Monotonicity testing over general poset domains

[Extended Abstract]

Eldar Fischer∗
Technion
Haifa, Israel
eldar@cs.technion.ac.il
Sofya Raskhodnikova†
LCS, MIT
Cambridge, MA 02139
sofya@mit.edu

Eric Lehman
LCS, MIT
Cambridge, MA 02139
e Lehman@mit.edu

Ronitt Rubinfeld
NEC Research Institute
Princeton, NJ
ronitt@research.nj.nec.com

Ilan Newman∗
University of Haifa
Haifa, Israel
ilan@cs.haifa.ac.il

Alex Samorodnitsky‡
Hebrew University
Jerusalem, Israel
salex@cs.huji.ac.il

ABSTRACT

The field of property testing studies algorithms that distinguish, using a small number of queries, between inputs which satisfy a given property, and those that are “far” from satisfying the property. Testing properties that are defined in terms of monotonicity has been extensively investigated, primarily in the context of the monotonicity of a sequence of integers, or the monotonicity of a function over the n-dimensional hypercube \(\{1, \ldots, m\}^n\). These works resulted in monotonicity testers whose query complexity is at most polylogarithmic in the size of the domain.

We show that in its most general setting, testing that Boolean functions are close to monotone is equivalent, with respect to the number of required queries, to several other testing problems in logic and graph theory. These problems include: testing that a Boolean assignment of variables is close to an assignment that satisfies a specific 2-CNF formula, testing that a set of vertices is close to one that is a vertex cover of a specific graph, and testing that a set of vertices is close to a clique.

We then investigate the query complexity of monotonicity testing of both Boolean and integer functions over general partial orders. We give algorithms and lower bounds for the general problem, as well as for some interesting special cases. In proving a general lower bound, we construct graphs with combinatorial properties that may be of independent interest.

∗Work done while at NEC Research Institute, Princeton, NJ.
†Supported by NTT grant. Some work done while at NEC Research Institute, Princeton, NJ.
‡Work done while at Institute for Advanced Study and at NEC Research Institute, Princeton, NJ. Supported in part by NSF (grant number CCR-9987845) and by a State of NJ grant.

Keywords

Algorithms, property testing, monotone functions.

1. INTRODUCTION

Property testing [24, 15] deals with a relaxation of decision problems where one is to determine whether an input satisfies a particular property or is far from satisfying it. This has recently become quite an active research area; see [23, 12] for surveys on the topic.

Monotonicity is a natural property of functions on posets. Given a poset \(P = (V, \leq_P)\) and a linear order \(L\), a function \(f : V \to L\) is monotone if \(f(u) \leq_f f(v)\) for all \(u, v \in V\) with \(u \leq_P u\). A function \(f\) is \(\epsilon\)-close to monotone if we can make \(f\) monotone by changing its value on at most an \(\epsilon\) fraction of the domain. Along with linearity and low-degree testing, monotonicity is one of the most studied properties in the context of property testing (see [10, 16, 8, 4, 2, 14, 11]).

Virtually all previous works deal with posets \(P\) that happen to be hypercubes of different sizes and dimensions, and provide 1-sided error testers whose complexity is at most polylogarithmic in the size of the domain. For the case where \(P\) is the linear order of size \(N\), and \(L\) is large enough, there is an optimal testing algorithm which uses \(O(\log N)\) time and queries [10]. The case when \(P\) is the n-dimensional hypercube \([M]^n\), \(M = \{1, \ldots, m\}\), where \(u = (u_1, \ldots, u_n) \leq v = (v_1, \ldots, v_n)\) if \(u_i \leq v_i\) for every \(i\), was first studied and shown to have monotonicity testers with query complexity \(O(\frac{n^2}{\epsilon^2} m^n |L|)\) by [16]. The most efficient known tester for this case has complexity \(O(\frac{n}{\epsilon^2} (\log m) (\log |L|))\) [8]. For Boolean functions, this complexity can be made independent of \(m\) [8].

Very little was known regarding lower bounds for monotonicity testing. For the case where \(P\) is the linear order of size \(N\), the lower bound in [11] which shows that the algorithm of [10] is tight gives the only previously known nonconstant lower bound for any monotonicity testing problem. In particular, no nonconstant lower bound was known for any poset when \(L = \{0, 1\}\), namely, for Boolean functions. The lower bound question for the case of the Boolean hypercube, as well as for general posets, remained unresolved.

Our Results. The main objective of this work is a systematic...
study of the query complexity of monotonicity testing on general posets. Our results fall into three categories: reductions between testing monotonicity and other testing problems, algorithms for testing monotonicity and lower bounds.

We begin by showing that monotonicity testing of Boolean functions over general posets is equivalent to the following three testing problems. The first is that of testing if a given assignment to Boolean variables is close to one that satisfies a fixed 2-CNF formula. The second is that of testing if a set of vertices (in a fixed) graph is close to a vertex cover. Here closeness captures the number of vertices that need to be added to make the set into a vertex cover. The third problem is that of testing if a set of vertices is close to a clique, where closeness refers to the number of vertices that need to be removed to make the set into a clique. These reductions provide additional motivation for studying monotonicity over general posets.

We next present an algorithm with an $O(\sqrt{|N|/\varepsilon})$ query complexity for testing monotonicity over general posets with $N$ elements, addressing an open problem posed by [8, 22]. This, in turn, yields $O(\sqrt{|N|/\varepsilon})$ query tests for all the above equivalent properties.

We then show that no non-adaptive test which makes only a polylogarithmic number of queries exists. Our main hardness result is a lower bound of $N^{\Omega(\log \log N)}$ queries for general Boolean non-adaptive monotonicity testing. This implies an adaptive lower bound of $\Omega(\log \log N)$. This addresses the open problem raised in [8, 22], for which no previous lower bound was known. It also shows that for the above equivalent properties, monotonicity testing is the “essential difficult part”.

To achieve our lower bound, we show that there is a graph with $N$ vertices that can be partitioned into $N^{\Omega((\log \log N)^{-1})}$ induced matchings of size $\varepsilon N$. The latter result is of independent interest: graphs with a similar property were constructed by Ruzsa and Szemerédi [25] to provide a lower bound for a Turán-like theorem. Recently they have been used by Håstad and Wigderson [18] for constructing better linearity tests.

Motivated by conjectures in graph theory, Meshulam [21] constructed similar graphs with different parameters. Our approach is different from all of the above.

We also present lower bounds for non-adaptive monotonicity tests over the (Boolean) $n$-dimensional hypercube. We prove an $\Omega(\sqrt{n})$ lower bound for 1-sided error algorithms, and an $\Omega(\log n)$ lower bound for 2-sided error algorithms. These results imply the corresponding adaptive lower bounds of $\Omega(\log n)$ and $\Omega(\log \log n)$, respectively. These are the first nontrivial lower bounds for the Boolean hypercube, for both 1-sided and 2-sided error algorithms, answering the open questions raised by the works of [16, 8, 22].

Finally, the question arises as to what other posets can be tested more efficiently than the general lower bound. We show that functions over certain types of posets have tests with a number of queries that is independent of the size of the poset. For Boolean functions, this includes posets whose Hasse diagrams are trees, posets having constant size antichains, and what we call “top-paralle” posets. For functions with arbitrary ranges, this applies to posets with a linear number of comparable element pairs. We also prove that for posets derived from graphs with bounded separators, monotonicity testing of functions with arbitrary ranges requires only a logarithmic number of queries.

**Organization.** Section 2 introduces basic definitions and general tools. Section 3 shows the equivalence to testing 2-CNF, vertex cover and clique. Section 4 provides the test for general posets. Section 5 prepares the tools for the lower bound for general posets, while the specific constructions are given in Section 6. Section 7 presents the lower bounds for monotonicity testing over the hypercube. Finally, Section 8 contains efficient algorithms for several special classes of posets.

### 2. PRELIMINARIES

**Property testing.** A property is a collection of functions with a fixed domain $V$ and a fixed range $L$. The distance between functions $f$ and $f'$, denoted by $\text{dist}(f, f')$, is the number of domain elements on which they differ. The distance $\text{dist}(f, P)$ of a function $f$ to a property $P$ is $\min_{f' \in P} \text{dist}(f, f')$. Its relative distance to $P$ is its distance to $P$ divided by the size of the domain. A function is $\varepsilon$-far from $P$ if its relative distance to $P$ is at least $\varepsilon$.

A property is $(\varepsilon, q)$-testable if there is a randomized algorithm that for every input function $f$ queries the values of $f$ on at most $q$ points of the domain, and with probability at least $\frac{2}{3}$ distinguishes between the case that $f$ has the property and the case that $f$ is $\varepsilon$-far from the property. The algorithm is referred to as an $(\varepsilon, q)$-test or simply an $\varepsilon$-test. An algorithm is non-adaptive if its queries do not depend on the answers to previous queries. An algorithm has $1$-sided error if it always accepts an input that has the property.

**Monotone functions and graph labelings.** Let $G = (V, E)$ be a directed graph. Let $f : V \to L$ be a labeling of $V$ with members of a linear order $L$. Then $f$ is monotone on $G$ if $f(v_i) \leq f(v_j)$ for all $(v_i, v_j) \in E$. The monotonicity property, denoted by $\text{mon}_G$, is the set of monotone labelings of $G$. If there is a directed path from $v_i$ to $v_j$ in $G$, we say that $v_i$ is below $v_j$ (or $v_j$ is above $v_i$) and denote it by $v_i \leq_G v_j$ (which is not an order relation in general). Every such pair of vertices of $G$ imposes a constraint $f(v_i) \leq f(v_j)$. A pair of vertices $(v_i, v_j)$ is violated if $v_i \leq_G v_j$ and $f(v_i) > f(v_j)$. Vertices $v_i$ and $v_j$ are equivalent in $G$ if $v_i \leq_G v_j$ and $v_j \leq_G v_i$, namely, if both are in the same strongly connected component.

Note that monotonicity of labelings of acyclic graphs corresponds to monotonicity of functions on posets. We often consider a special case of monotonicity restricted to Boolean functions or labelings (namely, with $L = \{0, 1\}$), which we call Boolean monotonicity.

**Handy lemmas for monotonicity testing.** A transitive closure of a graph $G = (V, E)$, denoted by $\text{TC}(G)$, is a graph $(V, E')$ where $(v_i, v_j)$ is in $E'$ if there is a directed path from $v_i$ to $v_j$ in $G$. Observe that two graphs with the same transitive closure give rise to the same monotonicity property $\text{mon}_G$.

**Lemma 1.** Let $f$ be a labeling for a graph $G(V, E)$. If $f$ is monotone on an induced subgraph $G' = (V', E')$ of $\text{TC}(G)$, then $f$’s distance to monotone is at most $|V' - V|$.
Proof. Assuming that \( f \) is monotone on an induced subgraph \( G' = (V', E') \) of TC \((G)\), we make \( f \) monotone on TC \((G)\) by relabeling only vertices in \( V - V' \).

Indeed, fix \( V' \) and let \( f|_{V'} \) be the partial labeling on \( V' \) that is monotone on \( G'(V', E') \). We extend \( f|_{V'} \) to \( V \) for one vertex \( v \in V - V' \) at a time, always keeping the partial labeling monotone on the induced current graph. We now show how to extend the domain of \( f \) by one vertex. Let \( v \in V - V' \) be a ‘minimal’ element in \( V - V' \) (namely, \( v \) is unreachable from any other vertex \( w \in V - V' \) that is not equivalent to \( v \)). Let \( T = \{ u \in V' \mid u \leq_G v \} \). We extend \( f \) to \( V \cup \{ v \} \) by letting \( f(v) \) be \( \max_{u \in T} \{ f(u) \} \) if \( T \neq \emptyset \) and the minimum value in the range otherwise. By transitivity, since \( f \) was monotone on \( V' \), the extended \( f \) is monotone on \( V \cup \{ v \} \).

Corollary 2. Let \( f \) be a labeling of \( G = (V, E) \). Then \( dist(f, mon_G) \) is equal to the minimum vertex cover of the graph of violated edges of TC \((G)\).

A matching in a graph is a collection of edges that share no common vertex. The next two lemmas relate a function’s distance to monotone to the number of edges it violates in the transitive closure of the graph. The first of them follows from Corollary 2 and the fact that the size of a maximum matching is at least 1/2 of the size of a minimum vertex cover.

Lemma 3. Let \( f \) be a labeling which is \( \varepsilon \)-far from monotone on a graph \( G \) with \( N \) nodes. Then TC \((G)\) has a matching of violated edges of size \( \varepsilon N/2\).

Lemma 4. Let \( f \) be a Boolean labeling which is \( \varepsilon \)-far from monotone over a graph \( G \) with \( N \) nodes. Then TC \((G)\) has a matching of violated edges of size \( \varepsilon N \).

Proof. Let \( P' \) be a poset of vertices in \( V \) with partial order defined by \( v \leq u \) if \( (v, u) \) is a violated pair in \( G \). Let \( A \subseteq V \) be a maximal antichain in \( P' \). Certainly, \( f \) is monotone on the subgraph of TC \((G)\) induced by \( A \), as \( A \) contains no violated pairs. Then by Lemma 1, \( dist(f, mon_G) \leq |V| - |A| \). By Dilworth’s theorem \([7]\), \( |A| \) is equal to the minimum number of disjoint chains that cover \( P' \). However, a chain in \( P' \) consists of at most two vertices as \( (v, u) \) and \( (u, v) \) cannot both be violated by a Boolean function. Hence, to cover \( |V| \) elements, at least \(|V| - |A|\) out of \( |A| \) chains have to be of length exactly two (otherwise, less than \(|V| \) elements are covered). This collection of at least \(|V| - |A| \geq dist(f, mon_G)\) disjoint chains of size two is a matching of violated pairs.

Reduction from monotonicity on general graphs to monotonicity on bipartite graphs. We now prove that testing monotonicity on arbitrary graphs is equivalent to testing monotonicity on bipartite DAG’s (which naturally correspond to posets).

Definition 1. For each directed graph \( G = (\{(u_1, \ldots, u_N)\}, E) \), let \( B_G \) be the bipartite graph \( \{(v_1, \ldots, v_N), (v'_1, \ldots, v'_N)\} \mid \{E_B\} = \{(v_i, v'_j) \mid v_j \text{ is reachable from } v_i \text{ in } E\} \). For each labeling \( f \) of \( G \), define the corresponding labeling \( f_B \) of \( B_G \) by \( f_B(v_i) = f_B(v'_j) = f(v_i) \).

Note that \( B_G \) is a transitive closed DAG with \( 2N \) vertices and the same number of edges as TC \((G)\).

Claim 5. Let \( f \) be a labeling on a graph \( G \). Then \( dist(f, mon_G) \leq dist(f_B, mon_{B_G}) \).

Proof. The proof is straightforward, and we omit it.

In fact \( dist(f, mon_G) = dist(f_B, mon_{B_G}) \). We postpone the proof of this stronger claim to the full version of the paper.

Theorem 6. If \( mon_{B_G} \) is \((\varepsilon/2, q)\)-testable for a graph \( G \) then \( mon_G \) is \((\varepsilon, q)\)-testable. The reduction preserves 1-sided error: a 1-sided test for \( mon_{B_G} \) gives a 1-sided test for \( mon_G \).

Proof. Let \( f \) be a labeling of \( G \) and let \( B_G \) be the associated graph with labeling \( f_B \) as defined above. By Claim 5, \( dist(f, mon_G) = dist(f_B, mon_{B_G}) \). If \( f \) is \( \varepsilon \)-far from monotone on \( mon_{B_G} \) then \( f_B \) is \( \varepsilon/2 \)-far from monotone on \( B_G \) because \( B_G \) has twice as many nodes. A test for \( G \) on input \( f \) can simulate a test for \( B_G \) on input \( f_B \), asking at most the same number of queries.

3. EQUIVALENCE OF BOOLEAN MONOTONICITY AND OTHER PROBLEMS

Testing 2-CNF assignments is equivalent to testing Boolean monotonicity. Recall that a Boolean formula is in conjunctive normal form (CNF) if it consists of clauses joined by AND, where every clause is an \( \lor \) of literals. (A literal is a Boolean variable or a negated Boolean variable.) If all clauses have two literals, the formula is a 2-CNF. Let \( \phi \) be a 2-CNF formula with variables \( X_1, \ldots, X_N \), and let \( f : \{X_1, \ldots, X_N\} \to \{0, 1\} \) be an assignment to its variables. The property SAT(\( \phi \)) is the set of satisfying assignments of \( \phi \). We show that the testability of SAT(\( \phi \)) is equivalent to the testability of Boolean monotonicity on digraphs.

Theorem 7. For every graph \( G \) with \( N \) vertices, there is a corresponding 2-CNF \( \phi_G \) on \( N \) variables such that if SAT(\( \phi_G \)) is \((\varepsilon, q)\)-testable then \( mon_{\phi_G} \) is also \((\varepsilon, q)\)-testable for Boolean labelings. The reduction preserves 1-sided error.

Proof. Let \( G = (V, E) \) be a digraph. With each \( v \in V \) associate a Boolean variable \( x_v \). Define the 2-CNF formula \( \phi_G \) on the set of variables \( X = \{x_v \mid v \in V\} \) as follows: for each edge \((u, v) \in E\), form the clause \( \neg x_u \lor \neg x_v \). A Boolean labeling \( f \) of \( G \) is a 1-sided test for \( \phi \) if \( f(x_v) = f(v) \). Clearly, \( dist(f, mon_G) = dist(f, SAT(\phi_G)) \). Thus, a test for SAT(\( \phi_G \)) can be used as a test for \( mon_G \).

Theorem 8. For every 2-CNF \( \phi \) on \( N \) variables, there is a corresponding graph \( G_\phi \) with 2\( N \) vertices such that if \( mon_{\phi_G} \) is \((\varepsilon/2, q)\)-testable for Boolean labelings then SAT(\( \phi \)) is \((\varepsilon, q)\)-testable. The reduction preserves 1-sided error.

Proof. Let \( \phi \) be a 2-CNF formula on a set \( X \) of \( N \) variables. (If \( \phi \) is unsatisfiable, it has a trivial test that rejects all assignments). With each Boolean variable \( x \in X \),
associate two vertices \( v_x \) and \( v_y \) that represent literals corresponding to \( x \). We use the convention \( v_x = \overline{v_y} \) and \( v_y = \overline{v_x} \). Define the implication graph, \( G_\phi \), on the set of the corresponding \( 2N \) vertices, as follows: for each clause \( x \lor y \), where \( x \) and \( y \) are literals, add edges \((v_x, v_y)\) and \((v_y, v_x)\). For any edge \((u, v)\) call edge \((\overline{v}, \overline{u})\) its dual edge. Note that dual edges appear in the implication graph in pairs, with the exception of edges of the form \([u, \overline{u}]\), which are dual to themselves.

Let \( f : X \to \{0, 1\} \) be an assignment to \( \phi \). Define the associated Boolean labeling \( f_G \) of \( G_\phi \) by \( f_G(v_x) = f(x) \) for all literals \( x \). If \( f \) satisfies \( \phi \), the corresponding labeling \( f_G \) is monotone on \( G_\phi \). It remains to prove that \( dist(f, SAT(\phi)) \leq dist(f_G, mon_{G_\phi}) \). To show this we transform \( f \) into a satisfying assignment for \( \phi \) by changing at most \( dist(f_G, mon_{G_\phi}) \) variable assignments. To this end, a Boolean labeling of an implication graph is called negation-compliant if \( v_x \) and \( v_y \) have different labels for all literals \( x \). Note that every negation-compliant labeling of \( G_\phi \) has a corresponding assignment to \( \phi \). Furthermore, if \( f \) is monotone and negation-compliant for \( G_\phi \) then the corresponding assignment \( \phi \) for \( \phi \), given by \( f(x) = f_G(v_x) \) for every literal \( x \), is a satisfying assignment for \( \phi \).

Note that for every literal \( x \), \( v_x \) and \( v_y \) are in the same strongly connected component because \( \phi \) is satisfiable. Also, if \( v_y \) is equivalent to \( v_x \) in \( G_\phi \) then \( \overline{v_y} \) is equivalent to \( \overline{v_x} \).

The following algorithm transforms \( f_G \) into a nearby monotone, negation-compliant labeling.

1. Convert \( f_G \) to a nearest monotone assignment \( \tilde{f}_G \) on \( G_\phi \). \( f_G \) is not necessarily negation-compliant.)

2. While \( G_\phi \) has nodes \( v_x \) with \( \tilde{f}_G(v_x) = \tilde{f}_G(\overline{v_x}) = 0 \):
   Find a maximal \( v_x \) (with respect to \( G_\phi \)) among those with \( \tilde{f}_G(v_x) = \tilde{f}_G(\overline{v_x}) = 0 \). Change \( \tilde{f}_G(v_x) \) to 1 for all \( v_y \) that are equivalent to \( v_x \) (including \( v_x \) itself).

3. While \( G_\phi \) has nodes \( v_x \) with \( \tilde{f}_G(v_x) = \tilde{f}_G(\overline{v_x}) = 1 \):
   Find a minimal \( v_y \) among those with \( \tilde{f}_G(v_y) = \tilde{f}_G(\overline{v_y}) = 1 \). Change \( \tilde{f}_G(v_y) \) to 0 for all \( v_x \) that are equivalent to \( v_y \) (including \( v_y \) itself).

First, we show that the resulting labeling \( \tilde{f}_G \) is monotone on \( G_\phi \). Indeed, \( f_G \) is monotone after step 1. Since it is monotone, nodes in the same strongly connected component (i.e., equivalent nodes with respect to \( G_\phi \)) have the same labels. Hence, after each change in step 2, equivalent nodes still have the same labels. Suppose \( \tilde{f}_G \) is monotone on \( G \) before some iteration of step 2 and is not monotone after it. Then some edge \((v_x, v_y)\) is violated by changing \( \tilde{f}(v_x) \) to 1. Then \( \tilde{f}_G(v_y) \) was 0 both before and after this iteration, and \( v_y \) is not equivalent to \( v_x \). Since \( v_y \geq v_x \), it must be that \( \tilde{f}_G(v_y) = 1 \) (otherwise, \( v_y \) would have changed before \( v_x \)). But then the dual edge \((\overline{v_y}, \overline{v_x})\) is violated before the iteration, giving a contradiction.

Similarly, if \( \tilde{f}_G \) is monotone on \( G \) before some iteration of step 3 then it is monotone after it.

Secondly, the resulting labeling \( \tilde{f}_G \) is negation-compliant because step 2 relabels all nodes \( v_x \) with \( \tilde{f}(v_x) = \tilde{f}(\overline{v_x}) = 0 \) and step 3 relabels all nodes with \( \tilde{f}(v_x) = \tilde{f}(\overline{v_x}) = 1 \).

Finally, let \( \tilde{f} \) be the assignment to \( X \) with \( \tilde{f}(x) = \tilde{f}_G(v_x) \) for every literal \( x \in X \). By the remarks above, \( \tilde{f} \) is a satisfying assignment for \( \phi \). It is not hard to show that \( dist(f, \tilde{f}) \leq dist(f_G, mon_{G_\phi}) \).

Other testing problems equivalent to 2-CNF testing. Recall that a monotone CNF is a CNF with only positive literals. We prove that testing 2CNF is equivalent to testing monotone 2CNF. Since we have shown that 2-CNF testing is equivalent to testing Boolean monotonicity over general graphs, which is equivalent to testing Boolean monotonicity on special kind of bipartite graphs, it is enough to prove the following theorem.

**Theorem 9.** Let \( G = (X, Y; E) \) be a bipartite digraph with all edges directed from \( X \) to \( Y \) and \( |X| = |Y| = N \). For each \( G \) there is a corresponding monotone 2-CNF \( \phi_G \) on \( N \) variables s.t. if \( SAT(\phi_G) \) is \((\varepsilon, \eta)\)-testable then monotonicity of Boolean functions over \( G \) is also \((\varepsilon, \eta)\)-testable.

**Proof.** Associate a variable \( z_v \) with every node \( v \in X \cup Y \). Each node \( y \in Y \) is represented by \( z_y \), while each node \( x \) in \( X \) is represented by \( \overline{z_x} \). Define a Boolean formula \( \phi_G \) on the set of variables \( Z = \{z_v \mid v \in X \cup Y\} \) as follows: form a clause \( \{z_v \lor z_y\} \) for each edge \((x, y) \in E\). A Boolean labeling \( f \) of \( G \) defines an assignment \( \tilde{f} \) for \( Z \) by \( \tilde{f}(z_v) = 1 - f(x) \) if \( x \in X \) and \( f(z_v) = f(y) \) if \( y \in Y \). Then an edge \((x, y) \) is violated if and only if the corresponding clause \( \{z_v \lor z_y\} \) is unsatisfied. Therefore, \( dist(f, mon_{G_\phi}) = dist(\tilde{f}, SAT(\phi_G)) \), and each test for \( \phi_G \) can be used as a test for \( mon_{G_\phi} \).

Let \( U = (V, E) \) be an undirected graph. For a \( S \subseteq V \), let \( f_S : V \to \{0, 1\} \) be a characteristic function of \( S \), i.e. \( f_S(v) = 1 \) if and only if \( v \in S \). A vertex cover of \( U \) is a subset of the vertices where every edge of \( U \) touches one of those vertices. A clique in \( U \) is a subset of the vertices that induces a complete graph in \( U \). The property \( VC(U) \) is the set of all characteristic functions \( f_S \) such that \( S \) is a vertex cover of \( U \). Similarly, the property \( CLIQUE(U) \) is the set of all characteristic functions \( f_S \) such that \( S \) is clique of \( U \).

**Theorem 10.** The following statements are equivalent:

- \( SAT(\phi) \) is \((\varepsilon, \eta)\)-testable for every monotone 2-CNF \( \phi \) on \( N \) variables.
- \( VC(U) \) is \((\varepsilon, \eta)\)-testable for every graph \( U \) on \( N \) nodes.
- \( CLIQUE(U) \) is \((\varepsilon, \eta)\)-testable for every graph \( U \) on \( N \) nodes.

Moreover, the reductions preserve 1-sided error.

The theorem follows from the following three lemmas.

**Lemma 11.** For every undirected graph \( U \) on \( N \) nodes there is a corresponding monotone 2-CNF \( \phi_U \) on \( N \) variables s.t. if \( SAT(\phi_U) \) is \((\varepsilon, \eta)\)-testable then so is \( VC(U) \).

**Proof.** Let \( U = (V, E) \) be an undirected graph. Associate a Boolean variable \( x_v \) with each \( v \in V \). Define the 2-CNF formula \( \phi_U \) on the set of variables \( X = \{x_v \mid v \in V\} \) as follows: form the clause \( \{x_v \lor x_u\} \) for each edge \((u, v) \in E\). A subset \( S \) of vertices in \( V \) defines an assignment \( \tilde{f} \) to variables in \( X \) by \( \tilde{f}(x_v) = f_S(v) \). Clearly \( dist(f_S, VC(U)) = dist(\tilde{f}, SAT(\phi_U)) \), and every \( \varepsilon \)-test for \( SAT(\phi) \) can be used as a test for \( VC(U) \).
Lemma 12. For every undirected graph \( U \) on \( N \) nodes there is a corresponding graph \( U' \) on \( N \) nodes such that \( VC(U') \) is \( (\varepsilon, q) \)-testable then so is \( CLIQUE(U) \).

Proof. Let \( U = (V, E) \) be an undirected graph. Define \( U' = (V, E') \) where \( E' \) is the set of vertex pairs that are not edges in \( E \). For a subset \( S \subseteq V \), let \( S' = V \setminus S \). Clearly, \( dist(f_\phi, CLIQUE(U)) = dist(f_\phi, CLIQUE(U')) \), and every \( \varepsilon \)-test for \( VC(U') \) can be used as an \( \varepsilon \)-test for \( CLIQUE(U) \).

Lemma 13. For every monotone 2-CNF \( \phi \) on \( n \) variables, there is a corresponding undirected graph \( U_\phi \) on \( n \) nodes such that \( CLIQUE(U_\phi) \) is \( (\varepsilon, q) \)-testable then so is \( SAT(\phi) \).

Proof. Let \( \phi \) be a monotone 2-CNF. Associate a node \( v_x \) with each variable \( x \) of \( \phi \). Define the undirected graph \( U_\phi \) on the set of vertices \( V = \{ v_x | x \in \phi \} \) as follows: start with a complete graph on \( V \) and then for each clause \( (x \lor y) \) in \( \phi \) delete an edge \( (u_x, u_y) \) from \( U \). An assignment \( f \) to the variables of \( \phi \) defines a subset \( S \) of the vertices of \( V \) by \( S = \{ v_x | f(x) = 0 \} \). Clearly, \( dist(f, SAT(\phi)) = dist(f_\phi, CLIQUE(U_\phi)) \), and every \( \varepsilon \)-test for \( CLIQUE(U) \) can be used as a test for \( SAT(\phi) \).

4. GENERAL UPPER BOUND

We present a simple 1-sided error \( \varepsilon \)-test for monotonicity (not necessarily Boolean) on bipartite graphs \( G = (X, Y; E) \) with \( |X| = |Y| = N \) and all edges are directed from \( X \) to \( Y \).

<table>
<thead>
<tr>
<th>Test ( T_1 ) for ( G = (X, Y; E) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Query ( q = 2\sqrt{N/\varepsilon} ), vertices uniformly and independently from each of ( X ) and ( Y ).</td>
</tr>
<tr>
<td>2. Reject if a violated pair of vertices is found; otherwise, accept.</td>
</tr>
</tbody>
</table>

Theorem 14. If \( G = (X, Y; E) \) as above, then algorithm \( T_1 \) is a 1-sided error \( (\varepsilon, O(\sqrt{N/\varepsilon})) \)-test for \( mon_G \).

Proof. The test accepts all monotone functions. Suppose a function is \( \varepsilon \)-far from monotone. Then by Lemma 3, there are \( \varepsilon N/2 \) vertex-disjoint violated pairs. Call them witnesses and their vertices, witness-vertices. A randomly chosen \( X \)-vertex is a witness with probability \( \varepsilon \).

Let \( F \) be the event that no violated pair is detected, \( F_X \) be the event that \( \leq \varepsilon q/2 \) \( X \)-witnesses are queried, \( F_Y \) be the event that \( \leq \varepsilon q/2 \) \( Y \)-witnesses are queried.

\[
\Pr[F] = \Pr[F_X] + \Pr[F_Y] + \Pr[F \mid F_X \land F_Y] \\
\leq e^{-8} + e^{-8} + \left( 1 - \frac{\varepsilon q/2}{\varepsilon N/2} \right)^{\varepsilon q/2} \\
\leq 2e^{-3} + e^{-\frac{2\varepsilon q}{N/2}} < \frac{1}{3}.
\]

Thus, the test fails with probability less than 1/3.

By Theorems 6–10, monotonicity over general graphs and properties in Section 3 have 1-sided error \( (\varepsilon, O(\sqrt{N/\varepsilon})) \)-tests.

5. GENERAL LOWER BOUNDS

This section addresses lower bounds for testing monotonicity on general graphs. We restrict our attention to the Boolean case which implies matching lower bounds for all properties in Theorem 10. We first define what we call Ruzsa-Szemerédi type graphs. We then show that monotonicity over such graphs (with suitable parameters) is hard to test non-adaptively.

Let \( U = (V, E) \) be an undirected graph and let \( M \subseteq E \) be a matching in \( U \), i.e. no two edges in \( M \) have a vertex in common. Let \( V(M) \) be the set of endpoints of edges in \( M \). A matching \( M \) in \( U \) is called induced if the induced graph \( U[V(M)] \) contains only the edges of \( M \). Namely, \( (u, v) \in E(U) \) if and only if \( (u, v) \in E(M) \) for all \( u, v \in V(M) \).

A (bipartite) graph \( U = (X, Y; E) \) is called \( (s, t) \)-Ruzsa-Szemerédi if its edge set can be partitioned into at least \( s \) induced matchings \( M_1, \ldots, M_s \), each of size at least \( t \).

Theorem 15. Let \( U = (X, Y; E) \) be an \((m, \varepsilon N)\)-Ruzsa-Szemerédi graph with \(|X| = |Y| = N\). Direct all edges of \( U \) from \( X \) to \( Y \) to obtain a graph \( G \). Then any non-adaptive \( \varepsilon \)-test for \( mon_G \) requires \( \Omega(\sqrt{m}) \) queries.

Proof. We use Yao’s principle, which says that to show a lower bound on the complexity of a randomized test, it is enough to present an input distribution on which any deterministic test with that complexity is likely to fail. Namely, we define distributions \( D_P, D_N \) on positive (monotone) and negative (\( \varepsilon \)-far from monotone) inputs, respectively. Our input distribution first chooses \( D_P \) or \( D_N \) with equal probability and then draws an input according to the chosen distribution. We show that any deterministic non-adaptive test with \( q = o(\sqrt{m}) \) queries has error probability larger than 1/3 with respect to the induced probability on inputs.

We now define the distributions \( D_P \) and \( D_N \), as well as the auxiliary distribution \( D_{N,M} \). For \( D_P \) and \( D_N \), choose a random \( i \in \{1, \ldots, m\} \) uniformly. For all nodes \( x \in X \) and \( y \in Y \) outside of matching \( M_i \), let \( f(x) = 1 \) and \( f(y) = 0 \). For \( D_P \), uniformly choose \( f(x) = f(y) = 0 \) or \( f(x) = f(y) = 1 \) independently for all edges \( (x, y) \in M_i \). For \( D_N \), independently choose \( f(x) = 1 - f(y) = 0 \) or \( f(x) = 1 - f(y) = 1 \) independently for all \( (x, y) \in M_i \).

\( D_P \) is supported only on monotone labelings, but \( D_N \) is not supported only on negative inputs. However, for \( N \) large enough, with probability more than \( 8/9 \) at least \( 1/3 \) of the edges of \( M_i \) are violated when the input is chosen according to \( D_{N,M} \), making the input \( \varepsilon/6 \)-far from monotone. Denote the latter event by \( A \) and define \( D_N = D_N|_A \), namely, \( D_N \) is \( D_N \) conditioned on the event \( A \). Note that for \( D_N \) an edge is violated only if it belongs to \( M_i \), since the matchings are induced.

Given a deterministic non-adaptive test that makes a set \( V' \) of \( q \) queries, the probability that one or more of \( M_i \)'s edges have both endpoints in \( V' \) is at most \( q^2/(4m) \) for both \( D_P \), \( D_N \). This is because the matchings are disjoint, and the vertex set \( V' \) induces at most \( q^2/4 \) edges of \( G \). For \( q = o(\sqrt{m}) \), with probability more than \( 1 - o(1) \) no edge of \( M_i \) has both endpoints in \( V' \). Conditioned on any choice of \( i \) for which \( M_i \) has no such edge, the distribution of \( f_{V'} \) is identical for both \( D_N \) and \( D_P \): every vertex outside of \( M_i \) is fixed to 1 if it is in \( X \) and to 0 if it is in \( Y \), and the value of every other vertex is uniform and independent over \( \{0, 1\} \). Let \( C(\phi) \) denote the...
Let $B, R, W \subseteq [m]^T$ be the sets of the blue, red and white points, respectively. Then $P = |B| + |R| + |W|$.

First, we claim that $|W| \leq |B| + \left\{ x: \exists i, x_i = 1 \right\}$. Indeed, consider a new matching between $W$ and $R$ defined by matching $w \in W$ to $w - 1_T$. Assume that $m \equiv 0 \pmod{3}$. Then the only unmatched points in $W$ are contained in the set $\left\{ x: \exists i, x_i = m \right\}$, proving this claim. Similarly $|W| \leq |B| + \left\{ x: \exists i, x_i = m \right\}$.

Next, observe that the only blue and red points (in the corresponding copies of $[m]^T$) unmatched by $M$ are these which have a coordinate whose value is in $\{1, 2, m - 1, m\}$. It follows that $|M| > (|B| + |R|)/2 - \left\{ x: \exists i, x_i \in \{1, 2, m - 1, m\} \right\} > P/3 - \left\{ \left\{ x: \exists i, x_i \in \{1, 2, m - 1, m\} \right\} + \left\{ x: \exists i, x_i = m \right\} \right\} > P/3 - \frac{m}{2} \cdot P$. Since $p = o(m)$, the claim of the lemma follows.

Now, let $T, T_1$ be two $\frac{n}{9}$-sets in $[n]$. We claim that no edge of $M_T$ is induced by $M_{T_1}$. Indeed, let $b$ be matched to $r$ by $M_T$, in particular $b - r = 2 \cdot 1_T$. If the edge $(b, r)$ is induced by $M_{T_1}$, then $b$ is colored blue and $r$ is colored red in the coloring defined by $T_1$. By the definition of the coloring, since $\sum_{i=1}^{m} b_i > \sum_{i=1}^{m} r_i$, $b$ is located in a blue level separated by a white level from the red level of $r$. This implies that $\sum_{i \in T_2} b_i - \sum_{i \in T_2} r_i > \frac{2}{3}$. On the other hand, $\sum_{i \in T_2} (b_i - r_i) = \sum_{i \in T_2} (b_i - r - r_i) = \sum_{i \in T_2} (2 \cdot 1_T) = 2 \cdot |T \cap T_1| \leq \frac{2m}{n} < \frac{2}{3}$, reaching a contradiction.

We would like to have a large family $\mathcal{F}$ of $\frac{n}{9}$-subsets of $[n]$, such that the intersection between any two of them is of size at most $\frac{n}{9}$, or, equivalently, such that the Hamming distance between any two of them is at least $\frac{2n}{9} - \frac{2n}{9} = \frac{2n}{9}$. So we need a lower bound on the size of a constant weight binary error-correcting code $\mathcal{C}$ with the following parameters: block length $n$, weight $w = \frac{n}{9}$, distance $d = \frac{2n}{9}$. Applying the Gilbert-Varshamov bound for constant weight codes [19], we get $\frac{n}{9} \log |\mathcal{C}| \geq H(1/3) - 1/3 \cdot H(4/9) - 2/3 \cdot H(2/3) - o(1) = 0.014 - o(1)$. Choose $m = n^2$ and define the edge-set $E(U)$ of $U$ by $E(U) = \bigcup_{T \subseteq M_T} E(U)$. By the preceding discussion, $U$ is a graph on $N = \frac{n^2}{9}$ vertices, whose edge-set is a disjoint union of $\mathcal{C}(\frac{n^2}{9}) = N^3/\mathcal{C}(\frac{n^2}{9})$ induced matchings of size $N/3 - o(N)$.
[25], following Behrend [5]. Their result, with some abuse of notation, can be stated as follows: \( N/3, N/2^{O(\sqrt{\log N})} \in \mathcal{P} \). We have already seen that for \( \epsilon = \Omega(1) \) there is an absolute constant \( c \) such that \( (N^c/\log \log N, \epsilon N) \in \mathcal{P} \). This trivially implies that there is a constant \( c \) such that for any positive \( \epsilon \), \((1/\epsilon \cdot N^{c/\log \log N}, \epsilon N) \in \mathcal{P} \). A more technically involved generalization of the construction in this section, postponed to the full version of the paper, gives: there is a constant \( c \) such that for any constant positive \( \epsilon \leq 1/4 \), \((N^{c/\log(1/\epsilon) / \log \log N}, \epsilon N) \in \mathcal{P} \).

Letting \( \epsilon \) go to 0 as \( N \) grows, it can be shown for \( 1/\epsilon \in \Omega \left( N^{c/\log \log \log N} \right) \), that there is an absolute constant \( c < 1 \), s.t. \((N^n, N/O \left( \sqrt{\log N / \log \log N} \right)) \in \mathcal{P} \).

7. HYPERCUBE LOWER BOUNDS

1-sided non-adaptive lower bound. Consider the set inclusion order on the vertices of the Boolean hypercube \( \{0, 1\}^n \). For \( x \in \{0, 1\}^n \), let \( ||x|| \) be its Hamming weight.

**Theorem 19.** \( \exists \epsilon > 0 \) such that every non-adaptive 1-sided error \( \epsilon \)-test for monotonicity of Boolean functions on the \( n \)-dimensional Boolean hypercube requires \( \Omega(\sqrt{n}) \) queries.

**Proof.** Note that a 1-sided error test must accept if no violation is uncovered; otherwise, the test fails on monotone functions consistent with the query results. For \( \ell = 1, \ldots, n \) define a function \( f_i : \{0, 1\}^n \rightarrow \{0, 1\} \) by:

\[
 f_i(x_1, \ldots, x_n) = \begin{cases} 
 1 & \text{if } ||x|| > n/2 + \sqrt{n} \\
 1 - x_i & \text{otherwise} \\
 0 & \text{if } ||x|| < n/2 - \sqrt{n} 
\end{cases}
\]

It is easy to see that for all \( 1 \leq i \leq n \), \( f_i \) is \( \epsilon \)-far from monotone, for some constant \( \epsilon > 0 \) independent of \( n \). The following immediately implies our theorem.

**Lemma 20.** For every non-adaptive \( q \)-query monotonicity test, there exists an index \( i \in [n] \), such that the test succeeds (finds a violation) on \( f_i \) with probability at most \( O(q/\sqrt{n}) \).

**Proof.** It suffices to prove the claim for tests that only query vertices with Hamming weight in the range \( n/2 \pm \sqrt{n} \), as queries out of this range do not participate in any violation.

We show that every set of \( q \) queries reveals a violation at most \( O(q/\sqrt{n}) \) of the functions \( f_i \). It follows that for every test that makes \( q \) queries, \( \sum_{i=1}^{n} \Pr[\text{a violation for } f_i \text{ is found}] = O(q/\sqrt{n}) \), and so there exists an \( f_i \) for which the test finds a violation with probability at most \( O(q/\sqrt{n}) \), as claimed.

Let \( Q \) be the set of queried vertices of \( \{0, 1\}^n \) of size \( q \). The queries detect a pair of vertices violated by \( f_i \) only if \( Q \) contains comparable vertices \( u \) and \( v \) that differ in coordinate \( i \). Construct an undirected graph with vertex set \( Q \), by drawing an edge between \( x \) and \( y \) if they are comparable. Consider a spanning forest of this graph. If such vertices \( u \) and \( v \) exist, they must lie in the same tree. Furthermore, there must exist adjacent vertices on the path between \( u \) and \( v \) that differ in coordinate \( i \). Therefore, the number of functions \( f_i \) for which the queries reveal a violation is at most the maximum number of edges in the forest (which is at most \( q - 1 \) multiplied by the maximum possible distance between adjacent vertices \( (2\sqrt{n}) \)). The total is at most \( O(q/\sqrt{n}) \).

2-sided lower bound. We give a logarithmic lower bound for non-adaptive 2-sided monotonicity tests of Boolean functions over \( \{0, 1\}^n \). This implies a non-constant (though doubly logarithmic) lower bound for adaptive testing of this property.

**Theorem 21.** \( \exists \epsilon > 0 \) such that every non-adaptive \( \epsilon \)-test for monotonicity of Boolean functions on the \( n \)-dimensional Boolean hypercube requires \( \Omega(\log n) \) queries.

**Proof.** The lower bound uses Yao’s method. Namely, we define two distributions over input functions, \( D_{P} \) and \( D_{N} \), such that for any set of \( q \leq \frac{1}{20} \log n \) vertices of the hypercube, the distributions induced on \( \{0, 1\}^n \) by restricting the functions to the \( q \) vertices are \( \epsilon \)-close, while an input chosen according to \( D_{P} \) is monotone, and an input chosen according to \( D_{N} \) is \( \epsilon \)-far from monotone for a constant \( \epsilon \).

**Two distributions.** For \( x \in \{0, 1\}^n \), we view \( x \) as both a binary vector of length \( n \) and a subset \( \{i : x_i = 1\} \) of \( [n] \).

**Definition 2.** Let \( \alpha = \frac{\epsilon}{10} \). Given \( B \subseteq [n] \), let \( \text{maj}(x \cap B) \) be 1 when \( |x \cap B| > \frac{1}{2}|B| \) and 0 otherwise.

The trimmed oligarchy function according to \( B \) is:

\[
 f_B(x) = \begin{cases} 
 1 & \text{if } ||x|| > n/2 + \alpha \sqrt{n} \\
 \text{maj}(x \cap B) & \text{if } ||x|| < n/2 - \alpha \sqrt{n} \\
 0 & \text{otherwise}
\end{cases}
\]

The trimmed anti-oligarchy function according to \( B \) is:

\[
 f_B^\oplus(x) = \begin{cases} 
 1 & \text{if } ||x|| > n/2 + \alpha \sqrt{n} \\
 0 & \text{if } ||x|| < n/2 - \alpha \sqrt{n} \\
 1 - \text{maj}(x \cap B) & \text{otherwise}
\end{cases}
\]

The theorem follows from the next two lemmas.

**Lemma 22.** There exists \( \epsilon > 0 \), such that for any nonempty set \( B \), \( f_B \) is monotone and \( f_B^\oplus \) is \( \epsilon \)-far from monotone.

**Proof.** It is easy to see that trimmed oligarchy functions are monotone. For trimmed anti-oligarchy functions, we will find \( \epsilon^2 n \) vertex-disjoint violated pairs.

Let \( m = |B| \). For every integer \( w \) such that \( \frac{1}{2} n - \alpha \sqrt{n} \leq w < \frac{1}{2} n \), and every integer \( v \) such that \( 0 \leq v < \frac{1}{2} m \), let \( U_{w,v} \) denote the set \( \{x \in \{0, 1\}^n : ||x|| = w \text{ and } |x \cap B| = v \} \), and \( V_{w,v} \) denote the set \( \{x \in \{0, 1\}^n : ||x|| = n-w \text{ and } |x \cap B| = m-v \} \). By definition, \( f(x) = 1 \) for every \( x \in U_{w,v} \) and \( f(x) = 0 \) for every \( x \in V_{w,v} \). \( U_{w,v} \) and \( V_{w,v} \) have the same size, since \( x \) is in \( U_{w,v} \) if the complement of \( x \) is in \( V_{w,v} \). We want to find a bijection \( \sigma : U_{w,v} \rightarrow V_{w,v} \) such that \( x \preceq \sigma(x) \) for every \( x \in U_{w,v} \).

Consider the bipartite graph over \( U_{w,v} \cup V_{w,v} \) with the poset relations as edges. It is easy to see that this graph has a constant degree, so a matching exists (by Hall’s Theorem) if this degree is nonzero. This happens if \( w, v \) satisfies \( \frac{1}{2} m - v \leq \frac{1}{2} n - w \) in addition to the conditions above. The union over all such \( w, v \) of the sets \( U_{w,v} \cup V_{w,v} \) covers a fixed fraction of the hypercube, so we are done.
To define $D_P$ and $D_N$ pick a random set $B \subseteq [n]$ by independently choosing each coordinate to lie in $B$ with probability $\frac{1}{2n-1}$. For $D_P$, take the corresponding $f_B$ and for $D_N$, take the corresponding $f_B^\perp$.

**Lemma 23.** $D_N$ and $D_P$ restricted to any set of $q = \frac{1}{5}\log n$ queries are $\epsilon$-close, for any $\epsilon > 0$.

**Proof sketch.** Let $q = \frac{1}{5}\log n$. Let $x_1, \ldots, x_q$ be a fixed subset of $\{0, 1\}^n$. We can assume, without loss of generality, that the points satisfy $n/2 - \alpha\sqrt{n} \leq \|x_i\| \leq n/2 + \alpha\sqrt{n}$. This is because the functions in $D_P$ and $D_N$ are constant and identical outside this range.

Inside the range, for any $B$, the corresponding oligarchy and anti-oligarchy functions complement each other. Therefore, the induced distributions $d_P$ and $d_N$ on $\{0, 1\}^n$ are mirror images of each other: $d_P(a) = d_N(\bar{a})$ for any $a \in \{0, 1\}^n$, where $\bar{a}$ is the complement of $a$. For a distribution $d$ on $\{0, 1\}^n$, let $\bar{d}$ be its mirror image. Call $d$ symmetric if $d = \bar{d}$. Our claim amounts to showing $d_P$ to be almost symmetric.

In fact, we construct a symmetric distribution $s$, such that $\|d_P - s\|_1 = o(1)$. This implies our claim since

\[
\|d_P - d_N\|_1 = \|d_P - d_P^\perp\|_1 \\ \leq \|d_P - s\|_1 + \|d_P^\perp - \bar{s}\|_1 = 2\|d_P - s\|_1.
\]

We get $s$ by a sequence of four distributions, each one close to its predecessor. The first element in the sequence is $d_1 = d_P$ and the last is $s$. The triangle inequality then implies that the distance between $d_P$ and $s$ is at most the sum of the distances between the consecutive elements of the sequence.

For $1 \leq i \leq q$, let $y_i \in \{0, 1\}^n$ be such that $\|y_i\| = n/2$ and $\|x_i - y_i\| \leq \alpha\sqrt{n}$. Let $d_2$ be the distribution on $\{0, 1\}^n$ induced by restricting the functions in $D_P$ to $y_1, \ldots, y_q$. Then $d_2$ is close to $d_1$ because w.h.p. over the choice of a function $f$ from $D_P$, changing the queries by at most $O(\sqrt{n})$ bits, does not change the value of the $f$ on the queries.

The $n/2$-sets $y_1, \ldots, y_q$ induce a standard partition of $[n]$ into $2^q$ disjoint subsets, indexed by $\{0, 1\}^q$. For $I \in \{0, 1\}^q$, the $I$th element of the partition is $A_I = \bigcap_{i \in I} y_i \cap \bigcap_{i \not\in I} \bar{y}_i$. Here $y_i$ is the set complement of $y_i$. We define $2^q$ random variables depending on $B$ by setting $R_I(B) = |B \cap A_I|$. If $A_I$ is empty, $R_I$ is identically 0.

Note that the $\{R_I\}$ are independent binomially distributed variables, and that they determine the restriction of $f$ to $y_1, \ldots, y_q$. In fact, $f(y) = 1$ if and only if $|y \cap B| > |\bar{y} \cap B|$, which is equivalent to $\sum_{I:1 \in I} R_I > \sum_{I:1 \not\in I} R_I$.

Since $E(\sum_{I:1 \in I} R_I) = np/2 = E(\sum_{I:1 \not\in I} R_I)$, we can replace each $R_I$ by a random variable $Z_I = R_I - ER_I$ with zero expectation.

Next, we would like to replace each $Z_I$ by a symmetric random variable $S_I$. Observe that once we do so, provided the new distribution $d_3$ on $\{0, 1\}^q$ is close to $d_2$, we are basically done. Indeed, a choice of a point in $\{0, 1\}^q$ according to $d_3$ is determined by the signs of $q$ linear expressions in $S_I$. Since $d_3$ is invariant under flipping the sign of all the $S_I$, a point $a$ and its complement $\bar{a}$ would be chosen with almost \(^2\) the same probability. Therefore, $d_3$ is close to a symmetric distribution $s$ on $\{0, 1\}^q$, completing the proof.

We know, say by the Berry-Esseen theorem ([9], p. 126) that a binomial distribution with parameters $k$ and $p$, such that $kp \gg 1$, is, in some sense, close to the normal distribution which is, of course, symmetric. We give a precise meaning to this intuition, proving directly that such a binomial distribution is stochastically close to a symmetric one. This allows us to replace $Z_I$ with large parameter $k = |A_I|$ by symmetric random variables.

As to $Z_I$ for which the parameter $k$ is small, it turns out that we can get rid of them simply by replacing them with 0.

**8. FAMILIES OF GRAPHS WITH EFFICIENT MONOTONICITY TESTS**

This section describes several families of efficiently testable graphs, including graphs with few edges in the transitive closure, graphs with small width, top–parallel graphs, trees and graphs with small separators. All tests presented have 1-sided error. Hence, we only need to analyze the probability of error for functions that are far from monotone. Throughout the section, we denote the transitive closure of a graph $G$ by $TC(G)$.

**8.1 Tests with query complexity independent of graph size**

Our first test $T_2(q)$ works when the fraction of vertex pairs violated by the input function is high and is useful for testing graphs with few edges in the transitive closure, as well as small width graphs. Note that test $T_2(q)$ queries at most $2q$ vertices.

**Test $T_2(q)$**

1. Pick $q$ edges from the transitive closure of the graph uniformly and independently.
2. For each edge, query its endpoints. Reject if it is violated; otherwise, accept.

**Lemma 24.** If $G$ is a graph with at most $cN^2$ edges in $TC(G)$, then algorithm $T_2$ with parameter $q$ set to $4c/\varepsilon$ is a $1$-sided error ($\varepsilon$, $8c/\varepsilon$)-test for monotonicity on $G$.

A graph $G$ has width $w$ if every set of mutually incomparable vertices has size at most $w$. The following shows that $T_2$ can be used to test small width graphs.

**Lemma 25.** If $G$ is a graph of width $w$, then algorithm $T_2$ with $q$ set to $2w/\varepsilon^2$ is a $1$-sided error ($\varepsilon$, $4w/\varepsilon^2$)-test for monotonicity of Boolean functions on $G$.

**Proof of Lemma 25.** Let $G$ be a graph of width $w$ and let $f$ be a Boolean labeling of $V(G)$ that is $\varepsilon$-far from monotone. We will show that the number of violated edges in the transitive closure is at least $\varepsilon^2 N^2/(2w) - O(1)$. Since the total number of edges in the graph is at most $N^2/2$, the test will find a violated edge with probability at least $1 - q^{-2} > 2/3$.

A real random variable $X$ is symmetric if for all $t$, $P(X \leq \mathbb{E}X - t) = P(X > \mathbb{E}X + t)$. The two notions of symmetric distributions in the proof should, hopefully, cause no confusion.

\(^2\)Because in order to get 1 in j-th coordinate we need the corresponding linear expression to be strictly positive, and to get 0 we need it to be only non-positive (zero included).
CLAIM 26. If \( \text{dist}(f, G) \geq d \) for a Boolean labeling \( f \), then there is a set \( T, |T| \leq w \), of 0-labeled vertices, s.t. \( T \) is incident to at least \( d \) violated pairs.

PROOF OF CLAIM. If \( \text{dist}(f, \text{mon}_G) \geq d \), by Lemma 4, \( \text{TC}(G) \) has a matching of violated edges of size \( d \). Call endpoints of the edges in the matching witnesses. Let \( Z \) be the set of 0-labeled witnesses and let \( T \subset Z \) be a minimal set of vertices s.t. \( \forall z \in Z \exists t \in T \) with \( z \leq_G t \). Clearly, \( T \) contains no comparable pairs, and hence is of size at most \( w \). Each 1-labeled witness is below one of the nodes in \( Z \) and hence in \( T \).

Applying the claim to \( \text{TC}(G) \) and removing the nodes in \( T \) from the graph repeatedly until no vertices are left, we observe that the number of violated edges in \( \text{TC}(G) \) is at least \( \varepsilon N + (\varepsilon N - w) + (\varepsilon N - 2w) + \cdots + (\varepsilon N \mod w) \sim (\varepsilon^2N^2)/(2w) \).

Top-parallel graphs. Let \( G_1 = (V_1, E_1) \) and \( G_2 = (V_2, E_2) \) be disjoint graphs. Graph \( G \) obtained by connecting \( G_1 \) and \( G_2 \) in parallel is defined by \( G = (V_1 \cup V_2, E_1 \cup E_2) \). Graph \( G \) obtained by connecting \( G_1 \) and \( G_2 \) using the top operation is defined by \( G = (V_1 \cup V_2, E_1 \cup E_2 \cup E_x) \), where \( E_x = \{(v_1, v_2) \mid v_1 \in V_1 \text{ and } v_2 \in V_2 \} \). Top-parallel graphs are defined recursively: the 1-vertex graph is top-parallel, and a graph formed by top or parallel operations from two top-parallel graphs is also top-parallel. Examples of top-parallel graphs include the transitive closure of a rooted tree with all edges directed either towards the root or away from the root, and the transitive closure of a complete layered graph. The proof of the following lemma is omitted.

LEMMA 27. If \( G \) is a top-parallel graph, it has a 1-sided \((\varepsilon, A/\varepsilon^2)\)-test for Boolean monotonicity.

Tree-like graphs. A directed graph \( G(V, E) \) is tree-like if it is obtained by arbitrarily directing each edge of a forest \( (V, E) \).

LEMMA 28. If \( G \) is a tree-like graph, it has a 1-sided error \((\varepsilon, 16/\varepsilon^2)\)-test for Boolean monotonicity.

The full proof is technical, and we omit it. We sketch a simplified proof for the case of a rooted tree \( G \) which is obtained from a tree \( T = (V, E) \) by choosing a special vertex \( r \in V \) and directing the edges along paths from other vertices to \( r \). Let \( \text{Low}(v) \) denote \( \{u \in V \mid u \leq_G v\} \) for \( v \in V \). Observe that in a rooted tree, \( \text{Low}(x) \cap \text{Low}(y) = \emptyset \) for all incomparable vertices \( x, y \). A vertex \( v \in V \) is \( \varepsilon \)-bad with respect to a Boolean labeling \( f \) if \( f(v) = 0 \) and more than \( \varepsilon \) fraction of vertices in \( \text{Low}(v) \) are labeled 1 by \( f \).

### Table: \( T_\varepsilon \) for Boolean Labelings on Rooted Trees

<table>
<thead>
<tr>
<th>( T_\varepsilon ) for Boolean Labelings on Rooted Trees</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Query ( \varepsilon ) nodes uniformly and independently.</td>
</tr>
<tr>
<td>2. For each queried node with label 0, query ( \varepsilon ) nodes below it uniformly and independently. Reject if a violated pair is found; otherwise, accept.</td>
</tr>
</tbody>
</table>

If at least an \( \varepsilon/2 \) fraction of vertices in \( G \) are \( \varepsilon/2 \)-bad with respect to the input labeling \( f \), the test rejects with probability \( \geq 2/3 \). Otherwise, we can obtain a monotone \( f' \) by changing \( f \) on less than \( \varepsilon \) fraction of the vertices. Let \( Z \) be the set of 0-labeled vertices which are not \( \varepsilon/2 \)-bad, and \( S \) be the set of maximal nodes of \( Z \). Set \( f'(x) = 0 \) if \( x \in \text{Low}(v) \) for some \( v \in S \) and to 1 otherwise. It is not hard to show that \( f' \) is monotone and close to \( f \), as claimed, which implies the correctness of the test.

A generalization of a \( \text{bad} \) vertex yields a similar test for tree-like graphs.

### 8.2 A test for graphs with small separators

Here we consider graphs that can be broken into relatively small connected components by removing a few vertices.

**Definition 3.** Let \( \mathcal{U} \) be an infinite family of undirected graphs that is closed under taking subgraphs. We say that \( \mathcal{U} \) is \( k \)-separable if every \( N \)-vertex graph \( U \in \mathcal{U} \) can be broken into connected components of size at most \( 2N/3 \) by removing a subset of at most \( k \) vertices, called a separator.

E.g., forests are 1-separable, bounded tree-width graphs have bounded separators and planar graphs are \( O(\sqrt{N}) \)-separable [20]. In the sequel \( k \) might be a sublinear non-decreasing function of \( N \).

Let \( G = (V, E) \) be a directed graph. Let \( U_G \) be the undirected graph obtained from \( G \) by undirecting its edges. Call \( G \) \( k \)-separable if \( U_G \) belongs to a \( k \)-separable family of graphs.

Consider a ‘standard’ tree structure over disjoint subgraphs of \( G \) generated by inductively taking out separators. Namely, generate a rooted tree \( T \) where each node \( x \in T \) is associated with a set of vertices \( V(x) \) of \( G \). Let \( V_0 \) be a separator for \( U_G \) of size \( \leq k \), and suppose that \( U_G(V - V_0) \) has \( l \) components.

The root \( x \) of \( T \) is associated with \( V_0 \) (i.e., \( V(x) = V_0 \)) and has \( l \) children, one for each component. The subtrees of the children are generated recursively from their respective components by the same procedure. The recursion stops at components of size less than \( k \log N \). The leaves are associated with vertex sets of their components. Note that the depth of the tree is \( O(\log N) \).

Let \( r = x_0, x_1, \ldots, x_j = x \) be the path from the root to a node \( x \) in \( T \). Denote \( \bigcup_{i=0}^{j} V(x_i) \) by \( \text{Path}(x) \). Namely, \( \text{Path}(x) \) contains all vertices of \( G \) associated with \( x \) and all vertices from separators that appear on the path from the root of \( T \) to \( x \). For a vertex \( v \in V \) let \( T(v) \) denote the node \( x \in T \) so that \( v \in V(x) \).

We present a 1-sided error test for \( G \) using the structure \( T \).

**Test for Graphs with Small Separators, \( T_\varepsilon \)**

1. Pick \( \varepsilon \) nodes of \( G \) uniformly and independently.
2. For each node \( v \), query all nodes in \( \text{Path}(T(v)) \). Reject if a violated pair is found; otherwise, accept.

Call a vertex \( v \) bad if \( \text{Path}(T(v)) \) contains a violated pair.

**Claim 29.** If a function is \( \varepsilon \)-far from monotone, at least \( \varepsilon/2 \) fraction of vertices are bad.

**Proof.** Consider a violated pair \((v, u)\). We will prove that either \( v \) or \( u \) is bad. The claim then follows as the graph has at least \( \varepsilon N/2 \) vertex-disjoint violated pairs (by Lemma 3).
If $T(v)$ and $T(u)$ are on the same path from the root to a leaf in $T$, then $v \in Path(T(u))$ or $u \in Path(T(v))$. W.l.o.g., suppose $v \in Path(T(u))$, then $u$ is bad because $Path(T(u))$ contains a violated pair $(v, u)$. If $T(v)$ and $T(u)$ are not on the same path from the root to a leaf, they got separated when $T$ was constructed, i.e., some vertex $w_2$ on a directed path from $v$ to $u$, in $G$, is in a common ancestor of $T(v)$ and $T(u)$. Since $(v, u)$ has to be violated, either $v$ or $u$ is bad.

Lemma 30. Let $G = (V, E)$ be a $k$-separable $N$-vertex graph. Then algorithm $T_k$ is a 1-sided error $(\varepsilon, O(\frac{1}{\varepsilon} \log N))$-test for monotonicity of functions (with general ranges) on $G$.

This generalizes the more efficient tests for Boolean functions over tree-like graphs and bounded-width graphs for which tighter results (by $\log N$ factor) are obtained in lemmas 28 and 25. It also provides an alternative $(\varepsilon, O(\sqrt{N} \log N))$-test for planar graphs, which performs more queries than the general algorithm from section 4, but requires fewer label comparisons. We note that this result cannot be dramatically improved as the general monotonicity test for the line (which is 1-separable) requires $\Omega(\log N)$ queries [11].

Acknowledgment:
We would like to thank Michael Sipser for helpful discussions.

9. REFERENCES