mathcal{G})$ for which $x, y \in V(\mathcal{H}(S,T))$, $z \notin V(\mathcal{H}(S,T))$, and $x \prec z \prec y$. By property 1, there exist some $s \in S(x)$ and some $t \in T(y)$. Then $s \preceq x \prec z \prec y \preceq t$ and, consequently, z belongs to some directed path from s to t. Thus, $z \in V(\mathcal{H}(s,t))$, and so $z \in V(\mathcal{H}(S,T))$. This is a contradiction.

We now prove property 3. Suppose $x, y \in V(\mathcal{H}(S,T))$ and $(x,y) \in E(\mathcal{G})$. By property 1, there exist $s \in S$ and $t \in T$ for which $s \preceq x$ and $y \preceq t$. Since $(x,y) \in E(\mathcal{G})$, we have $x \prec y$ and so $s \preceq x \prec y \preceq t$. Thus, the edge (x,y) belongs to a directed path from s to t. That is, $(x,y) \in E(\mathcal{H}(s,t))$ and so $(x,y) \in E(\mathcal{H}(S,T))$. ◁

3.2 Matching Decomposition Lemma for DAGs

In this section, we prove the following matching decomposition lemma. Recall that $\mathrm{TC}(\mathcal{G})$ denotes the transitive closure of $\mathcal{G}$, which is the graph with vertex set $V(\mathcal{G})$ and edge set $\{(x,y) \colon x \prec y\}$. Consider a matching M in $\mathrm{TC}(\mathcal{G})$. We represent $M \colon S \to T$ as a bijection between two disjoint sets $S, T \subseteq V(\mathcal{G})$ of the same size for which $s \prec M(s)$ for all $s \in S$. For a set $S' \subseteq S$, define $M(S') = \{M(s) \colon s \in S'\}$. Note that for convenience we will sometimes abuse notation and represent M as the set of pairs, $\{(s, M(s)) \colon s \in S\}$, instead of as a bijection.

▶ **Lemma 3.3** (Matching Decomposition Lemma for DAGs). *For every DAG $\mathcal{G}$ and every matching $M \colon S \to T$ in $\mathrm{TC}(\mathcal{G})$, there exist partitions $(S_i \colon i \in [k])$ of S and $(T_i \colon i \in [k])$ of T, where $M(S_i) = T_i$ for all $i \in [k]$, and the following hold.*

1. (Sweeping Graph Disjointness): *$V(\mathcal{H}(S_i, T_i)) \cap V(\mathcal{H}(S_j, T_j)) = \emptyset$ for all $i \neq j \in [k]$.*
2. (Matching Rearrangement Property): *For all $i \in [k]$ and $(x,y) \in S_i \times T_i$, if $x \prec y$ then there exists a matching $\widehat{M} \colon S_i \to T_i$ in $\mathrm{TC}(\mathcal{G})$ for which $(x,y) \in \widehat{M}$.*

Proof. In Algorithm 1, we show how to construct partitions $(S_i \colon i \in [k])$ for S and $(T_i \colon i \in [k])$ for T from a matching M in $\mathrm{TC}(\mathcal{G})$. We use the following notion of conflicting pairs.

▶ **Definition 3.4** (Conflicting Pairs). *Given a DAG $\mathcal{G}$ and four disjoint sets $X, Y, X', Y' \subset V(\mathcal{G})$, we say the two pairs (X,Y) and (X',Y') conflict if $V(\mathcal{H}(X,Y)) \cap V(\mathcal{H}(X',Y')) \neq \emptyset$.*

Algorithm 1 Algorithm for constructing conflict-free pairs from a matching M.

Require: A DAG $\mathcal{G}$ and a matching $M \colon S \to T$ in $\mathrm{TC}(\mathcal{G})$.

```
1: $\mathcal{Q}_0 \leftarrow \{(\{x\},\{y\}) \colon (x,y) \in M\}$                          ▷ Initialize pairs using $M$
2: for $s \geq 0$ do
3:     if two pairs $(X,Y) \neq (X',Y') \in \mathcal{Q}_s$ conflict then
4:         $\mathcal{Q}_{s+1} \leftarrow (\mathcal{Q}_s \setminus \{(X,Y),(X',Y')\}) \cup \{(X \cup X', Y \cup Y')\}$ ▷ Merge conflicting pairs
5:     else
6:         $s^* \leftarrow s$ and return $\mathcal{Q}_{s^*}$                       ▷ Terminate when there are no conflicts
```

The following observation is apparent and by design of Algorithm 1.

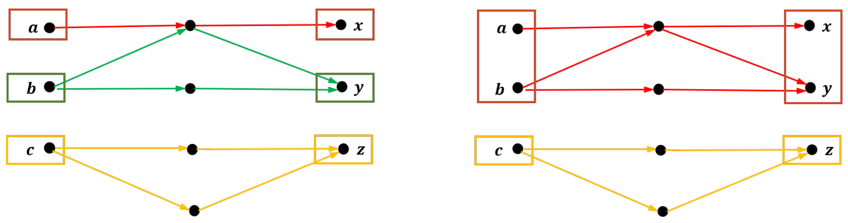

Figure 1 An illustration for Algorithm 1 with input matching $M = \{(a,x),(b,y),(c,z)\}$. We initialize $\mathcal{Q}_0 = \{(\{a\},\{x\}),(\{b\},\{y\}),(\{c\},\{z\})\}$. The pairs $(\{a\},\{x\})$ and $(\{b\},\{y\})$ conflict, so we merge them to obtain a new and final collection $Q_1 = \{(\{a,b\},\{x,y\}),(\{c\},\{z\})\}$.

▶ **Observation 3.5** (Loop Invariants of Algorithm 1). *For all $s \in \{0,1,\dots,s^*\}$, (a) $M(X) = Y$ for all $(X,Y) \in \mathcal{Q}_s$, (b) $(X\colon (X,\cdot) \in \mathcal{Q}_s)$ is a partition of S, and (c) $(Y\colon (\cdot,Y) \in \mathcal{Q}_s)$ is a partition of T.*

Given a matching $M\colon S \to T$ in $\mathrm{TC}(\mathcal{G})$, we run Algorithm 1 to obtain the set $\mathcal{Q}_{s^*}$. See Fig. 1 for an illustration. Define $k = |\mathcal{Q}_{s^*}|$ and let $\{(S_i,T_i)\colon i \in [k]\}$ be the set of pairs in $\mathcal{Q}_{s^*}$. By Observation 3.5, $(S_i\colon i \in [k])$ is a partition of S, $(T_i\colon i \in [k])$ is a partition of T, and $M(S_i) = T_i$ for all $i \in [k]$. Item 1 of Lemma 3.3 holds since Algorithm 1 terminates at step s only when all pairs in $\mathcal{Q}_s$ are non-conflicting (recall Definition 3.4). Thus, to prove Lemma 3.3 it only remains to prove item 2. To do so, we prove the following Claim 3.6, that easily implies item 2. Note that while we only require Claim 3.6 to hold for the special case of $s = s^*$, using an inductive argument on s allows us to give a proof for all $s \in \{0,1,\dots,s^*\}$.

▷ Claim 3.6 (Rematching Claim). For all $s \in \{0,1,\dots,s^*\}$, pairs $(X,Y) \in \mathcal{Q}_s$, and $(x,y) \in X \times Y$, there exists a matching $\widehat{M}\colon X \setminus \{x\} \to Y \setminus \{y\}$ in $TC(\mathcal{G})$.

Proof. The proof is by induction on s. For the base case, if $s = 0$, then, by inspection of Algorithm 1, for $(X,Y) \in \mathcal{Q}_0$, we must have $X = \{x\}$ and $Y = \{y\}$. Thus, setting $\widehat{M} = \emptyset$ trivially proves the claim.

Now let $s > 0$. Fix some $(X,Y) \in \mathcal{Q}_s$ and $(x,y) \in X \times Y$. Let $(X_1,Y_1),(X_2,Y_2) \in \mathcal{Q}_{s-1}$ be the pairs of sets in $\mathcal{Q}_{s-1}$ for which $x \in X_1$ and $y \in Y_2$. First, if $(X_1,Y_1) = (X_2,Y_2)$, then by induction there exists a matching $\widehat{M'}\colon X_1 \setminus \{x\} \to Y_1 \setminus \{y\}$ in $\mathrm{TC}(\mathcal{G})$. Note that by definition of Algorithm 1, we must have $X_1 \subseteq X$ and $Y_1 \subseteq Y$. Then the required matching is $\widehat{M} = \widehat{M'} \cup M|_{X \setminus X_1}$ where $M|_{(\cdot)}$ denotes the restriction of the original matching M to the set $(\cdot)$. Suppose $(X_1,Y_1) \neq (X_2,Y_2)$. This is the interesting case, and we give an accompanying illustration in Fig. 2. By definition of Algorithm 1, it must be that (X_1,Y_1) and (X_2,Y_2) *conflict* (recall Definition 3.4) and were merged to form $X = X_1 \cup X_2$ and $Y = Y_1 \cup Y_2$. Thus, there exists some vertex $z \in V(\mathcal{H}(X_1,Y_1)) \cap V(\mathcal{H}(X_2,Y_2))$ and $x_1 \in X_1, y_1 \in Y_1, x_2 \in X_2, y_2 \in Y_2$ for which $x_1 \preceq z \preceq y_1$ and $x_2 \preceq z \preceq y_2$.

We now invoke the inductive hypothesis to get matchings $\widehat{M}_1\colon X_1 \setminus \{x\} \to Y_1 \setminus \{y_1\}$ and $\widehat{M}_2\colon X_2 \setminus \{x_2\} \to Y_2 \setminus \{y\}$ in $\mathrm{TC}(\mathcal{G})$. Observe that $x_2 \preceq z \preceq y_1$ and thus we can match x_2 and y_1. The required matching in $\mathrm{TC}(\mathcal{G})$ is $\widehat{M} = \widehat{M}_1 \cup \widehat{M}_2 \cup \{(x_2,y_1)\}$. ◁

We conclude the proof of Lemma 3.3 by showing that Claim 3.6 implies item 2. We are given $(S_i,T_i) \in \mathcal{Q}_{s^*}$ for some $i \in [k]$ and $(x,y) \in S_i \times T_i$ where $x \prec y$. By Claim 3.6, there exists a matching $\widehat{M'}\colon S_i \setminus \{x\} \to T_i \setminus \{y\}$ in $\mathrm{TC}(\mathcal{G})$. We set $\widehat{M} = \widehat{M'} \cup \{(x,y)\}$. Since $x \prec y$, the final matching $\widehat{M}\colon S_i \to T_i$ is a matching in $\mathrm{TC}(\mathcal{G})$ which contains the pair (x,y). ◀

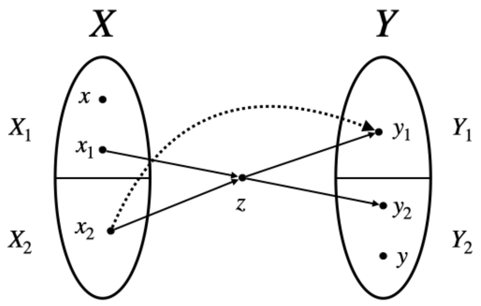

Figure 2 An illustration for the case of $(X_1, Y_1) \neq (X_2, Y_2)$ in the proof of Claim 3.6. The solid lines represent directed paths. The dotted line represents the pair (x_2, y_1) added to obtain the final matching $\widehat{M}$. The only vertices of $X \cup Y$ not participating in $\widehat{M}$ are x and y.

3.3 Specifying a Matching to Construct the Subgraphs $\mathcal{H}_1, \ldots, \mathcal{H}_k$

In this section, we apply Lemma 3.3 to a carefully chosen matching M in order to construct our vertex-disjoint subgraphs $\mathcal{H}_1, \ldots, \mathcal{H}_k$.

▶ **Definition 3.7** (Max-weight, Min-cardinality Matching). *A matching M in* $\mathrm{TC}(\mathcal{G})$ *is a max-weight, min-cardinality matching for f if M maximizes $\sum_{(x,y)\in M}(f(x) - f(y))$ and among such matchings minimizes $|M|$.*

Henceforth, let M denote a max-weight, min-cardinality matching. Let S and T denote the set of lower and upper endpoints, respectively, of M. We use the following well-known fact on matchings in the violation graph.

▶ **Fact 3.8** (Corollary 2 [29]). *For a* DAG *$\mathcal{G}$ and function $f \colon V(\mathcal{G}) \to \mathbb{R}$, the distance to monotonicity $\varepsilon(f)$ is equal to the size of the minimum vertex cover of the violation graph of f divided by $|V(\mathcal{G})|$.*

▶ **Fact 3.9.** *M is a matching in the violation graph of f that is also maximal. That is, (a) $f(x) > f(y)$ for all $(x, y) \in M$ and (b) $|M| \geq (\varepsilon(f) \cdot |V(\mathcal{G})|)/2$.*

Proof. First, for the sake of contradiction, suppose $f(x) \leq f(y)$ for some pair $(x, y) \in M$. Then we can set $M = M \setminus \{(x, y)\}$, which can only increase $\sum_{(x,y)\in M}(f(x) - f(y))$ and will decrease $|M|$ by 1. This contradicts the definition of M. Thus, $f(x) > f(y)$ for all $(x, y) \in M$ and so M is a matching in the *violation graph of f*. Second, since M maximizes $\sum_{(x,y)\in M}(f(x) - f(y))$, it must also be a *maximal* matching in the violation graph of f. Thus, (b) follows from Fact 3.8 and the fact that the size of any maximal matching is at least half the size of the minimum vertex cover. ◀

We now apply Lemma 3.3 to M, obtaining the partitions $(S_i \colon i \in [k])$ and $(T_i \colon i \in [k])$ for S and T, respectively, for which $M(S_i) = T_i$ for all $i \in [k]$. For each $i \in [k]$, let $\mathcal{H}_i = \mathcal{H}(S_i, T_i)$. We use the collection of sweeping graphs $\mathcal{H}_1, \ldots, \mathcal{H}_k$ to prove Theorem 1.3. Note that these subgraphs are all vertex-disjoint by item 1 of Lemma 3.3. We use item 2 of Lemma 3.3 to prove the following lemma regarding the (S_i, T_i) pairs. The proof crucially relies on the fact that M is a max-weight, min-cardinality matching.

▶ **Lemma 3.10** (Property of the Pairs (S_i, T_i)). *For all $i \in [k]$ and $(x, y) \in S_i \times T_i$, if $x \prec y$ then $f(x) > f(y)$.*

Proof. Suppose there exists $i \in [k]$, $x \in S_i$, and $y \in T_i$ for which $x \prec y$ and $f(x) \leq f(y)$. By item 2 of Lemma 3.3 there exists a matching $\widehat{M} \colon S \to T$ in $\mathrm{TC}(\mathcal{G})$ for which $(x,y) \in \widehat{M}$. In particular, since M and $\widehat{M}$ have identical sets of lower and upper endpoints,

$$\sum_{(s,t)\in\widehat{M}} (f(s) - f(t)) = \sum_{(s,t)\in M} (f(s) - f(t)) \text{ and } |\widehat{M}| = |M|.$$

Now set $\widehat{M}' = \widehat{M} \setminus \{(x,y)\}$ and observe that since $f(x) \leq f(y)$,

$$\sum_{(s,t)\in\widehat{M}'} (f(s) - f(t)) \geq \sum_{(s,t)\in M} (f(s) - f(t)) \text{ and } |\widehat{M}'| < |M|.$$

Therefore, M is not a max-weight, min-cardinality matching and this is a contradiction. ◀

3.4 Tying it Together: Defining the Boolean Functions $f_1, \dots, f_k$

We are now equipped to define the functions $f_1, \dots, f_k \colon V(\mathcal{G}) \to \{0,1\}$ and complete the proof of Theorem 1.3. First, given $i \in [k]$ and $z \in V(\mathcal{G}) \setminus V(\mathcal{H}_i)$, we say that z is *below* $\mathcal{H}_i$ if there exists $y \in V(\mathcal{H}_i)$ for which $z \prec y$, and z is *above* $\mathcal{H}_i$ if there exists $x \in V(\mathcal{H}_i)$ for which $x \prec z$. Since $\mathcal{H}_i$ is the (S_i, T_i)-sweeping graph, by item 2 of Claim 3.2, vertex z cannot be both below and above $\mathcal{H}_i$, simultaneously. Second, given $z \in V(\mathcal{H}_i)$, we define the set $T_i(z) = \{t \in T_i \colon z \preceq t\}$. Note that by item 1 of Claim 3.2, $T_i(z) \neq \emptyset$ for all $z \in V(\mathcal{H}_i)$, and so the quantity $\max_{t\in T_i(z)} f(t)$ is always well-defined.

▶ **Definition 3.11.** *For each $i \in [k]$, define the function $f_i \colon V(\mathcal{G}) \to \{0,1\}$ as follows. For every $z \in V(\mathcal{G})$,*

$$f_i(z) = \begin{cases} 1, & \textit{if } z \in V(\mathcal{H}_i) \text{ and } f(z) > \max_{t\in T_i(z)} f(t), \\ 0, & \textit{if } z \in V(\mathcal{H}_i) \text{ and } f(z) \leq \max_{t\in T_i(z)} f(t), \\ 1, & \textit{if } z \notin V(\mathcal{H}_i) \text{ and } z \text{ is above } \mathcal{H}_i, \\ 0, & \textit{if } z \notin V(\mathcal{H}_i) \text{ and } z \text{ is not above } \mathcal{H}_i. \end{cases}$$

See Fig. 3 for an illustration of the values of f_i. We first prove item 1 of Theorem 1.3. Recall that $M(S_i) = T_i$ for all $i \in [k]$. Let $M_i = M|_{S_i}$ denote the matching M restricted to S_i. Consider $x \in S_i$. By Lemma 3.10, $f(x) > f(y)$ for all $y \in T_i$ such that $x \prec y$. Thus, $f(x) > \max_{t\in T_i(x)} f(t)$ and so $f_i(x) = 1$. Now consider $y \in T_i$. Observe that $y \in T_i(y)$. Thus, clearly, $f(y) \leq \max_{t\in T_i(y)} f(t)$, and so $f_i(y) = 0$. Therefore, $f_i(x) = 1$ for all $x \in S_i$ and $f_i(y) = 0$ for all $y \in T_i$. In particular, $f_i(x) = 1 > 0 = f_i(M(x))$ for all $x \in S_i$ and so M_i is a matching in the violation graph of f_i. Thus, $\varepsilon(f_i) \geq \frac{|M_i|}{|V(\mathcal{G})|}$ for all $i \in [k]$. Then

$$\sum_{i=1}^{k} \varepsilon(f_i) \geq |V(\mathcal{G})|^{-1} \sum_{i=1}^{k} |M_i| = |V(\mathcal{G})|^{-1} \cdot |M| \geq |V(\mathcal{G})|^{-1} \cdot \frac{\varepsilon(f) \cdot |V(\mathcal{G})|}{2} = \frac{\varepsilon(f)}{2}$$

by the above argument and Fact 3.9. Thus, item 1 of Theorem 1.3 holds.

To prove item 2 of Theorem 1.3, we need to show that, for all $i \in [k]$, the following hold:

$$\mathcal{S}_{f_i}^- \subseteq E(\mathcal{H}_i) \text{ and } \mathcal{S}_{f_i}^- \subseteq \mathcal{S}_f^-.$$

We first prove that $\mathcal{S}_{f_i}^- \subseteq E(\mathcal{H}_i)$. Consider an edge $(x,y) \in E(\mathcal{G}) \setminus E(\mathcal{H}_i)$. We need to show that $f_i(x) \leq f_i(y)$. First, observe that if both $x, y \in V(\mathcal{H}_i)$, then by item 3 of Claim 3.2, we have $(x,y) \in E(\mathcal{H}_i)$. Thus, we only need to consider the following three cases. Recall that $f_i(x), f_i(y) \in \{0,1\}$.

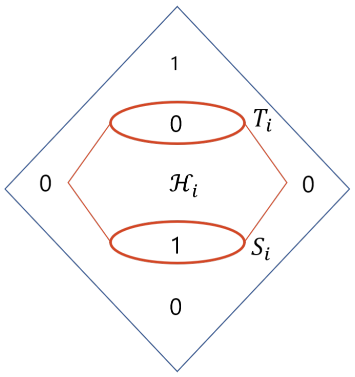

Figure 3 An illustration for the Boolean function f_i of Definition 3.11. The diamond represents the DAG $\mathcal{G}$ whose paths are directed from bottom to top. The hexagon represents the sweeping graph $\mathcal{H}_i = \mathcal{H}(S_i, T_i)$. The value of f_i is 1 for the vertices in S_i and 0 for the vertices in T_i. For vertices outside of $\mathcal{H}_i$, its value is 1 for the vertices that are above $\mathcal{H}_i$ and 0 for all other vertices.

1. $x \in V(\mathcal{H}_i)$, $y \notin V(\mathcal{H}_i)$: In this case, y is above $\mathcal{H}_i$, and so $f_i(y) = 1$. Thus, $f_i(x) \le f_i(y)$.
2. $x \notin V(\mathcal{H}_i)$, $y \in V(\mathcal{H}_i)$: In this case, x is below $\mathcal{H}_i$, and so x is *not* above $\mathcal{H}_i$ by item 2 of Claim 3.2. Thus, $f_i(x) = 0$, and so $f_i(x) \le f_i(y)$.
3. $x \notin V(\mathcal{H}_i)$, $y \notin V(\mathcal{H}_i)$: If x is above $\mathcal{H}_i$, then y is above $\mathcal{H}_i$ as well, and so $f_i(x) = f_i(y) = 1$. Otherwise, x is *not* above $\mathcal{H}_i$ and so $f_i(x) = 0$. Thus, $f_i(x) \le f_i(y)$.

Therefore, $S^-_{f_i} \subseteq E(\mathcal{H}_i)$.

We now prove that $\mathcal{S}^-_{f_i} \subseteq \mathcal{S}^-_f$. Consider an edge $(x, y) \in \mathcal{S}^-_{f_i}$. Then $f_i(x) = 1$ and $f_i(y) = 0$. Since $S^-_{f_i} \subseteq E(\mathcal{H}_i)$, we get $(x, y) \in E(\mathcal{H}_i)$ and so $x, y \in V(\mathcal{H}_i)$. By definition of the functions f_i, it holds that $f(x) > \max_{t \in T_i(x)} f(t)$ and $f(y) \le \max_{t \in T_i(y)} f(t)$. Since $x \prec y$, then $T_i(y) \subseteq T_i(x)$, because all vertices reachable from y are also reachable from x. Therefore,

$$f(x) > \max_{t \in T_i(x)} f(t) \ge \max_{t \in T_i(y)} f(t) \ge f(y).$$

Thus, $f(x) > f(y)$, and so $(x, y) \in \mathcal{S}^-_f$. As a result, $S^-_{f_i} \subseteq \mathcal{S}^-_f$ and item 2 of Theorem 1.3 holds. This concludes the proof of Theorem 1.3.

4 Testing Monotonicity of Real-Valued Functions

In this section, we prove Theorem 1.6. Some details have been omitted from this version. The omitted portion can be found in Section 4.3 of the full version [11]. We show that the tester of [32] for Boolean functions can be employed to test monotonicity of real-valued functions. The tester is simple: it queries two comparable vertices x and y and rejects if the pair exhibits a violation to monotonicity for f. The tester tries different values τ for the distance between x and y, that is, the number of coordinates on which they differ. The key step in the analysis of [32] (and in our analysis) is to show that for some choice of τ, the tester will detect a violation to monotonicity with high enough probability. The extra factor of r in the query complexity of our tester arises because we are forced to choose τ which is a factor of $(r - 1)$ smaller than for the Boolean case. Intuitively, the reason for this is that as the walk length τ increases, the probability that the function value stays below a certain threshold decreases. We make this precise in Section 4.2.

We first define the distribution from which the tester samples x and y. Following this, we present the tester as Algorithm 2. Let p denote the largest integer for which $2^p \le \sqrt{d/\log d}$. In Algorithm 2, we sample pairs of vertices at distance τ, where τ ranges over the powers of two up to 2^p.

▶ **Definition 4.1** (Pair Test Distribution). *Given parameters $b \in \{0,1\}$ and a positive integer τ, define the following distribution $\mathcal{D}_{\texttt{pair}}(b,\tau)$ over pairs $(x,y) \in (\{0,1\}^d)^2$. Sample $\boldsymbol{x}$ uniformly from $\{0,1\}^d$. Let $\boldsymbol{S} = \{i \in [d] \colon \boldsymbol{x}_i = b\}$. If $\tau > |\boldsymbol{S}|$, then set $\boldsymbol{y} = \boldsymbol{x}$. Otherwise, sample a uniformly random set $\boldsymbol{T} \subseteq \boldsymbol{S}$ of size $|\boldsymbol{T}| = \tau$. Obtain $\boldsymbol{y}$ by setting $\boldsymbol{y}_i = 1 - \boldsymbol{x}_i$ if $i \in \boldsymbol{T}$ and $\boldsymbol{y}_i = \boldsymbol{x}_i$ otherwise.*

Algorithm 2 Monotonicity Tester for $f \colon \{0,1\}^d \to \mathbb{R}$.

```
Require: Parameters $\varepsilon \in (0,1)$, dimension $d$, and image size $r$; oracle access to function
    $f \colon \{0,1\}^d \to \mathbb{R}$.
1: for all $b \in \{0,1\}$ and $\tau \in \{1,2,4,\dots,2^p\}$ do
2:     repeat $\widetilde{O}\left(\min\left(\frac{r\sqrt{d}}{\varepsilon^2}, \frac{d}{\varepsilon}\right)\right)$ times:
3:         Sample $(\boldsymbol{x},\boldsymbol{y}) \sim \mathcal{D}_{\texttt{pair}}(b,\tau)$.
4:         if $b = 0$ and $f(\boldsymbol{x}) > f(\boldsymbol{y})$ then reject.        ▷ if $b = 0$ then $\boldsymbol{x} \preceq \boldsymbol{y}$
5:         if $b = 1$ and $f(\boldsymbol{x}) < f(\boldsymbol{y})$ then reject.        ▷ if $b = 1$ then $\boldsymbol{x} \succeq \boldsymbol{y}$
6: accept.
```

Our tester only uses comparisons between function values, not the values themselves. Thus, for the purposes of our analysis we can consider functions with the range $[r]$ w.l.o.g.

When $\tau = 1$, the algorithm is simply sampling edges from the d-dimensional hypercube. The distribution from which we sample is not the uniform distribution on edges, but following an argument from [32], we can assume that for $\tau = 1$, our tester has the same guarantees as the edge tester.

The choice of the distance parameter τ for which the rejection probability of the tester is high depends on the existence of a certain "good" bipartite subgraph of violated edges. Our analysis differs from the analysis of [32] both in how we obtain the "good" subgraph of violated edges and in the choice of the optimal distance parameter τ.

We extend the following definitions from [32]. Let $G(A,B,E_{AB})$ denote a directed bipartite graph with vertex sets A and B and all edges in E_{AB} *directed* from A to B.

▶ **Definition 4.2** ((K,Δ)-Good Graphs). *A directed bipartite graph $G(A,B,E_{AB})$ is (K,Δ)-good if for X,Y such that either $X = A$, $Y = B$ or $X = B$, $Y = A$, we have: (a) $|X| = K$. (b) Vertices in X have degree exactly Δ. (c) Vertices in Y have degree at most 2Δ. The graph G is (K,Δ)-left-good if $X = A$ and (K,Δ)-right-good if $X = B$.*

The *weight* of $x \in \{0,1\}^d$, denoted by $|x|$, is the number of coordinates of x with value 1.

▶ **Definition 4.3** (Persistence). *Given $f \colon \{0,1\}^d \to [r]$ and an integer $\tau \in \left[1, \sqrt{\frac{d}{\log d}}\right]$, a vertex $x \in \{0,1\}^d$ of weight in the range $\frac{d}{2} \pm O(\sqrt{d \log d})$ is* τ-right-persistent *for f if*

$$\Pr_{\boldsymbol{y}}[f(\boldsymbol{y}) \le f(x)] > \frac{9}{10},$$

where $\boldsymbol{y}$ is obtained by choosing a uniformly random set $\boldsymbol{T} \subset \{i \in [d] \colon x_i = 0\}$ of size τ and setting $\boldsymbol{y}_i = 1$ if $i \in \boldsymbol{T}$ and $\boldsymbol{y}_i = x_i$ otherwise[4]*. We define τ-left-persistence symmetrically.*

We use the following technical claim implicitly shown in the analysis of the tester of [32].

[4] Note that $\tau \ge |\{i \in [d] \colon x_i = 0\}|$ by our assumption on x and τ.

▷ Claim 4.4 ([32]). Suppose there exists a (K, Δ)-right-good subgraph $G(A, B, E_{AB})$ of the directed d-dimensional hypercube, such that (a) $E_{AB} \subseteq \mathcal{S}_f^-$, (b) $K\sqrt{\Delta} = \Theta(\frac{\varepsilon(f)\cdot 2^d}{\log d})$, and (c) at least $\frac{99}{100}|B|$ of the vertices in B are $(\tau' - 1)$-right-persistent for some τ' such that $\tau' \cdot \Delta \ll d$. Then there exists a constant $C' > 0$, such that for $(\boldsymbol{x}, \boldsymbol{y}) \sim \mathcal{D}_{\texttt{pair}}(0, \tau')$,

$$\Pr_{\boldsymbol{x},\boldsymbol{y}}[f(\boldsymbol{x}) > f(\boldsymbol{y})] \geq \frac{C' \cdot \tau'}{d} \cdot \frac{K}{2^d} \cdot \Delta.$$

The analogous claim holds given a (K, Δ)-left-good subgraph with many $(\tau'-1)$-left-persistent vertices in A and $(\boldsymbol{x}, \boldsymbol{y})$ drawn from $\mathcal{D}_{\texttt{pair}}(1, \tau')$.

In Section 4.1, we prove Lemma 4.6 which obtains a good subgraph for f satisfying conditions (a) and (b) of Claim 4.4. In Section 4.2, we prove Lemma 4.8 which gives an upper bound on the fraction of non-persistent vertices, enabling us to satisfy condition (c). The remainder of the proof of Theorem 1.6 is deferred to the full version [11]. In particular, in Section 4.3 of the full version, we use Lemma 4.6 and Lemma 4.8 to show that the conditions of Claim 4.4 are satisfied and then use this to prove Theorem 1.6.

4.1 Existence of a Good Bipartite Subgraph

In this section, we prove Lemma 4.6 on the existence of good bipartite subgraphs for real-valued functions, which was proved in [32] for the special case of Boolean functions. This lemma crucially relies on our isoperimetric inequality for real-valued functions (Theorem 1.2). We first state (without proof) a combinatorial result of [32], which we need for our lemma.

▶ **Lemma 4.5** (Lemma 6.5 of [32]). *Let $G(A, B, E_{AB})$ be a directed bipartite graph whose vertices have degree at most 2^s. Suppose in addition, that for any 2-coloring of its edges* $\texttt{col} : E_{AB} \to \{\text{red}, \text{blue}\}$ *we have*

$$\sum_{x\in A} \sqrt{\deg_{\text{red}}(x)} + \sum_{y\in B} \sqrt{\deg_{\text{blue}}(y)} \geq L, \tag{9}$$

where $\deg_{\text{red}}(x)$ *denotes the number of red edges incident on x and* $\deg_{\text{blue}}(y)$ *denotes the number of blue edges incident on y. Then $G(A, B, E_{AB})$ contains a subgraph that is (K, Δ)-good with $K\sqrt{\Delta} \geq \frac{L}{8s}$.*

We can now generalize Lemma 7.1 of [32].

▶ **Lemma 4.6.** *For all functions $f\colon \{0,1\}^d \to \mathbb{R}$, there exists a subgraph $G(A, B, E_{AB})$ of the directed, d-dimensional hypercube which is (K, Δ)-good, where $K\sqrt{\Delta} = \Theta(\frac{\varepsilon(f)\cdot 2^d}{\log d})$ and $E_{AB} \subseteq \mathcal{S}_f^-$.*

Proof. Our proof relies on Lemma 4.5. Condition (9) is clearly reminiscent of the isoperimetric inequality in Theorem 1.2. We want to partition the vertices in $\{0,1\}^d$ into sets A and B such that all the violated edges are directed from A to B and apply Theorem 1.2 to the resulting graph. In addition, we want (9) to hold for a big enough value of L. In the Boolean case, we can simply partition the vertices by function values. In contrast, for real-valued functions, a vertex $x \in \{0,1\}^d$ can be incident on both incoming and outgoing violated edges. To overcome this challenge we resort to the bipartiteness of the directed hypercube, where each edge is between a vertex with an odd weight and a vertex with an even weight. Partition $\mathcal{S}_f^-$ into two sets:

$$E_0 = \{(x, y) \in \mathcal{S}_f^- : |x| \text{ is even}\};$$

$$E_1 = \{(x, y) \in \mathcal{S}_f^- : |x| \text{ is odd}\}.$$

For $j \in \{0, 1\}$, let V_j and W_j denote the set of lower and upper endpoints, respectively, of the edges in E_j. We consider the two subgraphs $G_j(V_j, W_j, E_j)$ for $j \in \{0, 1\}$. Notice that the vertices in $V_0 \cup W_1$ have even weight and the vertices in $V_1 \cup W_0$ have odd weight. Obviously, V_0 and W_1 may not be disjoint, and similarly V_1 and W_0 may not be disjoint, and thus G_0 and G_1 may not be vertex-disjoint.

We quickly explain why we cannot simply use Lemma 4.5 with either G_0 or G_1. Fix a 2-coloring of the edges $E_0 \cup E_1$. By averaging, one of the graphs will have a high enough contribution to left-hand side of the isoperimetric inequality of Theorem 1.2. Assume this graph is G_0. As a result, condition (9) will hold for G_0 with $L = \Omega(\varepsilon \cdot 2^d)$. However, one cannot guarantee that condition (9) holds for *all* possible colorings of the edges of G_0. Our construction below describes how to combine G_0 and G_1 so that we can jointly "feed" them into Lemma 4.5.

We construct copies $\widehat{G}_0$ and $\widehat{G}_1$ of G_0 and G_1, so that $\widehat{G}_0$ contains a vertex labelled $(x, 0)$ for each vertex x of G_0, and $\widehat{G}_1$ contains a vertex $(x, 1)$ for each vertex x of G_1. For each edge (x, y) in G_0 we add an edge from $(x, 0)$ to $(y, 0)$ in $\widehat{G}_0$. We do the same for the edges of G_1. Note that each edge of $\mathcal{S}_f^-$ has exactly one copy, either in $\widehat{G}_0$ or $\widehat{G}_1$.

Let $\widehat{G}(\widehat{V}, \widehat{W}, \mathcal{S}_f^-)$ denote the union of the two vertex-disjoint graphs $\widehat{G}_0$ and $\widehat{G}_1$. That is,

$$\widehat{V} = \{(x, 0) \mid x \in V_0\} \cup \{(x, 1) \mid x \in V_1\},$$
$$\widehat{W} = \{(y, 0) \mid y \in W_0\} \cup \{(y, 1) \mid y \in W_1\}.$$

All the edges of $\widehat{G}$ are directed from $\widehat{V}$ to $\widehat{W}$. Although imprecise, we think of the edges of $\widehat{G}$ as $\mathcal{S}_f^-$, since each edge in $\mathcal{S}_f^-$ has exactly one copy in $\widehat{G}$.

Consider a 2-coloring $\texttt{col}\colon \mathcal{S}_f^- \to \{\text{red}, \text{blue}\}$. Observe that

$$\begin{aligned}\sum_{(x,\cdot)\in\widehat{V}} \sqrt{I_{f,\text{red}}^-(x)} + \sum_{(y,\cdot)\in\widehat{W}} \sqrt{I_{f,\text{blue}}^-(x)} &= \sum_{x\in V_0\cup V_1} \sqrt{I_{f,\text{red}}^-(x)} + \sum_{y\in W_0\cup W_1} \sqrt{I_{f,\text{blue}}^-(y)} \\ &= \sum_{\substack{x\in\{0,1\}^d \\ |x| \text{ is even}}} \sqrt{I_{f,\text{red}}^-(x)} + \sqrt{I_{f,\text{blue}}^-(x)} + \sum_{\substack{x\in\{0,1\}^d \\ |x| \text{ is odd}}} \sqrt{I_{f,\text{red}}^-(x)} + \sqrt{I_{f,\text{blue}}^-(x)} \\ &= \sum_{x\in\{0,1\}^d} \sqrt{I_{f,\text{red}}^-(x)} + \sum_{y\in\{0,1\}^d} \sqrt{I_{f,\text{blue}}^-(y)} \geq C \cdot \varepsilon(f) \cdot 2^d,\end{aligned}$$

where the inequality holds by Theorem 1.2.

By construction, $I_{f,\text{red}}^-(x) = \deg_{\text{red}}((x, \cdot))$ for all $(x, \cdot) \in \widehat{V}$ and $I_{f,\text{blue}}^-(y) = \deg_{\text{blue}}((y, \cdot))$ for all $(y, \cdot) \in \widehat{W}$. We have that condition (9) of Lemma 4.5 holds with $L = C \cdot \varepsilon(f) \cdot 2^d$. Thus, $\widehat{G}$ contains a subgraph $G_{\text{good}}(A, B, E_{AB})$ that is (K, Δ)-good with $K\sqrt{\Delta} \geq \frac{L}{8 \log d}$. Without loss of generality, assume $G_{\text{good}}(A, B, E_{AB})$ is (K, Δ)-right-good.

Let $G_{\text{good},0} = (A_0, B_0, E_{A_0B_0})$ denote the subgraph of G_{good} lying in $\widehat{G}_0$ and let $G_{\text{good},1} = (A_1, B_1, E_{A_1B_1})$ denote the subgraph of G_{good} lying in $\widehat{G}_1$. Since $B_0 \cap B_1 = \emptyset$, we know that either $|B_0| \geq K/2$ or $|B_1| \geq K/2$. Suppose $|B_0| \geq K/2$. Moreover, since $\widehat{G}_0$ and $\widehat{G}_1$ are vertex-disjoint subgraphs, the degree of a vertex of $A_0 \cup B_0$ in $G_{\text{good},0}$ is the same its degree in G_{good}. Thus, $G_{\text{good},0}$ is a $(K/2, \Delta)$-right-good subgraph of the d-dimensional directed hypercube for which $\frac{K}{2}\sqrt{\Delta} \geq \frac{L}{16 \log d}$.

By removing some vertices from B_0, and redefining K if necessary, we may assume that $K\sqrt{\Delta} = \Theta\left(\frac{\varepsilon(f)\cdot 2^d}{\log d}\right)$. This completes the proof of Lemma 4.6. ◀

4.2 Bounding the Number of Non-Persistent Vertices

We prove Lemma 4.8 that bounds the number of non-persistent vertices for a function f and a given distance parameter τ. All results in this section also hold for τ-left-persistence.

For a function $f\colon \{0,1\}^d \to \mathbb{R}$, we define I_f^- as $\frac{|\mathcal{S}_f^-|}{2^d}$.

▶ **Corollary 4.7** (Corollary of Theorem 6.6, Lemma 6.8 of [32]). *Consider a function $h\colon \{0,1\}^d \to \{0,1\}$ and an integer $\tau \in \left[1, \sqrt{\frac{d}{\log d}}\right]$. If $I_h^- \leq \sqrt{d}$ then*

$$\Pr_{\boldsymbol{x}\sim\{0,1\}^d}\left[\boldsymbol{x} \textit{ is not } \tau\textit{-right-persistent for } h\right] = O\left(\frac{\tau}{\sqrt{d}}\right). \tag{10}$$

We generalize the above result to functions with image size $r \geq 2$.

▶ **Lemma 4.8.** *Consider a function $f\colon \{0,1\}^d \to [r]$ and an integer $\tau \in \left[1, \sqrt{\frac{d}{\log d}}\right]$. If $I_f^- \leq \sqrt{d}$, then*

$$\Pr_{\boldsymbol{x}\sim\{0,1\}^d}\left[\boldsymbol{x} \textit{ is not } \tau\textit{-right-persistent for } f\right] = (r-1)\cdot O\left(\frac{\tau}{\sqrt{d}}\right).$$

Proof. For all $t \in [r]$, define the threshold function $h_t\colon \{0,1\}^d \to \{0,1\}$ as:

$$h_t(x) = \begin{cases} 1 & \text{if } f(x) > t, \\ 0 & \text{otherwise.} \end{cases}$$

Observe that for all $t \in [r]$, we have $\mathcal{S}_{h_t}^- \subseteq \mathcal{S}_f^-$, and thus $I_{h_t}^- \leq I_f^- \leq \sqrt{d}$. By Corollary 4.7, we have that (10) holds for $h = h_t$ for all $t \in [r]$. Next, we point out that a vertex $x \in \{0,1\}^d$ is τ-right-persistent for f if and only if x is τ-right-persistent for the Boolean function $h_{f(x)}$. Too see this, consider a vertex z such that $x \prec z$. First, note that $h_{f(x)}(x) = 0$. Second, note that $h_{f(x)}(z) = 1$ if and only if $f(z) > f(x)$ by definition of $h_{f(x)}$. Therefore, $f(z) \leq f(x)$ if and only if $h_{f(x)}(z) \leq h_{f(x)}(x)$. Finally, note that all vertices are persistent for h_r since $h_r(x) = 0$ for all $x \in \{0,1\}^d$. Using these observations, we have

$$\begin{aligned}
&\Pr_{\boldsymbol{x}\sim\{0,1\}^d}[\boldsymbol{x} \text{ is not } \tau\text{-right-persistent for } f] \\
&= \Pr_{\boldsymbol{x}\sim\{0,1\}^d}\left[\boldsymbol{x} \text{ is not } \tau\text{-right-persistent for } h_{f(\boldsymbol{x})}\right] \\
&\leq \Pr_{\boldsymbol{x}\sim\{0,1\}^d}[\exists t \in [r-1]\colon \boldsymbol{x} \text{ is not } \tau\text{-right-persistent for } h_t] \\
&\leq \sum_{t=1}^{r-1} \Pr_{\boldsymbol{x}\sim\{0,1\}^d}[\boldsymbol{x} \text{ is not } \tau\text{-right-persistent for } h_t] \\
&= \sum_{t=1}^{r-1} O\left(\frac{\tau}{\sqrt{d}}\right) = (r-1)\cdot O\left(\frac{\tau}{\sqrt{d}}\right),
\end{aligned}$$

where the second inequality is by the union bound and the last equality is due to the fact that (10) holds for all h_t, $t \in [r]$. ◀

References

1 Nir Ailon and Bernard Chazelle. Information theory in property testing and monotonicity testing in higher dimension. *Information and Computation*, 204(11):1704–1717, 2006.

2 Nir Ailon, Bernard Chazelle, Seshadhri Comandur, and Ding Liu. Estimating the distance to a monotone function. *Random Structures and Algorithms*, 31(3):371–383, 2007.

3 Tugkan Batu, Ronitt Rubinfeld, and Patrick White. Fast approximate *PCP*s for multidimensional bin-packing problems. *Information and Computation*, 196(1):42–56, 2005.

4 Aleksandrs Belovs. Adaptive lower bound for testing monotonicity on the line. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 31:1–31:10, 2018.

5 Aleksandrs Belovs and Eric Blais. A polynomial lower bound for testing monotonicity. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 1021–1032, 2016.

6 Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev. l_p-testing. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 164–173, 2014.

7 Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners. *SIAM J. Comput.*, 41(6):1380–1425, 2012.

8 Hadley Black, Deeparnab Chakrabarty, and C. Seshadhri. A $o(d)\cdot$ polylog n monotonicity tester for Boolean functions over the hypergrid $[n]^d$. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2133–2151, 2018.

9 Hadley Black, Deeparnab Chakrabarty, and C. Seshadhri. Domain reduction for monotonicity testing: A $o(d)$ tester for Boolean functions in d-dimensions. In Shuchi Chawla, editor, *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1975–1994, 2020.

10 Hadley Black, Deeparnab Chakrabarty, and C. Seshadhri. Directed isoperimetric theorems for boolean functions on the hypergrid and an $\widetilde{O}(n\sqrt{d})$ monotonicity tester. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, 2023.

11 Hadley Black, Iden Kalemaj, and Sofya Raskhodnikova. Isoperimetric inequalities for real-valued functions with applications to monotonicity testing. *CoRR*, abs/2011.09441, 2020. `arXiv:2011.09441`.

12 Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21(2):311–358, 2012.

13 Eric Blais, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Lower bounds for testing properties of functions over hypergrid domains. In *Proceedings, IEEE Conference on Computational Complexity (CCC)*, pages 309–320, 2014.

14 Mark Braverman, Subhash Khot, Guy Kindler, and Dor Minzer. Improved monotonicity testers via hypercube embeddings. In *Proceedings, Innovations in Theoretical Computer Science (ITCS)*, pages 25:1–25:24, 2023.

15 Jop Briët, Sourav Chakraborty, David García Soriano, and Ari Matsliah. Monotonicity testing and shortest-path routing on the cube. *Combinatorica*, 32(1):35–53, 2012.

16 Deeparnab Chakrabarty, Kashyap Dixit, Madhav Jha, and C. Seshadhri. Property testing on product distributions: Optimal testers for bounded derivative properties. *ACM Trans. on Algorithms*, 13(2):20:1–20:30, 2017.

17 Deeparnab Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 419–428, 2013.

18 Deeparnab Chakrabarty and C. Seshadhri. An optimal lower bound for monotonicity testing over hypergrids. *Theory of Computing*, 10:453–464, 2014.

19 Deeparnab Chakrabarty and C. Seshadhri. An $o(n)$ monotonicity tester for Boolean functions over the hypercube. *SIAM Journal on Computing*, 45(2):461–472, 2016.

20 Deeparnab Chakrabarty and C. Seshadhri. Adaptive Boolean monotonicity testing in total influence time. In *Proceedings, Innovations in Theoretical Computer Science (ITCS)*, pages 20:1–20:7, 2019.

21 Xi Chen, Anindya De, Rocco A. Servedio, and Li-Yang Tan. Boolean function monotonicity testing requires (almost) $n^{1/2}$ non-adaptive queries. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 519–528, 2015.

22 Xi Chen, Rocco A. Servedio, and Li-Yang Tan. New algorithms and lower bounds for monotonicity testing. In *Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 286–295, 2014.

23 Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond Talagrand functions: new lower bounds for testing monotonicity and unateness. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 523–536, 2017.

24 Kashyap Dixit, Sofya Raskhodnikova, Abhradeep Thakurta, and Nithin Varma. Erasure-resilient property testing. *SIAM Journal on Computing*, 47(2):295–329, 2018.

25 Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved testing algorithms for monotonicity. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 97–108, 1999.

26 Funda Ergun, Sampath Kannan, Ravi Kumar, Ronitt Rubinfeld, and Mahesh Viswanathan. Spot-checkers. *J. Comput. System Sci.*, 60(3):717–751, 2000.

27 Shahar Fattal and Dana Ron. Approximating the distance to monotonicity in high dimensions. *ACM Trans. on Algorithms*, 6(3):52:1–52:37, 2010.

28 Eldar Fischer. On the strength of comparisons in property testing. *Information and Computation*, 189(1):107–116, 2004.

29 Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In *Proceedings, ACM Symposium on Theory of Computing (STOC)*, pages 474–483, 2002.

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

31 Shirley Halevy and Eyal Kushilevitz. Testing monotonicity over graph products. *Random Structures and Algorithms*, 33(1):44–67, 2008.

32 Subhash Khot, Dor Minzer, and Muli Safra. On monotonicity testing and Boolean isoperimetric-type theorems. *SIAM Journal on Computing*, 47(6):2238–2276, 2018.

33 Eric Lehman and Dana Ron. On disjoint chains of subsets. *Journal of Combinatorial Theory, Series A*, 94(2):399–404, 2001.

34 Grigory A. Margulis. Probabilistic characteristics of graphs with large connectivity. *Problemy Peredachi Informatsii*, 10(2):101–108, 1974.

35 Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin Varma. Parameterized property testing of functions. *ACM Trans. Comput. Theory*, 9(4):17:1–17:19, 2018.

36 Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Erik Waingarten. Approximating the distance to monotonicity of Boolean functions. *Random Structures and Algorithms*, 60(2):233–260, 2022.

37 Ramesh Krishnan Pallavoor Suresh. *Improved Algorithms and New Models in Property Testing.* PhD thesis, Boston University, 2020.

38 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *J. Comput. Syst. Sci.*, 72(6):1012–1042, 2006.

39 Sofya Raskhodnikova. Monotonicity testing. *Masters Thesis, MIT*, 1999.

40 Michel Talagrand. Isoperimetry, logarithmic Sobolev inequalities on the discrete cube, and Margulis' graph connectivity theorem. *Geom. Func. Anal.*, 3(3):295–314, 1993.