Electronic Colloquium on Computational Complexity, Report No. 17 (2003)

On converting CNF to DNF

Peter Bro Miltersen* Jaikumar Radhakrishnan† Ingo Wegener‡

March 27, 2003

Abstract

The best-known representations of boolean functions f are those as disjunctions of terms (DNFs) and as conjunctions of clauses (CNFs). It is convenient to define the DNF size of f as the minimal number of terms in a DNF representing f and the CNF size as the minimal number of clauses in a CNF representing f. This leads to the problem to estimate the largest gap between CNF size and DNF size. More precisely, what is the largest possible DNF size of a function f with polynomial CNF size? We show the answer to be $2^{n-\Theta(n/\log n)}$.

1 Introduction

Boolean functions $f\colon \{0,1\}^n \to \{0,1\}$ can be represented by circuits, formulas, branching programs, and many restricted variants of these fundamental computation models. Here we investigate depth-2, unbounded fan-in circuits over AND, OR, and NOT better known as DNFs and CNFs.

We repeat some well-known concepts (for more details see Wegener (1987)). The set of variables is denoted by $X_n = \{x_1, \ldots, x_n\}$. Literals are variables and negated variables. Terms (or monomials) are conjunctions of literals. Clauses are disjunctions of literals. Each boolean function can be represented as a disjunction of terms, mostly denoted as disjunctive normal form (DNF). It would be historically more correct to use the notion of disjunctive forms for all these representations and disjunctive *normal* form for the *unique* representation as disjunction of minterms or terms of length n. However, using the term DNF in the less restrictive sense now seems to be more or less standard in computer science, so we stick to this less restrictive notion in this paper. The size of a DNF equals the number of its terms and the DNF size of f is the size of a minimum-size DNF for f. By duality we can represent f as a conjunction

*Dept. of Computer Science, University of Aarhus, Denmark. e-mail: bromille@daimi.au.dk. Supported by BRICS, Basic Research in Computer Science, a centre of the Danish National Research Foundation.

†Tata Institute of Fundamental Resarch, Mumbai, India. email: jaikumar@tifr.res.in.

‡FB Informatik LS2, Univ. Dortmund, 44221 Dortmund, Germany. e-mail: wegener@ls2.cs.uni-dortmund.de. Supported by DFG-grant We 1066/9.

ISSN 1433-8092

of clauses called conjunctive normal forms (CNFs) and obtain the complexity measure CNF size of f.

We are interested in the maximal blow-up of size when switching from CNF representation to DNF representation (or vice versa). By the law of distributivity, CNFs with m clauses of length k can be simulated by DNFs with k^m terms not longer than m. If the clauses do not share any variable, this blow-up cannot be avoided. However, this implies that the number of clauses is very small compared to the number of variables. The main question of this paper is the following one which does not have such a simple answer.

What is the largest possible DNF size of functions $f\colon \{0,1\}^n \to \{0,1\}$ *with polynomial CNF size?*

The problem is motivated by its fundamental nature: DNF size and CNF size are fundamental complexity measures. Practical circuit designs like programmable logic arrays (PLAs) are based on DNFs and CNFs. Lower bounds on unbounded fan-in circuits are based on the celebrated switching lemma of Håstad (1989) which is a statement about converting CNFs to DNFs where some variables randomly are replaced by constants. Hence, it seems that the exact relationship between CNFs and DNFs ought to be understood as completely as possible. Fortunately, CNFs and DNFs have simple combinatorial properties allowing the application of current combinatorial arguments to obtain such an understanding. In contrast, the results of Razborov and Rudich (1997) show that this is not likely to be possible for complexity measures like circuit size and circuit depth.

Another motivation for considering the question is the study of SAT algorithms and heuristics with "mild" exponential behavior; a study which has gained a lot of momentum in recent years (e.g., Monien and Speckenmeyer (1985), Paturi et al. (1998), Dantsin et al. (2000), Schöning (2002), Hofmeister et al. (2002), and Dantsin et al. (2003)). Despite many successes, the following fundamental question is still open: Is there an algorithm that decides SAT of a CNF with n variables and m clauses (without any restrictions on the length of clauses) in time $m^{O(1)}2^{cn}$ for some constant $c < 1$? The obvious brute force algorithm solves the problem in time $m^{O(1)}2^n$. One method for solving SAT is to convert the CNF to a DNF, perhaps using sophisticated heuristics to keep the final DNF and any intermediate results small (though presumably not optimally small, due to the hardness of such a task). Once converted to a DNF, satisfiability of the formula is trivial to decide. A CNF-DNF conversion method for solving SAT, phrased in a more general constraint satisfaction framework was recently studied experimentally by Katajainen and Madsen (2002). Answering the question above limits the worst case complexity of any algorithm obtained within this framework.

A final motivation for considering the question is a purely combinatorial one. Consider the *monotone* version of our main question: What is the largest possible DNF size of a monotone function $f : \{0,1\}^n \to \{0,1\}$ with polynomial CNF size. For monotone functions f it is well known (see, e.g., Wegener (1987), Chapter 2, Theorem 2.2) that the minimal-size CNF consists of all prime clauses which are shortest clauses c containing only non-negated variables and implying that f equals 0, more precisely $c(a) = 0$ implies $f(a) = 0$. The dual statement (considering prime implicants rather than prime clauses) holds for the minimal-size DNF. We can consider the CNF consisting of all prime clauses as a set system (a non-uniform hypergraph) $S_1, S_2, \ldots, S_m \subseteq V = \{1, \ldots, n\}$ where S_i represents the clause consisting of all x_j, $j \in S_i$. A set $A \subseteq V$ corresponds to the term of all x_k, $k \in A$. Such a set A corresponds to a prime implicant of f iff $A \cap S_i \neq \emptyset$ for all $i \in \{1, \ldots, m\}$ and this property does not hold for any proper subset of A. Hence, the monotone version of our main question is equivalent to the following natural question of extremal combinatorics: *What is the maximum possible number of distinct* minimal vertex covers *of a non-uniform hypergraph containing n vertices and $n^{O(1)}$ edges?*

Somewhat surprisingly, the exact question we consider does not seem to have been considered before. However, related research has been made. As mentioned, Håstad's switching lemma can be considered as a result about approximating CNFs by DNFs. The problem of converting polynomial-size CNFs and DNFs into representations by restricted branching programs for the purpose of hardware verification has been considered since a long time (see Wegener (2000)). The best lower bounds for ordered binary decision diagrams (OBDDs) and read-once branching programs (BP1s) are due to Bollig and Wegener (1998) and are of size $2^{\Omega(n^{1/2})}$ even for monotone functions representable as disjunctions of terms of length 2.

In this paper, we prove tight bounds on the largest possible blow-up of size when converting polynomial-size CNFs to DNFs. Section 2 contains example functions where the blow-up is large:

- We present a function with a CNF of size $n^{O(1)}$ whose minimum-size DNF has size $2^{n-\Theta(n/\log n)}$.
- We present a *monotone* function with a CNF of size $n^{O(1)}$ whose minimum-size DNF has size $2^{n-\Theta(n \log\log n/\log n)}$.

In Section 3, we prove that the example for the general case is optimal:

- Any function with a CNF of polynomial size can be represented by a DNF of size at most $2^{n-\Omega(n/\log n)}$.

A main technical tool of our proof of the upper bound is Håstad's switching lemma, which is not surprising as this lemma can be interpreted as a lemma about approximating CNFs with DNFs. Nevertheless, we need to add some extra combinatorial machinery to get the good bound above. We do not have a better upper bound for monotone functions, even though the lower bound for monotone functions based on the example is smaller. Hence, our understanding of the monotone case is actually worse than our understanding of the general case. This is somewhat surprising; usually monotone complexity is easier to understand than general complexity.

For the class of CNF-DNF conversion based SAT algorithms described above, our results imply that no algorithm within this framework has complexity $m^{O(1)}2^{cn}$ for some constant $c < 1$, though we cannot rule out an algorithm of this kind with complexity $m^{O(1)}2^{n-\Omega(n/\log n)}$ which would still be a very interesting result.

For the problem of estimating the maximum size of the number of minimal vertex covers of a non-uniform hypergraph, our results have the following direct corollary:

– Any hypergraph with n vertices and $n^{O(1)}$ edges has at most $2^{n-\Omega(n/\log n)}$ distinct minimal vertex covers. Furthermore, for each n, there is a hypergraph with n vertices and $n^{O(1)}$ edges with $2^{n-\Theta(n \log\log n/\log n)}$ distinct minimal vertex covers.

We leave getting a tight bound as an interesting open problem.

2 Functions with a large blow-up

It is well known that the parity function PAR_n and its negation are the only functions with maximal CNF size (and also DNF size) which equals 2^{n-1}. The conjunction of parity functions on small variable sets has a small CNF size but one may expect that it has large DNF size.

Definition 1 *The function $f_{k,n} : \{0,1\}^n \to \{0,1\}$ is defined in the following way. The variable set is partitioned into $\lceil n/k \rceil$ sets $S_1, \ldots S_{\lceil n/k \rceil}$ where $|S_i| = k$ for $i < \lceil n/k \rceil$. The function g_i is the parity function of all S_i-variables and the function $f_{k,n}$ the conjunction of all g_i.*

In order to avoid messy notation we assume in the following that n/k is an integer (all sets S_i have size k).

Theorem 2 *The CNF size of $f_{k,n}$ equals $n \cdot 2^{k-1}/k$ and the DNF size of $f_{k,n}$ equals $2^{n-n/k}$.*

Proof Each parity function g_i has a CNF of size 2^{k-1} and the conjunction of these n/k CNFs leads to a CNF for $f_{k,n}$ of size $n \cdot 2^{k-1}/k$. For a lower bound on the CNF size we use the result (see, e.g., Wegener (1987), Chapter 2, Lemma 2.2) that the number of *essential* prime clauses is a lower bound on the CNF size. A prime clause is called essential if it is the only *prime* clause c covering some $a \in f^{-1}(0)$, i.e., $c(a) = 0$. Prime clauses of $f_{k,n}$ have to ensure that some g_i computes 0 and, therefore, each prime clause contains all S_i-variables for some i. Therefore, the prime clause c of g_i covering $a_i \in g_i^{-1}(0)$ is a prime clause of $f_{k,n}$. It is essential since it is the only one covering the input $(b_1, \ldots, b_{n/k})$ where $g_i(b_j) = 1$ for all $j \neq i$ and $b_i = a_i$.

If $f_{k,n}(a) = 1$, then $a = (a_1, \ldots, a_k)$ and $g_i(a_i) = 1$ for all i. In particular, $f_{k,n}(b) = 0$ for all Hamming neighbors b of a. This implies that the minterms covering $a \in f_{k,n}^{-1}(1)$ are essential prime implicants and that the disjunction of these minterms is the minimal-size DNF. We have 2^{k-1} choices for a_i and, therefore, $(2^{k-1})^{n/k} = 2^{n-n/k}$ choices for a. □

In order to discuss this result we consider some parameter settings:

- If k equals a positive integer c, the CNF size grow linearly and the DNF size equals $2^{n-n/c}$. Choosing c large enough, we obtain in the exponent a term larger than $(1-\epsilon)n$ for each constant $\epsilon > 0$.
- If $k = c \log n$, the CNF size grows polynomially and the DNF size equals $2^{n-c^{-1}n/\log n}$. Choosing c large enough, we obtain in the exponent an arbitrarily small constant for the $(n/\log n)$-term.
- If $k = o(n)$, the CNF size grows at most weakly exponential $(2^{o(n)})$ and the DNF size can be made to $2^{n-\alpha(n)}$ for each $\alpha = \omega(1)$.

Definition 3 *The function $f^*_{k,n}$ is defined with respect to the majority function g^*_i in the same way as $f_{k,n}$ with respect to g_i.*

The majority function on N variables computes 1 iff the input contains at least $\lceil N/2 \rceil$ ones. Majority is the monotone function with the largest number of $\binom{N}{\lceil N/2 \rceil}$ prime implicants (an argument based on Sperner's lemma) and the largest number of $\binom{N}{\lceil N/2 \rceil}$ prime clauses. In the following, we set $r := \binom{k}{\lceil k/2 \rceil}$. By Stirling's formula, $r = \Theta(2^k \cdot k^{-1/2})$.

Theorem 4 *The CNF size of $f^*_{k,n}$ equals $n \cdot r/k = \Theta(n \cdot 2^k \cdot k^{-3/2})$ and the DNF size of $f^*_{k,n}$ equals $r^{n/k} = 2^{n-\Theta((n \log k)/k)}$.*

Proof The proof is easier than the proof of Theorem 2 since it is sufficient to count the number of prime clauses and prime implicants. Clauses of $\lceil k/2 \rceil$ variables of the same block S_i are the prime clauses of $f^*_{k,n}$. It is obvious that each shortening is not a clause of $f^*_{k,n}$. Hence, the number of prime clauses equals $n \cdot r/k$.

In order to ensure that $f^*_{k,n}(a) = 1$ it is necessary and sufficient to have at least $\lceil k/2 \rceil$ ones in each block. Hence, a prime implicant of $f^*_{k,n}$ is a conjunction of prime implicants of each block. There are n/k blocks and r choices per block which implies the theorem. □

For $k = c \log n$, the CNF size is polynomially bounded and the DNF can be made as large as $2^{n-\varepsilon n \log n/\log\log n}$ for each $\varepsilon > 0$ by choosing c large enough.

3 Upper bounds on the blow-up

We use Håstad's switching lemma. The following version due to Beame (1994) is convenient for us.

Lemma 5 (Håstad, Beame) *Let a CNF f with clause length r on n variables be given and let ρ be a random restriction leaving l variables free. Then*

$$\Pr[f_\rho \text{ does not have a decision tree of depth } d] < (7rl/n)^d.$$

We first present an upper bound with a fairly easy proof. The bound is not optimal for the case of CNFs of polynomial size. We later obtain the optimal upper bound for this case as a refinement of the easy argument.

Theorem 6 *Let $f : \{0,1\}^n \to \{0,1\}$ be a function with a CNF of size at most $2^{o(n^{1/2})}$. Then f has a DNF of size at most $2^{n-\Omega(n^{1/2})}$.*

Proof

Let ρ be a random restriction to the variables of f leaving $n/2$ variables free.

Assume that the CNF for f contains some clause containing more than $n^{1/2}$ literals and fix such a clause. Assume without loss of generality that the clause has at least as many unnegated as negated variables. We want to bound from above the probability that the clause is *not* killed by ρ, i.e., the probability that the clause is *not* made trivially true under ρ. This is at most the probability that none of the unnegated variables of the clause is assigned 1 by ρ. With probability $1-2^{-\Omega(n)}$, ρ assigns the value 1 to some constant fraction of all the variables $\{x_1, x_2, \ldots, x_n\}$. Given this event, Chernoff bounds give us that the probability that none of the (at least $n^{1/2}/2$) unnegated variables of the clause are among the variables assigned 1, is at most $2^{-\Omega(n^{1/2})}$.

Since there are at most $2^{o(n^{1/2})}$ terms of size more than $n^{1/2}$ and since each is killed by ρ with probability at least $1-2^{-\Omega(n^{1/2})}$, the probability that *all* clauses of the CNF have size at most $n^{1/2}$ in the CNF for f_ρ derived from the CNF for f, is at least $1-2^{o(n^{1/2})}2^{-\Omega(n^{1/2})} = 1-2^{-\Omega(n^{1/2})}$.

Now apply a second random restriction σ on the remaining free variables of f_ρ leaving $n^{1/2}$ variables free. Conditioned by the event that all the clauses of f_ρ do have size at most $n^{1/2}$, the switching lemma implies that the probability that the function $f_{\sigma\circ\rho}$ does not have a decision tree of depth $n^{1/2}/2$ is smaller than $2^{-\Omega(n^{1/2})}$. Thus, combining the two random restrictions into one, we have proved that a random restriction to the variables of f leaving $n^{1/2}$ variables free, has a decision tree of depth $n^{1/2}/2$ with probability at least $1-2^{-\Omega(n^{1/2})}$.

We can fix the positions of the free variables while preserving this probability. Thus, we have proved the following statement: There is a set S of $n^{1/2}$ variables among the variables of f so that if the remaining variables are fixed at random, the resulting function has a DNF of size at most $2^{n^{1/2}/2}$ with probability at least $1-2^{-\Omega(n^{1/2})}$.

Now we make a DNF for the original function as follows: For each assignment of the variables not in S for which it is *not* the case that the resulting subfunction of f on the variables of S has a DNF of size at most $2^{n^{1/2}/2}$ (a bad assignment), we make the trivial DNF. For each assignment for S which it *is* the case that the resulting subfunction of f on the variables of S has a DNF of size at most $2^{n^{1/2}/2}$

(a good assignment), we take this DNF. We combine the DNFs we obtain for the various assignments of S into a single DNF for f in the obvious way.

The contribution to the total size by the good assignments to the total size is at most $2^{n-n^{1/2}}2^{n^{1/2}/2}$, i.e. $2^{n-\Omega(n^{1/2})}$. The contribution of the bad settings is at most $2^{-\Omega(n^{1/2})}2^{n-n^{1/2}}2^{n^{1/2}}$, i.e. $2^{n-\Omega(n^{1/2})}$. Thus, the total size is at most $2^{n-\Omega(n^{1/2})}$. $\square$

We now make a more refined argument giving the optimal bound for polynomial-size CNFs.

Theorem 7 *Let $f : \{0,1\}^n \to \{0,1\}$ be a function with a CNF of size $n^{O(1)}$. Then f has a DNF of size at most $2^{n-\Omega(n/\log n)}$.*

Proof

Given a CNF for f with $m \leq n^c$ clauses for some constant c. Consider first applying a random restriction ρ leaving $n/2$ variables free. The key to the proof is to consider the following event E:

E: *After applying ρ, there is a set S of at most $n/10$ free variables so that every clause of the CNF that is not killed by ρ has at most $100c\log n$ free variables outside of S.*

The utility of considering E is this: If the event E occurs, we are in a very good position for finding a small DNF for f_ρ since no matter how the variables of S are assigned values, the CNF for f_ρ has clause size at most $100c\log n$ which is very good for applying the switching lemma.

We now argue that E actually occurs with probability $1 - 2^{-\Omega(n)}$. Suppose E does not occur. Then we shall argue that we can find a set T of at most $n/(100c\log n)$ clauses in the CNF containing in total at least $n/10$ variables so that none of these clauses are killed by our restriction. Indeed, if E does not occur we can find such a set by first taking an arbitrary clause of size at least $100c\log n$ not killed by ρ. Such a set must exist as otherwise the empty set would work as T. If this clause has at least $n/10$ variables we are done. Otherwise, the set of variables in the clause has size less than $n/10$, so since E does not occur, there must be another clause with at least $100c\log n$ "new variables" (otherwise the clause we chose would work as T and E would occur). We add this clause to our set and consider the union of the clauses added so far. If this union has size at least $n/10$ we are done, otherwise there must be a clause not killed by ρ with at least $100c\log n$ "new variables", etc. As E does not occur, we can continue the process until the set found has the desired property. Since in each step $100c\log n$ new variables are found, the process can only continue for at most $n/(100c\log n)$ steps and hence we will find a set T of the desired size.

Given the set T of clauses, let $V(T)$ denote the set of variables appearing in the clauses of T. For each variable we make a note about whether it appears negated or unnegated (or both), so we may also think of $V(T)$ as a set of literals. We estimate the probability of E not occuring:

$$
\begin{aligned}
& \Pr[E \text{ does not occur }] \\
\leq\ & \Pr[\text{ Some set } T \text{ of at most } n/(100c\log n) \text{ clauses not killed by } \rho \\
& \quad \text{has } |V(T)| \geq n/10] \\
\leq\ & \sum_{T, |T|<n/(100c\log n), |V(T)| \geq n/10} \Pr[\text{ No literal in } V(T) \text{ is assigned 1 by } \rho]
\end{aligned}
$$

Let us upper bound the probability that for a fixed set of literals $V(T)$ of size at least $n/10$, no literal in $V(T)$ is assigned 1 by ρ. Recall that ρ assigns random values to $n/2$ randomly chosen variables. We may choose such a restriction by considering each variable in $\{x_1, x_2, \ldots, x_n\}$ at a time, choosing whether to fix it and in case we decide to fix it, flipping a coin about which value to give it. To obtain the right probability distribution, the probability by which we decide whether to fix it should be $(n/2 - k)/n$ where k is the number of variables we already fixed. Suppose we follow this procedure, but consider the variables of $V(T)$ before the other variables. As $(n/2 - n/10)/(2n) > 1/8$, we have that the probability that none of the literals in $V(T)$ is assigned 1 is less than $(7/8)^{n/10}$.

Thus, we may continue our sequence of inequalities. We use the notation $\binom{n}{<k}$ for the sum of all $\binom{n}{m}, m < k$.

$$
\begin{aligned}
& \sum_{T, |T|<n/(100c\log n), |V(T)| \geq n/10} \Pr[\text{ No literal in } V(T) \text{ is assigned 1 by } \rho] \\
<\ & \binom{n^c}{< n/(100c\log n)} (7/8)^{n/10} \\
\leq\ & 2^{n/100} (7/8)^{n/10} \\
\leq\ & 2^{-\Omega(n)}.
\end{aligned}
$$

Fix the positions of the free variables of ρ to a fixed set S while preserving this probability. For every assignment of the variables not in S, we shall now make a DNF for f and combine them as in the previous proof.

For assigments to the variables not in S defining a restriction ρ where E does not occur, we make the naive DNF for f_ρ. From our upper bound for the probability that E does not occur, the total number of terms to the final DNF from these cases is at most $2^{n-\Omega(n)}$ so we can ignore these cases for the purpose of the DNF size bound we have to prove.

For assignments to the variables not in S corresponing to restrictions ρ where E *does* occur, we find the set T guaranteed to exist by the definition of E. No matter how the variables of T are assigned values, the CNF for f_ρ has clause size at most $100c\log n$. For every assignment π to the variables of T, we consider the function $f_{\pi\circ\rho}$. Now take a random restriction σ of $f_{\pi\circ\rho}$ leaving $L = n/(1000c\log n)$ variables free. By the switching lemma, the probability that $f_{\sigma\circ\pi\circ\rho}$ does not have a decision tree of depth $L/2$ is at most 2^{-L}. We fix the free variables of σ while preserving this probability and make a DNF for $f_{\pi\circ\rho}$ as in the previous proof by combining the good and bad cases of the switching lemma. The total contribution of clauses to the final DNF for f from these cases is at most $O(2^{n-\Omega(L)})$, as in the previous proof.

Thus, the total size of the DNF we construct for f is at most $2^{n-\Omega(n/\log n)}$, as desired. $\square$

4 Conclusion and open problems

We have proved tight bounds for the largest possible blow up in size when converting a polynomial-size CNF to an equivalent optimal-size DNF. The following questions remain open.

- What is the largest possible blow up in size when converting a polynomial-size monotone CNF to an equivalent optimal-size monotone DNF? Equivalently, what is the largest possible number of distinct minimal vertex covers for a hypergraph with n vertices and $n^{O(1)}$ edges? We have given the upper bound $2^{n-\Omega(n/\log n)}$ and the lower bound $2^{n-O(n \log\log n/\log n)}$. Getting tight bounds seems challenging.
- We have in this paper focused on the case of polynomial-size CNFs. We can of course ask about the blow up in size starting from a CNF of any particular size and our bounds generalize to other sizes in the obvious way. In particular, the case of a *linear*-size CNF seems interesting, i.e., we consider starting with a CNF of size $m = cn$ for some constant c. In Section 2, we give examples of functions with a CNF of size cn for which the optimal-size DNF is $\approx 2^{\alpha_c n}$ for some constant α_c depending on c with $\alpha_c \to 1$ as $c \to \infty$. Our upper bound of Section 3 can be easily modified to show: If f has a CNF of size cn, then f has a DNF of size at most $2^{\beta_c n}$ where β_c is a constant depending on c. The constant β_c that can be derived from the current proof is bigger than α_c and very close to 1 even for small c. It would be interesting to get the two multiplicative constants in these exponents tight for fixed c.

References

[1] Beame, P. (1994). A switching lemma primer. Technical Report UW-CSE-95-07-01, Department of Computer Science and Engineering, University of Washington, November 1994. Available online at `www.cs.washington.edu/homes/beame/`.

[2] Bollig, B. and Wegener, I. (1998). A very simple function that requires exponential size read-once branching programs. Information Processing Letters 66, 53–57.

[3] Dantsin, E., Goerdt, A., Hirsch, E.A., and Schöning, U. (2000). Deterministic algorithms for k-SAT based on covering codes and local search. Pro-

ceedings of the 27th International Colloqium on Automata, Languages and Programming. Springer. LNCS 1853, 236–247.

[4] Dantsin, E., Goerdt, A., Hirsch, E.A., Kannan, R., Kleinberg, J., Papadimitriou, C., Raghavan, P., and Schöning, U. (2003). A deterministic $(2-2/(k+1))^n$ algorithm for k-SAT based on local search. Theoretical Computer Science, to appear.

[5] Håstad, J. (1989). Almost optimal lower bounds for small depth circuits. In: Micali, S. (Ed.): *Randomness and Computation.* Advances in Computing Research 5, 143–170. JAI Press.

[6] Hofmeister, T., Schöning, U., Schuler, R., and Watanabe, O. (2002). A probabilistic 3-SAT algorithm further improved. Proceedings of STACS'2002. LNCS 2285, 192–202.

[7] Katajainen, J. and Madsen, J.N. (2002). Performance tuning an algorithm for compressing relational tables. Proceedings of SWAT 2002. LNCS 2368, 398–407.

[8] Monien, B. and Speckenmeyer, E. (1985). Solving satisfiability in less than 2^n steps. Discrete Applied Mathematics 10, 287–295.

[9] Paturi, R., Pudlàk, P., Saks, M.E., and Zane, F. (1998). An improved exponential-time algorithm for k-SAT. Proceedings of the 39th IEEE Symposium on the Foundations of Computer Science 1998, 628–637.

[10] Razborov, A. and Rudich, S. (1997). Natural proofs. Journal of Computer and System Sciences 55, 24–35.

[11] Schöning, U. (2002). A probabilistic algorithm for k-SAT based on limited local search and restart. Algorithmica 32, 615–623, 2002.

[12] Wegener, I. (1987). *The Complexity of Boolean Functions.* Wiley. Free available via http://ls2-www.cs.uni-dortmund.de/∼wegener.

[13] Wegener, I. (2000). *Branching Programs and Binary Decision Diagrams – Theory and Applications.* SIAM Monographs on Discrete Mathematics and Applications.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'