Symmetric Approximation Arguments for Monotone Lower Bounds without Sunflowers

Christer Berg[*] Staffan Ulfberg[*]

23. December 1997

Abstract

We propose a symmetric version of Razborov's method of approximation to prove lower bounds for monotone circuit complexity. Traditionally, only DNF formulas have been used as approximators, whereas we use both CNF and DNF formulas. As a consequence we no longer need the Sunflower lemma that has been essential for the method of approximation. The new approximation argument corresponds to Haken's recent method for proving lower bounds for monotone circuit complexity (counting bottlenecks) in a natural way.

We provide lower bounds for the BMS problem introduced by Haken, Andreev's polynomial problem, and for Clique. The exponential bounds obtained are the same as the previously best known for the respective problems.

1 Introduction

The difficulty of proving super-linear lower bounds for general circuit complexity has led to the study of various restrictions of this computational model. One of these is monotone circuits, i.e., only AND gates and OR gates are allowed. Since many important problems in complexity theory are monotone, this computational model seems to be a natural one. Examples of monotone graph problems are Clique, Vertex cover, and Hamiltonian path, which are all NP-complete. However, for long there were no stronger lower bounds for monotone than for general circuits; the best one was only $4n$ (by Tiekenheinrich [14]). A major breakthrough came in 1985, when Razborov [11] invented the method of approximation. It allowed him to prove a super-polynomial lower bound as he showed that Clique requires monotone circuits of size $n^{\Omega(\log n)}$. Shortly thereafter, Andreev [3] applied Razborov's technique to another function and was thereby able to prove an exponential lower bound. Later, both these results were improved by Alon and Boppana [1], and in particular, they were the first to prove an exponential lower bound for Clique. For a nice exposition of this result, see Section 4 in the survey by Boppana and Sipser [5].

The idea behind the approximation method is the following. The output of each gate in a given circuit is approximated by a function, the approximator for

[*]Department of Numerical Analysis and Computing Science, Royal Institute of Technology, S-100 44 Stockholm, Sweden. *E-mail:* {berg,staffanu}@nada.kth.se.

that gate. The goal is to find a set of approximator functions with the following properties. For every gate in the circuit, the approximator should introduce only few errors. Also, the approximator for the output gate should differ from the function computed by the circuit for many input values. If this goal is accomplished we have proved that the circuit must be large since errors in the approximator for the output gate must emerge from the approximation of some gate in the circuit.

In 1995, Haken [7] proposed a new method, which he named "Bottleneck counting," for proving lower bounds for the size of monotone circuits. He demonstrated the method for a graph problem that resembles Clique. Instead of trying to approximate the behavior of the circuit, he defined a function μ which maps input graphs to gates of the circuit. To prove that the circuit contains many gates, he then showed that on one hand the total number of graphs mapped by μ is large, but on the other hand that only few graphs are being mapped to each gate.

One purpose of this paper is to show that Haken's approach to lower bounds is in fact an approximation argument in disguise: it can be turned into an approximation argument by using approximators that for a gate e introduce errors for exactly those inputs that are mapped to e by Haken's function μ. The approximation version of Haken's proof is in Section 5.

It is of course nice that Haken's method of bottleneck counting works in essentially the same way as Razborov's approximation method, but unfortunately this leads to the conclusion that it cannot be used to prove lower bounds better than $\Omega(n^2)$ for general circuits. The reason for this is, as Razborov showed, that the method of approximation will not work very well for general circuits [12].

Pudlák [10] extended Razborov's approximation method to hold for circuits consisting bounded fan-in gates computing monotone real functions. The corresponding extension of Haken's method was made by Haken and Cook [8]. Based on Haken's idea, Jukna [9] was able to derive a general criterion for lower bounds that holds for both monotone boolean and monotone real circuits. Wigderson independently discovered that Haken's method can be turned into an approximation argument, and he also noted that this holds for the extension made by Haken and Cook. The extension to real monotone circuits can be found in [4].

Another purpose of the paper is to make the approximation method simpler and more symmetric. Our approximation argument, which is derived from the translation of Haken's proof, differs from that of Razborov in that we use *two* approximator functions for each gate. Razborov used only one approximatior function which is a monotone depth two circuit. Out of the two possibilities for the form of such circuits, a disjunction of conjunctions or the other way around, Razborov chose the former.

In this paper, however, we do not make this choice at all since we use two approximator functions for each gate—one for each of the two possible forms. This seems like a natural thing to do, and one consequence is that we do not have to use the Sunflower lemma by Erdős and Rado [6]. The Sunflower lemma is traditionally used to reduce the size of the approximator functions while not reducing their approximation qualities too much. The fact that we do not need this lemma is somewhat surprising since it is a central part in earlier approximation results. Although we are unable to improve the bounds obtained by Alon and Boppana, this approach leads to shorter and, to our minds, simpler proofs.

Note that while most of the known lower bounds obtained using the method of approximation may benefit from being reformulated in this way, it is not clear that all of them do. We do not, for example, know how to present Razborov's super-polynomial lower bound for perfect matching (which implies that monotone circuits are strictly less powerful than non-monotone) using CNF and DNF approximators.

1.1 Recent work

Ideas from the bottleneck counting argument were also used by Amano and Maruoka [2] to develop approximators similar to the ones presented in this paper. They used the approximators to prove lower bounds for Clique and Andreev's polynomial problem.

Simon and Tsai [13] showed that the bottleneck counting argument is actually equivalent to the method of approximation. They provided a proof of the fact that errors in the approximation method correspond to mapped inputs in bottleneck counting.

2 Definitions

A *monotone circuit* Ψ is a boolean circuit with AND gates ($\wedge$) and OR gates ($\vee$), but without NOT gates. We let all gates have arbitrary fan-in and fan-out, and measure the size of a circuit, size(Ψ), by the number of gates it contains.

The monotone functions for which we provide lower bounds are all graph problems. An input graph on n vertices is represented in a standard way: as $\binom{n}{2}$ variables $x_{i,j}$ whose value is 1 if the edge (i,j) exists.

The output of a gate e in Ψ when the input x is applied to the circuit is denoted by $e(x)$. Let the output gate be e_o so that the circuit Ψ computes the boolean function $e_o(x)$.

Our proofs are based on the method of approximation which was invented by Razborov [11]. It involves the use of a boolean function to approximate the output of every gate in a given circuit. The approximator functions used by Razborov, Andreev [3], and Alon and Boppana [1] are all monotone DNF formulas.

The proofs in this paper, however, are influenced by the work of Haken [7], and every gate e in a circuit Ψ is approximated by two functions f_e^{D} and f_e^{C}, the *approximators* for the gate e. The approximator f_e^{D} has the form $C_1 \vee C_2 \vee \cdots \vee C_t$, where C_i is a conjunction of limited size: all conjunctions have less than c distinct literals. The approximator f_e^{C} has the form $D_1 \wedge D_2 \wedge \cdots \wedge D_s$, where D_i is a disjunction containing less than d distinct literals. (Notice that we put no explicit restriction on the numbers s and t.)

The following characteristics of the approximator functions f_e^{D} and f_e^{C} are essential to the proof.

1. The approximators $f_{e_o}^{\mathrm{D}}(x)$ and $f_{e_o}^{\mathrm{C}}(x)$ fail to correctly represent the output of Ψ for many inputs x.

2. For every gate e, the number of inputs for which the approximators introduce errors (measured in a specific way) is small.

For every x for which the approximators for e_o fail, the error must have been introduced in at least one of the gates in Ψ. We can therefore draw the conclusion that Ψ contains many gates.

For convenience we consider the inputs to the circuit as being gates themselves, and for an input variable $x_{i,j}$ we simply define $f^{\mathrm{D}}_{x_{i,j}} = f^{\mathrm{C}}_{x_{i,j}} = x_{i,j}$. We also define empty disjunctions to have the value 0, and empty conjunctions to have the value 1.

Definition 1. *For a gate e and an input x for which $e(x) = e_o(x)$, we say that the approximator f^{D}_e* fails *for x if $f^{\mathrm{D}}_e(x) \neq e(x)$, and that the approximator f^{C}_e* fails *for x if $f^{\mathrm{C}}_e(x) \neq e(x)$.*

If either f^{D}_e or f^{C}_e fails for x, we say that the approximators for the gate e fail for the input x.

A central part of the proofs is counting the number of errors that are introduced in the gates of Ψ. This is defined as follows.

Definition 2. *An approximator, f^{D}_e or f^{C}_e, is said to* introduce an error *for the input x if it fails on the input x, but none of the approximators for the input gates to e fail for the input x.*

If either f^{D}_e or f^{C}_e introduces an error for the input x, we say that the approximators for the gate e introduce an error for the input x.

We now describe how an $\wedge$ gate e is approximated assuming that its input gates are already approximated by $f^{\mathrm{D}}_{e_1}, f^{\mathrm{D}}_{e_2}, \ldots, f^{\mathrm{D}}_{e_m}$ and $f^{\mathrm{C}}_{e_1}, f^{\mathrm{C}}_{e_2}, \ldots, f^{\mathrm{C}}_{e_m}$.

The approximator f^{C}_e is simply defined as

$$f^{\mathrm{C}}_e = \bigwedge_{i=1}^{m} f^{\mathrm{C}}_{e_i} = \bigwedge_{i=1}^{s} D_i,$$

in which all disjunctions still have length at most d. If none of $f^{\mathrm{C}}_{e_1}, f^{\mathrm{C}}_{e_2}, \ldots, f^{\mathrm{C}}_{e_m}$ fails for the input x, neither will f^{C}_e, so the approximator f^{C}_e introduces no errors.

The approximator f^{D}_e should be a disjunction of conjunctions; it is formed by converting f^{C}_e into the form

$$\bigvee_{i=1}^{t} C_i.$$

The standard way for doing this is to form a conjunction C_i for every possible way to pick one literal from each disjunction D_i in f^{C}_e. All conjunctions that have at least c distinct literals are then discarded, which means that long conjunctions are approximated by the constant 0.

In the proofs to follow we need to establish upper bounds for the number of errors introduced when forming f^{D}_e from f^{C}_e. To get as good bounds as possible, we have to be more careful when forming f^{D}_e (so that more conjunctions get shorter than c). We will shortly describe the process we actually use in detail.

Returning to the definitions of f^{D}_e and f^{C}_e, we note that f^{D}_e is produced from f^{C}_e, which in turn is constructed from the approximators $f^{\mathrm{C}}_{e_i}$ of the input gates. Thus, when approximating an $\wedge$ gate, we do not use the approximators $f^{\mathrm{D}}_{e_i}$ for the input gates.

The approximator functions for an $\vee$ gate e are formed analogously. We construct f^{D}_e from the approximators $f^{\mathrm{D}}_{e_i}$ of the input gates; this introduces no errors. The disjunction of conjunctions is then converted into a conjunction of disjunctions, and disjunctions with at least d distinct literals are discarded (they are in effect approximated by the constant 1). Notice that the way the approximators are constructed we have $f_e^{\mathrm{D}} \leq f_e^{\mathrm{C}}$ for all gates e.

We now end this section by describing the details involved when converting f_e^{C} to f_e^{D} in an $\wedge$ gate (this part is analogous for $\vee$ gates as well).

A set of conjunctions is formed that have the property that at least one of them is satisfied if and only if all disjunctions in f_e^{C} are satisfied. The approximator f_e^{D} is then formed from all resulting conjunctions shorter than c.

The rewriting process can be viewed as the construction of a tree: Each edge in the tree is labeled by a literal. For every node v in the tree we define a corresponding conjunction that is formed by all literals on the path from the root to v.

At the root we create one labeled edge to a new child for each of the literals in the first disjunction. We say that we have expanded D_1 under the root.

Suppose w is a leaf that was created while expanding D_i, and that C is the conjunction corresponding to w. Then, $D_1, \ldots, D_i$ are all satisfied if C is satisfied. We now take care of D_{i+1}.

First consider the case that when C is satisfied, we know that for all accepting instances of the problem at least one of the literals in D_{i+1}, say $x_{u,v}$, must be satisfied as well. (This happens if D_{i+1} contains a literal in common with C, but it could also happen in some other situations if there are restrictions on the possible inputs.) We then make only one new child under w for the literal $x_{u,v}$ and skip the rest of D_{i+1}. This results in fewer leaves in the tree and therefore fewer conjunctions.

Otherwise, we expand the disjunction D_{i+1} under w.

The result is that if the conjunction corresponding to one of the new children is satisfied, so is D_{i+1}. Conversely, if all of $D_1, \ldots, D_{i+1}$ are satisfied, there is a node on depth $i+1$ in the tree whose corresponding conjunction is satisfied.

When there are no more leaves for which there are remaining disjunctions to expand, we are done and we get one conjunction for each leaf in the tree. Leaves whose corresponding conjunctions have at least c distinct literals are then removed, and this is the reason why f_e^{D} may make an error for some inputs.

Note that such inputs are always accepted by some conjunction (corresponding to a node in the tree) that has exactly c distinct literals, and bounding the number of such conjunctions is the reason for constructing the tree. By counting the number of inputs accepted by each conjunction, we can therefore get an upper bound for the number of errors introduced by f_e^{D}.

3 The proof method

Although they differ in details, the basic strategy for the proofs is the same in the following sections. In this section we therefore describe the strategy by giving generic versions of the theorem, and of the lemmas that are used in the proof of the theorem.

The generic theorem and lemmas contain various unspecified entities, such as numerical functions α, β, and γ. The outline can be turned into an actual proof of an actual theorem by specifying these entities (and checking their required properties).

Let $\mathrm{MGP}(n)$ be a monotone language on graphs with n input variables (which indicate the presence or absence of edges). For some function $h(n)$ we want to prove the following.

Generic Theorem. The monotone circuit complexity for $\mathrm{MGP}(n)$ is at least $h(n)$.

Generic proof. Suppose we are given a circuit Ψ that correctly decides $\mathrm{MGP}(n)$. In order to prove that Ψ must consist of at least $h(n)$ gates, we study two subsets of the possible input graphs: the *positive test graphs,* which is a subset of the graphs in $\mathrm{MGP}(n)$, and the *negative test graphs*, which is a subset of the graphs not in $\mathrm{MGP}(n)$. Let $\gamma_1(n)$ be the number of positive test graphs and $\gamma_0(n)$ the number of negative test graphs.

Instead of directly proving that a circuit that decides $\mathrm{MGP}(n)$ must be large, we prove that a circuit that separates the positive and negative test graphs from each other must be large; choosing the test graphs in a suitable way is of course essential.

We start by showing that the approximators for the output gate of Ψ fail for most inputs.

Generic Lemma A. At the output gate e_o, the approximators either fail for all negative test graphs or they fail for at least half of the positive test graphs.

Generic proof. Assume that the approximators do not fail for all negative test graphs; otherwise there is nothing to prove. Hence, there exists a negative test graph b such that $f^{\mathrm{C}}_{e_o}(b) = 0$ which implies that $f^{\mathrm{C}}_{e_o}$ contains a disjunction D. Furthermore, for all positive test graphs g on which $f^{\mathrm{C}}_{e_o}$ is correct, we must have $D(g) = 1$. Since D contains less than d distinct literals, we can bound the fraction of positive test graphs for which $f^{\mathrm{C}}_{e_o} = 1$ by multiplying the fraction of positive test graphs for which any specific edge is present with d. A suitable choice of d proves the lemma. $\square$

The next objective is to bound the number of inputs for which the approximators introduce errors at a single gate. Assume e is an $\wedge$ gate. In this case f^{C}_e introduces no errors at all, and hence, we need only study f^{D}_e. Also, since $f^{\mathrm{D}}_e \leq f^{\mathrm{C}}_e$, no errors are ever introduced for negative test graphs in $\wedge$ gates. The next two lemmas introduce the functions α and β as bounds for the number of errors introduced in a single $\wedge$ and $\vee$ gate respectively.

Generic Lemma B. At an $\wedge$ gate e, the approximator f^{D}_e introduces an error for at most $\alpha(n, c, d)$ positive test graphs.

Generic proof. To prove the lemma, we give an upper bound on the number of positive test graphs g such that $C(g) = 1$ (g is *accepted* by C), where C is any conjunction with c distinct literals corresponding to a node in the tree that represents the rewriting process.

For each node in the tree that has more than one child, the number of children is bounded by d, and descending to such a child always adds a distinct literal to the corresponding conjunction. There are therefore at most d^c conjunctions with exactly c distinct literals. It remains to find out the number of positive test graphs accepted by each such conjunction.

This is easily done if the edges in the positive test graphs are present independently from each other (as is the case in the lower bound for Andreev's polynomial problem in Section 4). If the problems are specified in terms of vertices (e.g., the BMS problem in Section 5 and Clique in Section 6), we have to be a bit more careful since edges are not independent in the natural test graphs for these problems. Actually, in the proofs for these specific problems, we do not limit the number of distinct literals in conjunctions, but introduce another more natural measurement of their size.

The lemma should now follow from multiplying the number of conjunctions that have c distinct literals by the number of inputs accepted by each such conjunction. $\square$

The following lemma is analogous and it is proved by bounding the number of errors introduced on negative test graphs when f_e^{C} is formed from f_e^{D}.

Generic Lemma C. At an $\vee$ gate e, the approximator f_e^{C} introduces an error for at most $\beta(n, c, d)$ negative test graphs.

To finish the proof of the generic theorem we need to consider the two cases from Generic Lemma A. First, if the approximators fail for all negative test graphs, Generic Lemma C implies that $\mathrm{size}(\Psi) \geq \gamma_0(n)/\beta(n, c, d)$. Otherwise the approximators fail for at least half of the positive test graphs and then Generic Lemma B implies that $\mathrm{size}(\Psi) \geq \gamma_1(n)/(2\alpha(n, c, d))$. So if $h(n)$ is less than both these values, we are done. $\square$

4 Andreev's polynomial problem

The best lower bound for the size of monotone circuits computing any monotone function was proved by Alon and Boppana [1]. The function for which this bound was obtained was the same that Andreev [3] used when he was the first to prove an exponential lower bound for a monotone problem.

We are given a bipartite graph $G = (U, V, E)$ with vertex sets $U = \mathrm{GF}[q]$ and $V = \mathrm{GF}[q]$, where q is a prime power. The problem $\mathrm{POLY}(q, s)$ is to decide whether there exists a polynomial p over $\mathrm{GF}[q]$ of degree at most $s-1$ such that $\forall i \in U\colon (i, p(i)) \in E$.

Theorem 3 (Alon and Boppana [1]). *The monotone circuit complexity for* $\mathrm{POLY}(q, s)$ *is* $q^{\Omega(s)}$ *when* $s \leq \frac{1}{2}\sqrt{q/\ln q}$.

Proof. We first define the set of positive test graphs used in the proof. There are q^s different polynomials of degree at most $s - 1$, each of which corresponds to a positive test graph in a natural way: the polynomial p defines a test graph with the edges $E = \{(i, p(i)) \mid i \in U\}$.

The negative test graphs are constructed randomly, by having each possible edge appear with probability $1-\varepsilon$ for $\varepsilon = (2s \ln q)/q$. Notice that this construction may result in all possible bipartite graphs on $2q$ vertices; therefore we need the following lemma.

Lemma 4. *The probability that a negative test graph is in* $\mathrm{POLY}(q,s)$ *is at most* q^{-s}*.*

Proof. The probability that the q specific edges that correspond to a certain polynomial are present in a randomly chosen negative test graph is $(1-\varepsilon)^q$. So the probability that the edges corresponding to at least one of the q^s possible polynomials are present in a random graph is at most

$$q^s(1-\varepsilon)^q \leq q^s e^{-\varepsilon q} = q^s q^{-2s} = q^{-s}. \qquad \square$$

We choose the parameters $c = s$ and $d = q^{2/3}/2$.

Lemma 5 (Specialization of Generic Lemma A)**.** *At the output gate* e_o*, we either have that the approximator* $f_{e_o}^{\mathrm{C}}$ *is identically 1, or that the approximators for* e_o *fail for at least half of the positive test graphs.*

Proof. If $f_{e_o}^{\mathrm{C}}$ is identically 1 we are in the first case. Otherwise, there is a graph b such that $f_{e_o}^{\mathrm{C}}(b) = 0$, and hence there is a disjunction D in $f_{e_o}^{\mathrm{C}}$. There are q^{s-1} positive test graphs that contain any specific edge since fixing an edge is equivalent to fixing the value of a polynomial in one point. Thus, there are at most $dq^{s-1} = q^{s-1/3}/2$ positive test graphs that contain at least one edge from D. Hence D (and therefore also $f_{e_o}^{\mathrm{C}}$) is 1 for at most a fraction $q^{s-1/3}/(2q^s) = q^{-1/3}/2$ of the positive test graphs. $\square$

Lemma 6 (Specialization of Generic Lemma B)**.** *At an* $\wedge$ *gate* e*,* f_e^{D} *introduces an error for at most* d^c *positive test graphs.*

Proof. When rewriting f_e^{C} to f_e^{D}, we can form at most d^c conjunctions that contain c distinct literals, and for the current function, each removed conjunction introduces an error on at most one positive test graph. This follows since $c = s$ and since a polynomial of degree $s-1$ is completely specified by s function points. $\square$

Lemma 7 (Specialization of Generic Lemma C)**.** *At an* $\vee$ *gate* e*, the probability that* f_e^{C} *introduces an error for a random negative test graph is at most* $(c\varepsilon)^d$*.*

Proof. We bound the probability of an error being introduced for a random negative test graph by multiplying the number of disjunctions with d distinct literals, c^d, by the probability that the d literals are 0 in one disjunction, which is ε^d. $\square$

Notice that, in particular, the upper bound in the lemma holds for the number of errors introduced for negative test graphs that are not in the language $\mathrm{POLY}(q,s)$.

To prove the theorem we have to consider the two cases from Lemma 5. If the approximators fail for at least half of the positive test graphs we use

Lemma 6 which states that the number of errors introduced in a single $\wedge$ gate is at most d^c to get

$$\operatorname{size}(\Psi) \geq \frac{q^s}{2d^c} = \frac{2^s q^{s/3}}{2} \in q^{\Omega(s)}.$$

If, on the other hand, the approximator $f^{\mathrm{C}}_{e_o}$ is identically 1, the approximators for e_o fail for a random negative test graph with probability $1 - q^{-s} \geq 1/2$ by Lemma 4. The probability of introducing an error in a single $\vee$ gate is at most $(c\varepsilon)^d$; thus

$$\begin{aligned}\operatorname{size}(\Psi) &\geq \frac{1}{2(c\varepsilon)^d} \\ &= \frac{1}{2}\left(\frac{q}{2s^2 \ln q}\right)^{q^{2/3}/2} = \frac{1}{2} 2^{q^{2/3}/2} \in q^{\Omega(s)}. \qquad \square\end{aligned}$$

5 Broken Mosquito Screens

The Broken Mosquito Screens (BMS) problem was introduced by Haken [7] to illustrate how to count bottlenecks to show that monotone P $\neq$ NP. In this section we show that the same result can be obtained using our approximator formalism.

Haken defines the BMS problem for graphs with m^2-2 vertices as the problem of distinguishing *good* and *bad* graphs from each other. Graphs containing $m-1$ cliques of size m and one clique of size $m-2$ (but no other edges) are good; graphs containing $m-1$ independent sets of size m and one independent set of size $m-2$ but all other edges are bad. Hence, by taking the dual of a good graph (i.e., inverting all the input bits) we get a bad graph. Notice that not all graphs are either good or bad, but the definition can be extended to include all graphs. Haken shows that a monotone circuit that distinguishes good graphs from bad must be large.

We use the following extended definition of the BMS problem.

Definition 8. *Instances of* $\mathrm{BMS}(m)$ *are graphs with* m^2-2 *vertices* ($m > 2$)*. A graph is in the language* $\mathrm{BMS}(m)$ *if there exists a partition of the vertices into* $m-1$ *sets of size* m *and one of size* $m-2$*, so that each of these subsets forms a clique.*

Using this definition, it is easy to see that the BMS problem is monotone and in NP.

Theorem 9 (Haken [7])**.** *The monotone circuit complexity for* $\mathrm{BMS}(m)$ *is* $2^{\Omega(\sqrt{m})}$*.*

Proof. We let the set of positive test graphs G for the BMS problem be all graphs with m^2-2 vertices that contain $m-1$ cliques of size m and one clique of size $m-2$, but no other edges. The set of negative test graphs B is all graphs on m^2-2 vertices that contain $m-1$ independent sets of size m and one independent set of size $m-2$, but where all other edges are present. Our positive and negative test graphs correspond to Haken's good and bad graphs respectively.

Clearly, all positive test graphs are in the BMS language. Negative test graphs are not in the BMS language since they contain only $m-2$ cliques of size m.

Next we count the number of test graphs. Counting the number of positive test graphs is the same as counting the number of ways that a set of m^2-2 elements can be partitioned into $m-1$ sets containing m elements and one set containing $m-2$ elements. We get

$$|G| = \frac{(m^2-2)!}{(m!)^{(m-1)}(m-2)!(m-1)!}, \tag{1}$$

and by duality $|B| = |G|$.

For each conjunction and disjunction, let a graph correspond to it in the natural way: for every literal, include the corresponding edge in the graph.

Suppose we want to determine whether a positive test graph satisfies a conjunction C. This is the case when the graph corresponding to C is a subgraph of the positive test graph. Therefore, we only have to know how the vertices of the graph corresponding to C are divided into connected components to determine what positive test graphs are accepted by C.

In this section we do not limit the length of conjunctions and disjunctions by their number of literals, but instead we define their size by m^2-2 minus the number of connected components in their corresponding graphs, and require that their size is less than c and d respectively.

We choose $c = d = \lfloor\sqrt{m}\rfloor$ (since $c = d$ we only use c), i.e., the graphs corresponding to the conjunctions and disjunctions in the approximators have more than $m^2 - 2 - \lfloor\sqrt{m}\rfloor$ connected components. Note that, implicitly, the number of literals in conjunctions and disjunctions is bounded by $c^2/2 \le m/2$.

Lemma 10 (Specialization of Generic Lemma A)**.** *At the output gate e_o, the approximators either fail for all negative test graphs, or they fail for at least half of the positive test graphs.*

Proof. If the approximator $f^{\mathrm{C}}_{e_o}$ for the output gate fails for all the graphs in B we are in the first case. Otherwise, let $b \in B$ be a graph for which the approximator does not fail at e_o.

Since $f^{\mathrm{C}}_{e_o}(b) = 0$, there is a disjunction D in $f^{\mathrm{C}}_{e_o}$, and as noted above, this disjunction contains less than $c^2/2 \le m/2$ literals.

The fraction of graphs in G that contain a specific one of these literals is less than $1/m$, which can be seen as follows.

A graph in G has $(m-1)\binom{m}{2} + \binom{m-2}{2}$ edges out of the possible $\binom{m^2-2}{2}$ ones. Since all edges are equally likely, this means that a specific edge appears in a fraction

$$\frac{(m-1)\binom{m}{2} + \binom{m-2}{2}}{\binom{m^2-2}{2}} < \frac{1}{m}$$

of the graphs in G.

Thus, the fraction of G that contains at least one of the $m/2$ literals in D is at most $1/2$. So for at least half of all $g \in G$, we have $D(g) = 0$ and therefore also $f^{\mathrm{C}}_{e_o}(g) = 0$. □

Next, we establish an upper bound for the number of test graphs for which the approximators introduce an error at a single gate.

Lemma 11 (Specialization of Generic Lemma B)**.** *At an $\wedge$ gate e, the approximator f_e^{D} introduces an error for at most*

$$\frac{m^{4c}(m^2-2-2c)!}{2^c(m!)^{(m-1)}(m-2)!(m-1)!} \tag{2}$$

positive test graphs.

Proof. We introduce errors on positive test graphs by removing conjunctions whose corresponding graphs contain at most m^2-2-c connected components. When counting the number of positive test graphs for which an error is introduced, we consider different orderings of the vertices within the partitions and the order of the partitions as different graphs, and in the end we divide by the same denominator as in (1).

When building the tree, let w be the node that was created while expanding the disjunction D_i, and let C be the conjunction corresponding to w.

Consider the case that D_{i+1} contains a literal $x_{u,v}$ for which both endpoints are in the same connected component in the graph corresponding to C. Then, we create only one single child under w and label the edge $x_{u,v}$. In this case, we know that $x_{u,v}$ is satisfied if C is, so dropping all other children does not introduce errors for any positive test graphs. Clearly, ignoring to create some children never introduces errors for negative test graphs.

Otherwise, w gets less than $c^2/2$ children, since each disjunction contains less than $c^2/2$ literals. In this case the number of connected components decreases by one for the children, so we will make c such expansions before we reach a node where the graph for the corresponding conjunction contains m^2-2-c connected components. Therefore, there are at most $(c^2/2)^c \leq m^c/2^c$ such conjunctions.

We now count the number of inputs accepted by a single conjunction whose corresponding graph H contains m^2-2-c connected components. Let a denote the number of vertices in H that are part of connected components with more than one vertex. It follows that the number of isolated vertices in H is m^2-2-a, and that the number of connected components in H with more than one vertex is $a-c$.

To get an upper bound for the number of positive test graphs that are compatible with H we count as follows. Each connected component in H with more than one vertex must go into one of the m partitions, which makes at most m^{a-c} choices. Then, each vertex in those connected components can be placed in at most m ways each within the partition. Lastly, the remaining m^2-2-a vertices can be placed freely in $(m^2-2-a)!$ ways. To sum up, the total number of choices is at most

$$m^{a-c}m^a(m^2-2-a)! = m^{2a-c}(m^2-2-a)!,$$

which is increasing with a. Since the graph H contains $a-c$ connected components with more than one vertex, the total number a of vertices in such components is at most $2c$. This follows since the number of connected components with more than one vertex is at most c. So an upper bound for the number of choices is

$$m^{3c}(m^2-2-2c)!.$$

Multiplying the maximum number of conjunctions whose corresponding graphs contain m^2-2-c connected components by the maximum number of inputs accepted by each, and finally dividing with the same denominator as in (1) yields the lemma. □

The proof of the lemma above differs slightly from the corresponding proof by Haken. The reason for this is that we make a construction of the approximator for an $\wedge$ gate from the approximators for the input gates, whereas Haken only proves the existence of a set of short conjunctions that describe the gate well enough.

Because of the duality of the test graphs and since $c = d$, the following lemma can be proved analogously to the previous one.

Lemma 12 (Specialization of Generic Lemma C)**.** *At an $\vee$ gate e, the number negative test graphs for which the approximator f_e^{C} introduces an error is bounded by (2).*

We now consider the two cases from Lemma 10. Either the approximators for the output gate e_o fail for at least half the positive test graphs, or they fail for all negative test graphs. Since $|G| = |B|$, and since we have the bound (2) for the maximum number of errors introduced in a single gate, the minimum size of Ψ is

$$\frac{2^c(m^2-2)!}{2m^{4c}(m^2-2-2c)!}.$$

If we expand this expression we get

$$\begin{aligned}
\mathrm{size}(\Psi) &\geq 2^c \frac{(m^2-2)\cdots(m^2-2-2c+1)}{2m^{4c}} \\
&\geq \frac{2^c}{2}\frac{(m^2-2-2c+1)^{2c}}{(m^2)^{2c}} \\
&= \frac{2^{\lfloor\sqrt{m}\rfloor}}{2}\left(\frac{m^2-2\lfloor\sqrt{m}\rfloor-1}{m^2}\right)^{2\lfloor\sqrt{m}\rfloor} \\
&\in 2^{\Omega(\sqrt{m})}. \quad \square
\end{aligned}$$

6 Clique

In this section we prove that deciding whether a graph on n vertices contains any complete subgraph of size m (the problem CLIQUE(m,n)) requires monotone circuits of exponential size.

Theorem 13 (Alon and Boppana [1])**.** *The monotone circuit complexity for* CLIQUE(m,n) *is* $2^{\Omega(\sqrt{m})}$ *when* $m \leq n^{2/3}$.

Proof. Let the positive test graphs be all possible graphs on n vertices with a clique of size m and no other edges. This makes $\binom{n}{m}$ positive test graphs. We make one negative test graph for each possible coloring of the n vertices using $m-1$ colors by connecting vertices of different colors by edges. Note that some colorings result in the same graph, but we treat them as different for counting purposes and get $(m-1)^n$ negative test graphs.

As in the proof of the BMS problem, we define the size of disjunctions by n minus the number of connected components in their corresponding graphs, and require that their size is less than d.

For the conjunctions, however, we introduce a new notation of size that counts the number of vertices they *touch.* The number of vertices that a conjunction touches is the number of different vertices to which any of the edges connect. A conjunction is required to touch less than c vertices.

Choose $c = \lfloor\sqrt{m}\rfloor$, so that conjunctions touch less than $\lfloor\sqrt{m}\rfloor$ vertices. Implicitly, the number of literals in conjunctions is bounded by $c^2/2 \leq m/2$.

Let $d = \lfloor n/(8m)\rfloor$ so that the graphs corresponding to disjunctions have more than $n - \lfloor n/(8m)\rfloor$ connected components; this implies that they touch less than $2d \leq n/(4m)$ vertices.

Lemma 14 (Specialization of Generic Lemma A)**.** *At the output gate e_o, the approximators either fail for all negative test graphs or they fail for at least half of the positive test graphs.*

Proof. Assume that there is a negative test graph b such that $f^{\mathrm{C}}_{e_o}(b) = 0$ (otherwise we are done already). Then there is a disjunction D in $f^{\mathrm{C}}_{e_o}$, and we show that this disjunction can be 1 for at most half of the positive test graphs.

A necessary condition for D to be 1 on a positive test graph g is that one of the vertices it touches is in the clique of g. Since any given vertex is part of the clique in a fraction m/n of the positive test graphs, a collection of at most $n/(8m)$ vertices has a vertex in common with less than half the positive test graphs. □

Lemma 15 (Specialization of Generic Lemma B)**.** *At an $\wedge$ gate e, the approximator f^{D}_e introduces an error for at most*

$$\binom{n-c}{m-c}\left(\frac{n}{2m}\right)^c$$

positive test graphs.

Proof. When building the tree, let w be the node that was created while expanding the disjunction D_i, and let C be the conjunction corresponding to w.

Consider the case that D_{i+1} contains a literal $x_{u,v}$ for which both endpoints are already touched by C. Then a positive test graph that satisfies C also satisfies D_{i+1}, so we only create one single child under w containing $x_{u,v}$.

Otherwise, D_{i+1} contains a mix of literals, for some none of the two endpoints are touched by C, and for some one endpoint is touched by C.

First consider the literals that have one endpoint that is already touched by C: they all have another endpoint that is not touched by C. For each such endpoint, arbitrarily select one of the literals that connects to the endpoint, and form one child for it. The reason that we may disregard some literals from consideration is that two conjunctions that touch the same vertices will accept the same positive test graphs. The total number of children created this way is less than $2d$ since D_{i+1} touches at most $2d$ vertices.

For the literals that connect to two vertices that are not touched by C, we first select one of their endpoints in some arbitrary way and create a child for it; however, we do not label the edges to these children. At most $2d$ such children are created. Then, we create less than $2d$ children under each of these children for the other endpoint, and the edges down to these children are labeled by literals.

The way we have constructed the tree implies that no node has more than $4d = n/(2m)$ children, and when descending to a child from a node with more than one child, the number of vertices touched by the corresponding conjunction increases by 1. Therefore, there are at most $(n/(2m))^c$ conjunctions touching c vertices.

The number of positive test graphs accepted by a single conjunction touching c vertices is the number of ways to choose the remaining $m - c$ vertices for the clique out of the total remaining $n - c$ vertices, which is $\binom{n-c}{m-c}$.

Finally, the lemma follows from multiplying the bound for the number of conjunctions touching c vertices by the maximum number of inputs they may accept. □

Lemma 16 (Specialization of Generic Lemma C)**.** *At an $\vee$ gate e, the approximator f_e^{C} introduces an error at most $(m/2)^d(m-1)^{n-d}$ negative test graphs.*

Proof. We start by bounding the number of disjunctions corresponding to nodes in the tree, whose corresponding graphs contain $n - d$ connected components.

When building the tree, let w be the node that was created while expanding the conjunction C_i, and let D be the disjunction corresponding to w.

Consider the case that C_{i+1} contains a literal $x_{u,v}$ for which both endpoints are in the same connected component in the graph corresponding to D. Then a negative test graph that falsifies D also falsifies C_{i+1}, since u and v must have the same color. It is therefore enough to create only one single child under w and label the edge $x_{u,v}$.

Otherwise, w gets at most $m/2$ children, since each conjunction contains at most $m/2$ literals. In this case the number of connected components decreases by one for the children, so we will make d such expansions before we reach a node for which the graph for the corresponding disjunction contains $n - d$ connected components. In all, there can hence be at most $(m/2)^d$ such disjunctions.

It remains to count the number of negative test graphs rejected by a single disjunction whose corresponding graph H contains $n-d$ connected components. The negative test graphs that are rejected by such a disjunction are those that use only one color within each connected component, and their number is $(m-1)^{n-d}$.

Multiplying the number of disjunctions whose corresponding graphs contain $n - d$ connected components by the number of negative test graphs rejected by each yields the bound in the lemma. □

We now determine the lower bound for the circuit size. There are two cases to consider: either $f_{e_o}^{\mathrm{D}}$ fails on at least half the positive test graphs so that

$$\begin{aligned}
\operatorname{size}(\Psi) &\geq \frac{\binom{n}{m}}{2\binom{n-c}{m-c}\left(\frac{n}{2m}\right)^c} \\
&= \frac{1}{2}\frac{n!\,(m-c)!}{(n-c)!\,m!}\frac{(2m)^c}{n^c} \\
&\geq \frac{1}{2}2^c\frac{(n-c)^c}{m^c}\frac{m^c}{n^c} \\
&\in 2^{\Omega(\sqrt{m})},
\end{aligned}$$

or the approximators fail for all negative test graphs so that

$$\begin{aligned} \text{size}(\Psi) &\geq \frac{(m-1)^n}{(m-1)^{n-d}(m/2)^d} \\ &= 2^d \left(\frac{m-1}{m}\right)^d \\ &\in 2^{\Omega(n/m)}. \end{aligned}$$

Since $2^{\Omega(\sqrt{m})} \subseteq 2^{\Omega(n/m)}$ for $m \leq n^{2/3}$, the theorem follows. □

Acknowledgment

We thank Johan Håstad for helpful discussions and comments on earlier versions of this paper.

References

[1] Noga Alon and Ravi B. Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7:1–22, 1987.

[2] Kazuyuki Amano and Akira Maruoka. Potential of the approximation method. In *Proc. 37th Ann. IEEE Symp. Found. Comput. Sci.*, pages 431–440, 1996.

[3] Alexander E. Andreev. On a method for obtaining lower bounds for the complexity of individual monotone functions. *Sov. Math. Dokl.*, 31:530–534, 1985.

[4] Christer Berg, Staffan Ulfberg, and Avi Wigderson. Symmetric approximation for monotone real circuits. manuscript, 1996.

[5] Ravi B. Boppana and Michael Sipser. The complexity of finite functions. In Jan van Leeuwen, editor, *Handbook of theoretical computer science*, volume A, Algorithms and complexity, pages 757–804. Elsevier/MIT Press, 1990.

[6] Paul Erdős and Richard Rado. Intersection theorems for systems of sets. *J. London Math. Soc.*, 35:85–90, 1960.

[7] Armin Haken. Counting bottlenecks to show monotone P $\neq$ NP. In *Proc. 36th Ann. IEEE Symp. Found. Comput. Sci.*, pages 36–40, 1995.

[8] Armin Haken and Stephen Cook. An exponential lower bound for the size of monotone real circuits. Submitted to J. Comput. System Sci., 1996.

[9] Stasys Jukna. Finite limits and monotone computations over the reals. In *Twelfth Annual IEEE Conference on Computational Complexity*, 1997.

[10] Pavel Pudlák. Lower bounds for resolution and cutting planes proofs and monotone computations. *J. Symbolic Logic, to appear*, 1997.

[11] Alexander A. Razborov. Lower bounds on the monotone complexity of some boolean functions. *Sov. Math. Dokl.*, 31:354–357, 1985.

[12] Alexander A. Razborov. On the method of approximations. In *Proc. Twenty-first Ann. ACM Symp. Theor. Comput.*, pages 167–176, 1989.

[13] Janos Simon and Shi-Chun Tsai. A note on the bottleneck counting argument. In *Twelfth Annual IEEE Conference on Computational Complexity*, 1997.

[14] Jürgen Tiekenheinrich. A $4n$ lower bound on the monotone network complexity of a one-output boolean function. *Inform. Process. Lett.*, 18:201–202, 1984.