An Exponential Lower Bound for the Size of Monotone Real Circuits

Armin Haken Stephen Cook

February 26, 1997

Abstract

We prove a lower bound, exponential in the eighth root of the input length, on the size of monotone arithmetic circuits that solve an NP problem related to clique detection. The result is more general than the famous lower bound of Razborov and Andreev, because the gates of the circuit are allowed to compute arbitrary monotone binary real-valued functions (including AND and OR). Our proof is relatively simple and direct, and uses the method of counting bottlenecks. The generalization was proved independently by Pudlák using a different method, who also showed that the result can be used to obtain an exponential lower bound on the size of unrestricted cutting plane proofs in the propositional calculus.

1 Introduction

The Razborov/Andreev [Raz85] [AB87] [And85] exponential lower bound on the size of monotone Boolean circuits which detect cliques represented a breakthrough in the theory of monotone circuit complexity. The proof introduced the method of approximation, which has been used for other important lower bounds (see [BS90] and [Weg87] for expositions).

Here we use the method of *bottlenecks* to give a simpler proof of a similar result. Our lower bound applies not just to monotone Boolean circuits, but more generally to circuits using gates which compute arbitrary monotone binary real-valued functions. The bound is proved specifically for the broken mosquito screen problem (BMS), an NP problem related to clique detection. The result implies an exponential lower bound for the clique problem itself, since the clique problem is complete with respect to polynomial monotone projections for "monotone NP" (see [BS90]). An instance of the BMS is presented to the circuit as an n-tuple of values from $\{0,1\}$, and the output is either 0 or 1. We prove that every monotone real circuit which solves the BMS for some n must have a number of gates which is at least exponential in the eighth root of n.

The method of bottlenecks maps a large set of input vectors to gates in the circuit so that not too many vectors are mapped to any one gate. Dividing an underestimate of the size of the mapped set by an overestimate of how many can be mapped to one gate, yields exponentially many gates. The mapping manages to hit "bottlenecks" in the circuit by sending an input vector to the first gate in the circuit at which a certain amount of *progress* in classifying the input is made. The measure of progress is the length of a *fence* which intuitively keeps track of how many bits of the input are actually used by the computation at that point.

This method is similar to that used by Haken [Hak85] (and subsequently others [Urq87] [CS88]) to prove an exponential lower bound on the length of resolution proofs in the propositional calculus. In that case a large set of critical truth assignment vectors was mapped to clauses in the proof.

The result in the present paper has an application to cutting plane proofs [CCT87] in the propositional calculus. Cutting plane proofs provide a complete refutation system for unsatisfiable sets of propositional clauses. They efficiently simulate resolution proofs, and in fact are known to provide exponentially shorter proofs on some examples (the pigeonhole clauses). An exponential lower bound for cutting plane proofs for clause sets based on a clique/coclique distinction, under the restriction that coefficients in the proof are polynomially bounded, was proved in [BPR95], by reducing the problem to lower bounds for monotone Boolean circuits, and applying the Razborov/Andreev result mentioned above. Building on this and other work, Pudlák [Pud95A] showed how the restriction on coefficient size could be eliminated if the monotone circuit lower bound could be generalized to apply to circuits with monotone arithmetic gates. He later showed [Pud95B] how to adapt the approximation method of Razborov to obtain the needed lower bound, thus providing an independent proof of the main result in the present paper (for the case of the clique problem).

As stated above, our lower bound for BMS implies a lower bound for the clique problem, and therefore by Pudlák's argument it implies a lower bound for cutting plane proofs of the clause sets based on the clique/co-clique distinction. However our lower bound more directly implies an exponential lower bound for clause sets based on BMS. Each of these clause sets asserts that an instance to BMS is both "good" and "bad", and by Lemma 1 below it follows that each clause set is unsatisfiable. Pudlák's interpolation result shows how to transform a cutting plane refutation P of any one of these sets to a monotone real circuit C of size linear in the number of variables and inequalities in P, such that C separates good and bad instances of BMS. Thus by Theorem 1 below P must have exponential size.

The first author devised the main ideas in the proof presented here, and used them in the preliminary version [Hak95] of this paper to prove the lower bound for the case of monotone Boolean circuits. The second author showed how to generalize the proof to apply to arbitrary monotone real-valued gates.

2 Conventions

The inputs to the circuit are at the bottom and the output is at the top, so "lowest level" means closest to the input level. The circuit logic values can be any real numbers, but at the inputs to the circuit, the values are 0 and 1. Without loss of generality, the output values of the topmost gate of the circuit are also restricted to be 0 or 1. The output values 0 and 1 indicate rejection and acceptance respectively, of a vector of input values. The gates are unary or binary and are allowed to compute <u>any</u> monotone nondecreasing function of the inputs. Specifically, if the output is γ for inputs α and β, and the output is γ' for inputs α' and β', then $(\alpha \leq \alpha') \wedge (\beta \leq \beta') \Rightarrow (\gamma \leq \gamma')$. The results proved here are easily generalized to the case of bounded fan-in gates, but if unrestricted fan-in were allowed, the whole circuit could consist of just one gate.

The gates at a particular level in the circuit have a canonical left-right order defined by the circuit. For convenience, the inputs to the circuit are also considered to be gates. (If the negations of the inputs were also available to the circuit, it would be able to polynomially simulate a Turing machine. In that case a super-polynomial lower bound would imply P$\neq$NP.) When an input d is applied to the circuit, the output of gate E is denoted by $E(d)$.

3 Broken Mosquito Screens

The Broken Mosquito Screen problem (BMS) is a version of the CLIQUE problem for graphs that is designed to have an NP acceptance condition and also an NP rejection condition. Furthermore, it has a useful self-symmetry that allows the convenience of duality in definitions and arguments in the following proof. It is not a fully specified problem in that some graphs are "don't care" graphs and do not meet either the acceptance or the rejection condition. Our lower bound applies to any circuit which accepts and rejects when it should, regardless of what it does on don't care graphs. This feature of BMS is needed to get the lower bound for cutting plane proofs using Pudlák's method.

Definition 1 *Instances of BMS are encodings of graphs with $m^2 - 2$ vertices, where $m \geq 3$ is a convenient parameter for indexing. The graphs are represented in the standard way; as a string of bits that indicates for each pair of vertices whether there is an edge between them, with value 1 for the edge being present and value 0 for the edge absent.*

A graph is good, or accepted if there exists a partition of the vertices into $m-1$ sets of size m and one set of size $m-2$, so that each of these subsets forms a clique, i.e., all pairs of vertices within the subset are connected by edges. A graph is bad if there exists a partition of the vertices into $m-1$ subsets of size m and a subset of size $m-2$ so that each of these subsets forms an anticlique, i.e., no two of the vertices within a subset have an edge between them.

It is intuitively helpful to think of the graphs that are good instances of BMS drawn as an array of m rows and m columns of vertices, such that one row is missing two vertices. Within each row, all vertices are connected by edges. The bad instances can be thought of drawn with one column missing two vertices and no edges strictly within any one column. This image inspired the name "broken mosquito screen".

Clearly, both the acceptance and rejection conditions are NP. Since acceptance or rejection is not specified for all graphs, BMS is not a language in NP$\cap$co-NP, although the definition could easily be made more specific to yield a language in either NP or co-NP. To *solve* BMS means to separate the good from the bad instances. The essential combinatorial property of BMS is that the accepted and rejected instances are associated with partitions of the $m^2 - 2$ vertices into subsets of size m. By the pigeonhole principle, the accepted and rejected partitions cannot both exist in the same graph:

Lemma 1 *No instance of BMS is both good and bad. Furthermore, BMS is a monotone problem.*

Proof: Suppose the graph g is good. Let $V_1, \ldots, V_{m-1}$ be the required cliques of size m and let V_m be the remaining clique of size $m-2$. For g to also be bad, there must exist subsets $W_1, \ldots, W_{m-1}$ and W_m that are anticliques in g. Each set W_i for i from 1 to $m-1$ must contain exactly one vertex from each of V_1 to V_m. However, the W_i are supposed to be disjoint and V_m only contains $m-2$ vertices, so the required $m-1$ many W_i can't exist.

If more edges are added to a good graph (input values changed from 0 to 1) it continues to meet the acceptance condition with the same partition of the vertices. If edges are taken away from a bad graph, it continues to meet the rejection condition. ■

For a monotone circuit, the most difficult instances of BMS are the set of minimal good graphs and the set of maximal bad graphs:

Definition 2 *Define $\underline{G_0}$ to be the set of good instances of the BMS problem (graphs on m^2-2 vertices), that are minimal: Only the edges that are explicitly needed to meet the acceptance condition are present. Define $\underline{B_0}$ to be the set of bad instances of the BMS problem that are maximal: All edges are present except those that are explicitly required to be absent to meet the rejection condition.*

The lower bound argument only requires the monotone circuit to separate these two sets of graphs. Note that for a non-monotone circuit or for a Turing machine it is quite easy to separate G_0 from B_0. An algorithm just needs to count the number of edges at any one vertex.

4 The Lower Bound

The proof of the lower bound requires a sequence of definitions and lemmas, and takes up the remainder of this paper. To motivate the exposition, the theorem statement is given here.

Theorem 1 *Let $\underline{\Psi}$ be any monotone arithmetic circuit that solves the BMS problem with parameter $m \geq 5$. The circuit Ψ must contain at least $\frac{1.8^{\lfloor\sqrt{m/8}\rfloor}}{2}$ gates.*

In fact, all that is required of Ψ in the way of solving the BMS problem is that Ψ gives the output 1 on the set G_0 and output 0 on the set B_0. Since the input size $\underline{n}$ is $\frac{m^4-5m^2+6}{2}$, the circuit contains $2^{Kn^{1/8}}$ gates, for a constant $K > 0$. In fact, the estimates used in this paper yield $K > .32$. Using sharper estimates, the constant could be pushed past .38. Increasing the exponent of n to more than 1/8 would require new ideas. The theorem's **proof** is at the end of the paper.

4.1 Preview

The following subsections define a mapping μ from a subset A of $G_0 \cup B_0$ to the gates of Ψ, where $\|A\| \geq \|G_0\|$. For any gate E in Ψ, the ratio of $\|G_0\|/\|\mu^{-1}(E)\|$ is greater than $\frac{1.8^{\lfloor\sqrt{m/8}\rfloor}}{2}$. The mapping μ is defined sequentially. First one element of $G_0 \cup B_0$ is mapped and then that element is deleted from $G_0 \cup B_0$ yielding the set $G_1 \cup B_1$. The procedure continues until the set $G_j \cup B_j$ becomes too small for the further specification of μ to make sense. The set A is then $(G_0 \cup B_0) \setminus (G_j \cup B_j)$. It will be shown that "not too many" elements of A can be mapped to any one gate in Ψ. The mapping μ always sends a graph to a gate at which the circuit is specifically concerned with processing that graph.

4.2 Fences

To define the mapping μ, a measure of "progress" is needed. An input vector g (which is the encoding of a graph) is mapped to a gate E at which the circuit Ψ makes particular progress in classifying g. Further argumentation shows that progress is not made for too many graphs at that same gate E. The desired measure of progress turns out to be "the length of a minimal fence".

Definition 3 *Let E be a gate in Ψ and let g be a good BMS graph in the set G_i. A $\underline{\text{fence}}$ around g at gate E and at time i is a conjunction $C = x_1 \wedge \cdots \wedge x_q$ where $x_1, \ldots, x_q$ are inputs to Ψ, (variables that take on the values 0 and 1, representing edges in the graph). Furthermore, $C(g) = 1$ and $(\forall b' \in B_i)[(E(b') < E(g)) \Rightarrow (C(b') = 0)]$. In other words, the fence C is required to compute just as good a separation of g from the set B_i, as does the gate E. The $\underline{\text{length}}$ of this fence C is the number of literals q. A $\underline{\text{minimal}}$ fence around g at gate E and time i is a minimum length fence for g, E and i. Dually, for a gate E in Ψ and b in B_i, a $\underline{\text{fence}}$ around b at gate E and at time i is a* *disjunction* $D = y_1 \vee \cdots \vee y_r$ *where* $y_1, \ldots, y_r$ *are inputs to* Ψ*, so that* $D(b) = 0$ *and* $(\forall g' \in G_i)[(E(g') > E(b)) \Rightarrow (D(g') = 1)]$*. So* D *does just as good a job as does* E *on* b *and on* G_i*.*

Note that a fence around g is likely to get other good graphs wrong (evaluate to 0). A fence is only concerned with one good graph and a whole set of bad graphs, or vice versa. Also note that for a particular g, there is one conjunction that works as a fence for any gate E and any time i. That fence is the conjunction of all the edges in g, and has length $\frac{m(m-1)^2+(m-2)(m-3)}{2}$. So the concept of "shortest fence" is well-defined.

For any input x to Ψ such that x is an edge in g, meaning g is classified correctly at the "gate" x, there is a fence of length 1 that works for g at that input gate and at any time. That fence is of course, just x taken as a conjunction of one literal. If input y to Ψ is not an edge in g, the empty conjunction, which simply has the constant value 1 and has length 0, is a fence for g at y and at all times.

The dual situation is that for a bad graph b, the disjunction of literals representing all edges missing from b works as a fence for all gates at all times. Also, the obvious disjunction of length 1, namely x, is a fence for input x if x is an edge missing from b. If y is an edge in b, the empty disjunction, (length 0) that has constant value 0, is a fence for b at gate y.

4.3 Construction of the Mapping

To start, the constant k and the mapping μ are defined:

Definition 4 *Let* $\underline{k}$ *be defined as* $m/2$*. Call a fence* <u>*long*</u> *if it is longer than* $k/2$*, otherwise call it* <u>*short*</u>*.*

Initially, all minimal fences at the input level of Ψ are short, but all minimal fences at the output level are long.

Definition 5 *Let* $\underline{E_0}$ *be the gate at the lowest level, and leftmost within the level in* Ψ*, such that there exists a graph* $d_0 \in G_0 \cup B_0$ *that requires a long fence at* E_0 *and at time* 0 *(the minimal fence is long). Here the specification of leftmost is not important, it is just to be definite. Define* $\underline{\mu(d_0)}$ *to be* E_0 *and let* $\underline{G_1 \cup B_1}$ *be* $G_0 \cup B_0$ *with* d_0 *taken out of either* G_0 *or* B_0 *depending if it is a good or a bad graph. Now repeat the process as often as possible: At time*

i let $\underline{E_i}$ be the lowest level and leftmost gate in Ψ such that there is a graph $d_i \in G_i \cup B_i$ that d_i requires a long fence at E_i at time i. Define $\underline{\mu(d_i)}$ to be E_i and delete d_i from $G_i \cup B_i$ to get $\underline{G_{i+1} \cup B_{i+1}}$. Eventually, the remaining graphs have short fences at all gates, and no more are mapped.

4.4 Size of the Domain of the Mapping

The mapping μ can only be defined as long as there exist minimal fences in the circuit that are long. However, a lot of graphs must be mapped:

Lemma 2 *The number of graphs mapped by μ is at least $\|G_0\|$.*

Proof: If all the graphs in G_0 or all the graphs in B_0 are mapped, the lemma holds since $\|B_0\| = \|G_0\|$ by the symmetry of the definitions. Otherwise, let $G_j \cup B_j$ be the set of unmapped graphs at the time j when the definition of μ no longer can be continued. Let $b' \in B_j$ and $g' \in G_j$ be unmapped graphs.

• There exists a short fence D' around b' at the output gate $\underline{F}$ of Ψ at time j. That requires that $D'(g_j) = 1$ for all g_j in G_j, since $F(b') = 0$ and $F(g_j) = 1$. Let D' be $y_1 \vee \cdots \vee y_h$, where $h \leq k/2$.

• The fraction of graphs in G_0 that contain edge y_1 is less than $1/m$: Counting the size of G_0 is the same as counting how many ways a set of size $m^2 - 2$ can be partitioned into $m - 1$ subsets of size m with $m - 2$ elements left over. When a good graph is known to contain edge y_1 it means the two endpoints of the edge are in the same subset of the partition. If two vertices are chosen randomly, the chance of the second vertex being in the same subset as the first is the number of other vertices in that subset divided by the number of other vertices in the graph. That fraction is $(m-1)/(m^2-3)$ or less, which is less than $1/m$ (for $m > 3$). Similarly, the fraction of G_0 that contains the other literals in D' is less than $1/m$ per literal.

• So the fraction of G_0 that contains any of the h literals in D' must be less than h/m which is at most $1/4$. Therefore, more than three quarters of G_0 must be classified wrongly by D' and must already be mapped before time j.

• A dual argument involving g' and B_j shows that at least three quarters of B_0 are also mapped, so the size of the mapped set is at least $\frac{3}{4}\|G_0\| + \frac{3}{4}\|B_0\|$ which is more than $\|G_0\|$. ∎

To come up with a number for the size of G_0, consider picking vertices of a graph out of an urn to systematically construct a partition of the required type. There are

$$\frac{\binom{m^2-2}{m,\ldots,m,m-2}}{(m-1)!}$$

possibilities, where the "$m,\ldots,m$" part has $m-1$ repetitions of "m",and the denominator is due to the fact that the $m-1$ subsets of size m are indistinguishable to the partition and can be chosen in any order. So

$$\|G_0\| = \frac{(m^2-2)!}{(m!)^{(m-1)}(m-2)!(m-1)!}. \tag{1}$$

4.5 Counting Graphs Mapped to One Gate

The upper bound on the number of graphs mapped by μ to one particular gate is also calculated by counting the number of partitions that can arise in such graphs, but there are some restrictions on the partition that make this bound sufficiently small.

Lemma 3 *If gate E has input gates E_1 and E_2, and if at time i graph d has fences of length s_1 at E_1 and s_2 at E_2, then at time i graph d has a fence at gate E of length at most s_1+s_2. If gate E has only one input gate E_1, then d has a fence at gate E of length at most s_1.*

Proof: If d is good, then the fence at gate E is the conjunction of the fences at gates E_1 and E_2. Otherwise the fence is the disjunction.

Since E computes a monotone function of its inputs, whenever $E(g) > E(b)$, it follows that $E_1(g) > E_1(b)$ or $E_2(g) > E_2(b)$. It is straightforward to check that the requirements of being a fence are all met by the above specification. ■

Lemma 4 *The number of graphs from $G_0 \cup B_0$ that can be mapped by μ to any one gate in Ψ is at most*

$$2\frac{(km)^{r/2}(m^2-m)^{r/2}(m^2-2-r)!}{(m!)^{(m-1)}(m-2)!(m-1)!} \tag{2}$$

for $\underline{r}$ the greatest even number less than or equal to $\sqrt{m/2}$.

Proof: Let E be a gate in Ψ. This argument gives an upper bound on how many good graphs are mapped to E. By symmetry, the number of bad graphs mapped to E satisfies the same bound. Thus the initial factor 2 in the formula.

• Let g_i be the first (lowest index in the definition of μ) good graph mapped to E. Let $d_1, \ldots, d_s$ be a complete list of the graphs in B_i, listed in order of their value at E, such that $E(d_1) \leq E(d_2) \leq \cdots \leq E(d_s)$.

• Each of these bad graphs d_j has a short fence at each input to E at time i, else it would be mapped to that lower level gate. By Lemma 3, d_j has a fence D_j, of length at most k, at E and at time i. Let $D_j = (y_{j,1} \vee \cdots \vee y_{j,k})$, where literals might be repeated in D_j if there are less than k distinct ones.

• Consider some index h ($h \geq i$), such that g_h is mapped to E. The graph g_h lies outside of (evaluates to 1 at) all those fences D_j such that $E(d_j) < E(g_h)$. Suppose this condition is satisfied for the first t graphs d_j, so g_h lies outside each of $D_1, \ldots, D_t$, and $E(d_{t+1}) \geq E(g_h)$. Select one literal from each fence D_j for j from 1 to t, which represents an edge in g_h. The conjunction of this set of literals is a fence for g_h at time i (and hence at time h) and therefore must include more than $k/2$ distinct literals.

• Note that a list of more than $k/2$ distinct edges must contain r different endpoints where $\frac{r^2-r}{2} > k/2$. So r is at least $\sqrt{k+r}$ which is greater than $\sqrt{m/2}$. It is convenient for r to be even, so subtract 1 if r is odd.

• The numerical bound (2) is derived similarly to formula (1) above, by counting choices for partitioning the m^2-2 vertices. In fact, the counting is done so that the denominator of formula (2) is the same for the same reason; the ordering of the equally sized subsets and the ordering of the vertices within a subset is immaterial for the partition. The calculation counts ordered partitions, so it overcounts unordered partitions by a factor of $(m!)^{(m-1)}(m-2)!(m-1)!$.

• To count how many ways one could choose a graph g so that $\mu(g) = E$, proceed by choosing edges, and thereby vertices, so that the fences $D_1, D_2 \ldots$ are satisfied until r vertices have been chosen.

• In the case of D_1, one of at most k different edges $y_{1,1}$ to $y_{1,k}$ can be chosen. That choice dictates that the two endpoints are in the same subset of the partition. There are m of these subsets, and to justify dividing the formula by the denominator, any of the subsets must be possible for the two vertices. Furthermore, within the subset, any of the $m(m-1)$ ordering positions for the two chosen vertices must be possible. So, for the first two vertices chosen there are only $km(m^2 - m)$ choices instead of $(m^2 - 2)(m^2 - 3)$ as in formula (1).

• When satisfying fence D_j (for $j > 2$), several things can happen: If there are already two vertices v_1 and v_2 chosen to be in the same subset of the partition, and the literal representing the edge from v_1 to v_2 is one of the disjuncts in D_j, then no vertices are added to the partition and the procedure moves on to D_{j+1}. Otherwise, one of the edges $y_{j,1}$ to $y_{j,k}$ must be chosen. It might be impossible to make such a choice if all the edges in D_j run between vertices that are already assigned to different subsets. In that case, the partition is abandoned as an instance of overcounting the graphs that can be mapped to E.

• In case an edge from D_j can be chosen, the choice gives one or two "new" vertices that need to fit into the partition. If only one of the endpoints is new, the new vertex must go into the same subset as the other endpoint. To justify the denominator that converts ordered to unordered partition counting, any of at most $m - 1$ places in the subset must be possible for the new vertex. So at most $k(m - 1)$ choices are made to get one more vertex. If both vertices are new, there are m choices for which subset of the partition they go into, and at most $m^2 - m$ choices of position for the two vertices within the subset. So at most $km(m^2 - m)$ choices are made to get two more vertices.

• Once r vertices have been chosen and partitioned to satisfy fences, the partition is completed by choosing the remaining $m^2 - 2 - r$ vertices out of an urn as in formula (1).

• The numerator of formula (2) is an overestimate of the product of the number of choices possible while choosing r vertices to satisfy fences, times $(m^2 - 2 - r)!$ choices made out of the urn. When two vertices are chosen at once from a fence, the factors are "(km)" and "$(m^2 - m)$". When only one vertex is chosen, the factor is "$k(m - 1)$". The term "$k(m - 1)$" is less than "km" and less than "$(m^2 - m)$", so to be safe, assume all vertices are chosen in pairs, yielding $r/2$ factors "km" and $r/2$ factors "$(m^2 - m)$".

• The formula (2) results. ■

4.6 The Lower Bound Proof

To be proved is Theorem 1 above, that the circuit Ψ contains at least $\frac{1.8^{\lfloor\sqrt{m/8}\rfloor}}{2}$ gates.

Proof: According to Lemma 2 the number of graphs mapped by μ is at least $\|G_0\|$ which is formula (1):

$$\frac{(m^2-2)!}{(m!)^{(m-1)}(m-2)!(m-1)!}.$$

• By Lemma 4 the maximum number of graphs that can be mapped to one gate E of Ψ is formula (2):

$$2\frac{(km)^{r/2}(m^2-m)^{r/2}(m^2-2-r)!}{(m!)^{(m-1)}(m-2)!(m-1)!}.$$

• So the quotient of formula (1) over formula (2) is a lower bound for the size of Ψ. That quotient is

$$\frac{(m^2-2)!}{2(km)^{r/2}(m^2-m)^{r/2}(m^2-2-r)!}. \tag{3}$$

• Note that apart from the "2" in the denominator, both the numerator and the denominator of formula (3) are products of m^2-2 factors. Of these factors, the last m^2-2-r cancel directly. The factors of interest are the first $r/2$. For the next $r/2$ factors, the numerator is always greater than the denominator, so canceling those factors preserves the inequality. The smallest of the first $r/2$ factors in the numerator is $(m^2-1-r/2)$, so formula (3) is greater than

$$\frac{(m^2-1-r/2)^{r/2}}{2(km)^{r/2}}. \tag{4}$$

• For all but very small m, $(m<5)$, the quantity $m^2-1-r/2$ is greater than $.9m^2$, and by definition of k, $km=.5m^2$. Thus, formula (4) is greater than $\frac{1.8^{r/2}}{2}$, giving the bound claimed in the theorem, since $r/2=\lfloor\sqrt{m/8}\rfloor$. ∎

References

[AB87] N. Alon and R. B. Boppana. The monotone circuit complexity of Boolean functions. *Combinatorica* 7, 1, pages 1–22, 1987.

[And85] A. E. Andreev. On a method for obtaining lower bounds for the complexity of individual monotone functions. *Doklady Akad. Nauk SSSR* 282, 5, pages 1033–1037, 1985. English translation in *Soviet Math. Dokl.* 31, pages 530–534, 1985.

[BPR95] M. Bonet, T. Pitassi, and R. Raz. Lower bounds for cutting planes proofs with small coefficients. In *Proc. Twenty-seventh Ann. ACM Symp. Theor. Comput.*, pages 575–584, 1995.

[BS90] R. B. Boppana and M. Sipser. The complexity of finite functions. In *Handbook of Theoretical Computer Science*, Vol. A, *Algorithms and Complexity*, J. van Leeuwen, Ed., MIT Press, pages 757–804, 1990.

[CS88] V. Chvátal and E. Szemerédi. Many hard examples for resolution. *JACM* 35,4, pages 759–768, 1988.

[CCT87] W. Cook, C. R. Coullard, and Gy. Turán. On the complexity of cutting plane proofs. *Disc. Appl. Math.*, pages 25–38, 1987.

[Hak85] A. Haken. The intractability of resolution. *Theor. Comp. Sci.* 39, pages 297–308, 1985.

[Hak95] A. Haken. Counting bottlenecks to show monotone P≠NP. In *Proc. 36th Annual Symp. Foundations of Computer Science*, pages 36–40, 1995.

[Pud95A] P. Pudlák. Direct proofs of interpolation theorems for resolution and cutting planes proof systems. Manuscript, February, 1995.

[Pud95B] P. Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. To appear in *JSL*.

[Raz85] A. A. Razborov. Lower bounds on the monotone complexity of some Boolean functions. *Dokl. Akad. Nauk SSSR*, 281,4 pages 798–801, 1985. English translation in *Soviet Math. Dokl.* 31,pages 354–357, 1985.

[Urq87] A. Urquhart. Hard examples for resolution. *JACM* 34,1 pages 209–219, 1987.

[Weg87] I. Wegener. *The Complexity of Boolean Functions.* B.G. Teubner and John Wiley & Sons, 1987.

ARMIN HAKEN
DIMACS, Rutgers
Piscataway, NJ, 08855-1179, USA
`haken@dimacs.rutgers.edu`

STEPHEN A. COOK
Department of Computer Science
University of Toronto
Toronto, ON, M5S 1A4, CANADA
`sacook@cs.toronto.edu`