# Chicago Journal of Theoretical Computer Science

*The MIT Press*

Volume 1999, Article 11


*Satisfiability Coding Lemma*


ISSN 1073–0486. MIT Press Journals, Five Cambridge Center, Cambridge, MA 02142-1493 USA; (617)253-2889; *journals-orders@mit.edu, journals-info @mit.edu*. Published one article at a time in LaTeX source form on the Internet. Pagination varies from copy to copy. For more information and other articles see

- *http://mitpress.mit.edu/CJTCS/*
- *http://www.cs.uchicago.edu/publications/cjtcs/*
- *ftp://mitpress.mit.edu/pub/CJTCS*
- *ftp://cs.uchicago.edu/pub/publications/cjtcs*

The *Chicago Journal of Theoretical Computer Science* is abstracted or indexed in *Research Alert®, SciSearch®, Current Contents®/Engineering Computing & Technology*, and *CompuMath Citation Index®*.
©1999 The Massachusetts Institute of Technology. Subscribers are licensed to use journal articles in a variety of ways, limited only as required to ensure fair attribution to authors and the journal, and to prohibit use in a competing commercial product. See the journal's World Wide Web site for further details. Address inquiries to the Subsidiary Rights Manager, MIT Press Journals; (617)253-2864; *journals-rights@mit.edu*.
The *Chicago Journal of Theoretical Computer Science* is a peer-reviewed scholarly journal in theoretical computer science. The journal is committed to providing a forum for significant results on theoretical aspects of all topics in computer science.

*Editor-in-Chief:* Janos Simon

*Consulting Editors:* Joseph Halpern, Eric Allender, Raimund Seidel

*Editors:*

| | | |
|---|---|---|
| Martin Abadi | Greg Frederickson | John Mitchell |
| Pankaj Agarwal | Andrew Goldberg | Ketan Mulmuley |
| Eric Allender | Georg Gottlob | Gil Neiger |
| Tetsuo Asano | Vassos Hadzilacos | David Peleg |
| Laszló Babai | Juris Hartmanis | Andrew Pitts |
| Eric Bach | Maurice Herlihy | James Royer |
| Stephen Brookes | Ted Herman | Alan Selman |
| Jin-Yi Cai | Stephen Homer | Nir Shavit |
| Anne Condon | Neil Immerman | Eva Tardos |
| Cynthia Dwork | Howard Karloff | Sam Toueg |
| David Eppstein | Philip Klein | Moshe Vardi |
| Ronald Fagin | Phokion Kolaitis | Jennifer Welch |
| Lance Fortnow | Stephen Mahaney | Pierre Wolper |
| Steven Fortune | Michael Merritt | |

*Managing Editor:* Michael J. O'Donnell

*Electronic Mail:* *chicago-journal@cs.uchicago.edu*

# Satisfiability Coding Lemma

Ramamohan Paturi Pavel Pudlák Francis Zane

31 December 1999

**Abstract**

In this paper, we prove a lemma that shows how to encode satisfying solutions of a $k$-CNF (boolean formulae in conjunctive normal form with at most $k$ literals per clause) succinctly. Using this lemma, which we call the *satisfiability coding lemma*, we prove tight lower bounds on depth-3 circuits and improved upper bounds for the $k$-SAT problem. We show an $\Omega(n^{1/4}2^{\sqrt{n}})$ lower bound on the size of depth-3 circuits of AND, OR, NOT gates computing the parity function. This bound is tight up to a constant factor. We also present and analyze two simple algorithms for finding satisfying assignments of $k$-CNFs. The first is a randomized algorithm which, with probability approaching 1, finds a satisfying assignment of a satisfiable $k$-CNF formula $F$ in time $O(n^2|F|2^{n-n/k})$. The second algorithm is deterministic, and its running time approaches $2^{n-n/2k}$ for large $n$ and $k$. The randomized algorithm improves on previous algorithms for $k > 3$, though subsequent algorithms improve on it; the deterministic algorithm is the best known deterministic algorithm for $k > 4$.

## 1 Introduction

The problem of finding a satisfying assignment of a boolean formula in $k$-CNF (conjunctive normal form with at most $k$ literals per clause) has been long studied, and the corresponding decision problem for $k \geq 3$ was one of the first problems shown to be NP-complete [Coo71, Lev73]. The problem of proving lower bounds on the size of constant-depth (in particular, depth-3) circuits of unbounded fan-in AND and OR gates with positive and negative literals as inputs has also received considerable attention [BS90, HJP93, PSZ97]. In this paper, we present a structural property, the *satisfiability coding lemma*, of the set of satisfying assignments of $k$-CNF and apply this property to provide some solutions to both these problems.

There has been considerable progress in the study of the computational limitations of polynomial size and bounded-depth circuits of unbounded fan-in AND and OR gates ($AC^0$) with literals as inputs. The techniques used in [Hås86, Raz86, Smo87] for showing lower bounds on the size of bounded depth circuits establish that functions computed by such small size circuits have some useful (for lower bounds) property. For example, it has been shown that such functions are constant on a sufficiently large-dimensional subcube and also can be well approximated by low-degree polynomials, over either the reals or an appropriately chosen finite field. Thus, any function that does not have these properties cannot be computed by an $AC^0$ circuit. Frequently, the function considered is parity (that is, the output is 1 if the number of true input variables is, say, even) because it is extremal with respect to the properties mentioned above; the parity function is not constant, even on any one-dimensional subcube, and a good approximation of parity in the field of reals or in any field except $GF(2)$ requires almost linear degree. However, such useful techniques still have not resulted in the determination of the exact complexity of computing the parity function. More specifically, in the case of depth-3 circuits, the best-known general technique (the switching lemma [Hås86]) yields a lower bound of $\Omega(2^{c\sqrt{n}})$ for $c < 1/8$. Håstad, Jukna and Pudlák [HJP93] improved this lower bound to $\Omega(2^{0.618\sqrt{n}})$ by using a top-down argument. On the other hand, the best-known upper bound for computing parity is $O(n^{1/4}2^{\sqrt{n}})$, which is obtained by partitioning the variables into groups of size $\sqrt{n} - (1/4)\log n$, computing the parity of each group of variables, and then computing the parity of those results. In this paper, using the satisfiability coding lemma, we show that computing parity using depth-3 circuits requires $\Omega(n^{1/4}2^{\sqrt{n}})$ gates matching the upper bound.

As for the problem of finding a satisfying assignment of a $k$-CNF, there exists in the literature a variety of algorithms and analyses. We focus our attention on those algorithms that have a provable worst-case running time better than the $2^n$ steps required for an exhaustive search, where $n$ is the number of variables. A straightforward improvement is obtained by selecting a smallest-length clause and branching on all but one of the assignments to the variables in that clause, omitting the one assignment that makes the clause false. Monien and Speckenmeyer [MS85] analyze this algorithm carefully by arguing that such assignments either produce a clause of shorter length or are autark. An assignment to a set of variables is autark if all the clauses that contain the variables are satisfied by the assignment. Their analysis gives a worst-case running time $O(|F|2^{n\log(\alpha_k)})$ where $F$ is a $k$-CNF and $\alpha_k$ is the largest real root of the equation $\alpha_k^k - 2\alpha_k^{k-1} + 1 = 0$. For example, this gives a bound of $O(2^{0.6943n})$ for 3-CNF and an $O(2^{0.879n})$ bound for 4-CNF. For the case $k = 3$, better algorithms are established: Schiermeyer [Sch93] employs more involved heuristics to obtain an algorithm with $O(2^{0.659n})$ worst-case running time for finding a satisfying assignment for 3-CNF. This is improved to $O(2^{0.652n})$ by Zhang [Zha96]. In a more recent work [Sch96], Schiermeyer claims an improved algorithm with worst-case running time $O(2^{0.582n})$ for 3-CNF.

In this paper, we present some obvious and simple algorithms for finding a satisfying solution of a $k$-CNF and analyze their worst-case running time using the satisfiability coding lemma. The first of these algorithms is probabilistic and, with high probability, finds a satisfying assignment of a satisfiable $k$-CNF in time $O(|F|2^{n(1-1/k)})$. We also present a deterministic algorithm whose worst-case running time approaches $O(|F|2^{n(1-1/2k)})$ for large values of $n$ and $k$. Our randomized algorithm is the best-known algorithm for $k \geq 4$. Our deterministic algorithm is better than the known deterministic algorithms for $k \geq 5$. Since the submission of this paper, two improved algorithms have been presented: The algorithm in in [PPSZ98] builds on the techniques of this paper, while another by [Sch99] uses substantially different ideas. For 3-SAT, these algorithms achieve running times of $2^{0.446n}$ and $2^{0.416n}$, respectively.

Our main technique, the satisfiability coding lemma, is motivated by a simple question: How many isolated solutions can a $k$-CNF have? An isolated solution is a satisfying assignment whose distance-1 neighbors in the boolean cube (assignments which differ in exactly one variable) are not satisfying assignments. In other words, if any bit of an isolated solution is flipped, the formula is no longer satisfied. We also say that an input accepted by a circuit is isolated if any input at distance 1 from it is not accepted by the circuit. It is straightforward to see that a good upper bound on the number of isolated solutions would be helpful in proving tight lower bounds for computing parity using depth-3 circuits. If parity is computed by a small-size depth-3 circuit with an OR gate at the top, then one of its depth-2 subcircuits (a CNF) must have a large number of isolated solutions. Håstad's switching lemma implies an $O(2^{(1-c/k)n})$ bound (with $c < 1$) on the number of isolated solutions of a $k$-CNF. In fact, our satisfiability coding lemma is inspired by Razborov's proof [Raz95] of a variant of Håstad's switching lemma. Using the satisfiability coding lemma, we prove that a $k$-CNF can have at most $2^{(1-1/k)n}$ isolated solutions, which is in fact the best possible bound. In addition, by carefully counting the contributions of bottom-level gates with larger fan-ins, we prove that computing parity by depth-3 circuits requires $\Omega(n^{1/4}2^{\sqrt{n}})$ gates, obtaining a lower bound that matches the upper bound extremely closely.

The satisfiability coding lemma essentially says that isolated solutions of $k$-CNFs have *short descriptions.* More precisely, we prove that the set of isolated satisfying assignments of a $k$-CNF can be encoded with an average message length of $(n - n/k)$ bits. This lemma is useful not only in obtaining upper bounds on the number of isolated satisfying assignments of a $k$-CNF but also in efficiently finding an isolated satisfying assignment if one exists. If an isolated satisfying assignment exists, then it is sufficient to search the smaller space of short descriptions for one that encodes a satisfying assignment. To handle the general case, we generalize the concept of the isolated satisfying assignment to include nearly isolated satisfying assignments, and we show that such solutions have short descriptions as well. From this, we can show that any satisfiable $k$-CNF has either a nearly isolated solution or many satisfying assignments, and thus in each case we can find a satisfying solution quickly, either by searching through the space of short descriptions or by randomly guessing a solution.

The remainder of the paper is organized as follows: In Section 2, we prove the satisfiability coding lemma and its corollaries. In Section 3, we prove the tight lower bound for computing parity using depth-3 circuits of AND and OR gates with literals as inputs. In Section 4, we present our algorithms for finding a satisfying assignment of a $k$-CNF and analyze their running time.

# 2 Satisfiability coding lemma

We introduce some notation. A boolean formula $F = \bigwedge_{i=1}^{m} C_i$ is a $k$-CNF if each clause $C_i$ is a disjunction of at most $k$ literals. For $x$ in $S \subseteq \{0,1\}^n$, we say that $x$ is an isolated point of $S$ in the direction $i$ if flipping the $i$th bit of $x$ produces a point not in $S$. We say that $x$ is a *$j$-isolated* point of $S$ if it has exactly $(n-j)$ neighbors in $S$. We sometimes use the alternative notation $I(x)$ to denote the number of neighbors of $x \in S$ that are not in $S$. If $x$ is $n$-isolated, we simply call it an isolated point of $S$. We say that

$x$ is an isolated solution of a formula $F$ in the direction $i$ if $x$ satisfies the formula and is an isolated point of the set of all satisfying assignments of $F$ in the direction $i$. Other notions of isolation are extended similarly to general formulae and circuits.

Let $F$ be a $k$-CNF. The key observation is that if $x$ is an isolated solution of $F$ in the direction $i$, then there exists a clause $C_{(x,i)}$ such that exactly one of its literals is true under the assignment $x$ and that the true literal corresponds to the variable $i$. Otherwise, flipping the $i$th bit of $x$ would produce an assignment which still satisfied all the clauses of $F$. The clause $C_{(x,i)}$ is called *critical* for the variable $i$ at the solution $x$. Since a critical clause has only one true literal, a clause cannot be critical for two different variables at the same solution. If $x$ is an isolated solution of $F$, then there exist $n$ distinct critical clauses at $x$, one for each direction.

We make use of the existence of critical clauses for $j$-isolated solutions in obtaining short descriptions. Let $\sigma$ be a permutation of the set $\{1, 2, \ldots, n\}$ of variables. We define an encoding function $\Phi_\sigma$ to encode the satisfying assignments of the $k$-CNF $F$. Let $x \in \{0,1\}^n$ be a satisfying solution of $F$. Permute the bits of $x$ according to $\sigma$. For each $i$, delete the $i$th bit of the permuted string if there is a critical clause $C_{(x,\sigma(i))}$ for the variable $\sigma(i)$ at $x$ such that the variable $\sigma(i)$ occurs after all the other variables in the critical clause $C_{(x,\sigma(i))}$ according to the ordering $\sigma$. $\Phi_\sigma(x)$ is the resulting string. Observe that the sequence $\Phi_\sigma(x)$ of bits corresponds to the sequence of values of a subset of variables under the satisfying assignment $x$ presented according to the ordering $\sigma$.

Given $y = \Phi_\sigma(x)$, we uncover the bits of $x$ one at a time in the order given by $\sigma$ using the following decoding algorithm:

$F_1 := F$
**for** $i = 1, \ldots, n$
  **if** $F_i$ has a clause of length one consisting of the variable $\sigma(i)$,
    **then** we set the variable $\sigma(i)$ to make the clause true.
    **else** we set the variable $\sigma(i)$ to be the next unused bit of $y$.
  Let $F_{i+1}$ be the formula obtained by substituting the value of the variable $\sigma(i)$ in $F_i$.

We first show the correctness of the algorithm.

**Lemma 1** *If $x$ is a satisfying assignment of the formula $F$ and $y = \Phi_\sigma(x)$ is given as input to the decoding algorithm, then, $\forall i, 1 \leq i \leq n$, the $i$th* *bit uncovered by the algorithm is the value of variable $\sigma(i)$ in the satisfying assignment $x$.*

**Proof** For $j \geq 1$, assume that for $1 \leq i \leq j$, the $i$th bit uncovered by the decoding algorithm is the value of the variable $\sigma(i)$ in the assignment $x$. Also assume that at the beginning of the $i$th iteration of the algorithm, the unused bits of $y$ correspond to variables whose ranks, according to $\sigma$, are greater than or equal to $i$. We prove that the $(j+1)$st bit uncovered by the algorithm is indeed the value of the variable $\sigma(j+1)$ in the assignment $x$. We also prove that at the end of the $(j+1)$st iteration the unused bits of $y$ correspond to variables whose ranks according to $\sigma$ are higher than $(j+1)$.

Assume that the condition in the **if** clause is true. Then the value of the variable $\sigma(j+1)$ is forced, since $F_{j+1}$ has a length-1 clause in the variable $\sigma(j+1)$. By induction hypothesis, the partial assignment to the variables $\sigma(i)$ for $1 \leq i \leq j$ can be extended to the satisfying assignment $x$. Thus the bit assigned by the algorithm must coincide with the value of the variable $\sigma(j+1)$ in the assignment $x$. Furthermore, it must be the case that the bit corresponding to the variable $\sigma(j+1)$ is deleted from the assignment $x$ by the encoding algorithm in producing $y$. This is due to the fact that the clause in the original formula $F$ that gave rise to the length-1 clause in the variable $\sigma(j+1)$ in the formula $F_{j+1}$ is a critical clause for the variable $\sigma(j+1)$ and that all other variables in that clause occur before the variable $\sigma(j+1)$ in the ordering $\sigma$. Therefore, from induction hypothesis, we can conclude that all the unused bits of $y$ must correspond to variables whose ranks are higher than $(j+1)$ in the ordering $\sigma$.

If, on the other hand, $F_{j+1}$ has no length-1 clause in the variable $\sigma(j+1)$, then there is no critical clause for the variable $\sigma(j+1)$ at $x$ such that the variable $\sigma(j+1)$ has the highest rank with respect to $\sigma$ among the variables that appear in the critical clause. This implies that the bit corresponding to the variable $\sigma(j+1)$ is not deleted from the assignment $x$ to produce the string $y$ by $\Phi_\sigma$. Since all the unused bits of $y$ at the beginning of the $(j+1)$st iteration of the decoding algorithm correspond to variables whose ranks are greater than or equal to $(j+1)$, and since the bits of $y$ correspond to the values of the variables ordered by $\sigma$, the next unused bit in $y$ must correspond to the variable $\sigma(j+1)$. Since this bit is assigned to the variable $\sigma(j+1)$, all the remaining unused bits correspond to variables whose rank according to $\sigma$ is higher than $(j+1)$. This completes the proof of the lemma. $\square$

We next prove the satisfiability coding lemma and its corollaries.

**Lemma 2 (Satisfiability coding lemma)** *If $x$ is a $j$-isolated satisfying assignment of a $k$-CNF $F$, then its average (over all permutations $\sigma$) description length under the encoding $\Phi_\sigma$ is at most $n - j/k$.*

**Proof** Since $x$ is $j$-isolated, it has $j$ variables with critical clauses. Let $\sigma$ be a random permutation of the variables. For each variable with a critical clause at $x$, the probability that the variable appears as the last variable among all the variables in its critical clause is at least $1/k$, since no clause of $F$ has more than $k$ literals. Hence the corresponding bit in $x$ is deleted in the encoding $\Phi_\sigma$ with probability at least $1/k$. Hence, the expected number of bits deleted in the encoding of $x$ is at least $j/k$, which yields a description for $x$ of length at most $n - j/k$. □

We now prove an upper bound on the number of isolated solutions of a $k$-CNF. We need the following fact regarding the average length of an encoding.

**Fact 3** *If $\Phi : S \to \{0,1\}^*$ is a prefix-free encoding (one-to-one function) with average code length $l$, then $|S| \leq 2^l$.*

**Proof** Let $l_x$ denote the length of $\Phi(x)$ for $x \in S$. Then $l = \sum_{x\in S} l_x/|S|$. Since $\Phi$ is one-to-one and prefix-free, we have that $\sum_{x\in S} 2^{-l_x} \leq 1$. Thus,

$$\begin{aligned} l - \log|S| &= \sum_{x\in S} \frac{1}{|S|}(l_x - \log|S|) = -\sum_{x\in S}\frac{1}{|S|}(\log 2^{-l_x} + \log|S|) \\ &= -\sum_{x\in S}\frac{1}{|S|}\log(|S|2^{-l_x}) \geq -\log(\sum_{x\in S} 2^{-l_x}) \geq 0. \end{aligned}$$

The penultimate inequality follows from the concavity of the logarithm function. Hence, $|S| \leq 2^l$. □

**Lemma 4** *Any $k$-CNF $F$ can accept at most $2^{n-n/k}$ isolated solutions.*

**Proof** By the satisfiability coding lemma, the average description length (under the encoding $\Phi$) of an isolated solution (that is, an $n$-isolated solution) of $F$ is at most $n - n/k$. This is also true when the average is taken over all isolated solutions and all permutations. Hence, there exists a permutation $\sigma$ such that the average description length under the coding $\Phi_\sigma$ is at most $n - n/k$. Observe that the proof of Lemma 1 shows that the set encodings produced by $\Phi_\sigma$ is prefix-free. Hence, from Fact 3, the number of isolated solutions cannot exceed $2^{n-n/k}$. □

This bound is indeed the best possible. Let $n$ be a multiple of $k$. Group the variables into $n/k$ disjoint groups of $k$ variables each. Let $F_i$ be the $k$-CNF accepting the parity function of the $k$ variables in group $i$. Consider the $k$-CNF obtained by taking the conjunction of the $F_i$. All the satisfying assignments of this $k$-CNF have the same parity and thus are isolated. Moreover, the $k$-CNF has exactly $2^{n-n/k}$ satisfying assignments.

# 3 Lower bound for depth-3 circuits

In this section, we prove a tight lower bound on the number of gates required by a depth-3 circuit to compute parity. We introduce some additional notation:

Let $x$ be an isolated solution of a CNF $F$. For each $i$, fix a shortest critical clause $C_{(x,i)}$ for the variable $i$ at $x$. Note that all these clauses are distinct. We define the length of a clause as the number of literals it contains. Let $N_l(x)$ be the number of critical clauses at $x$ of length $l$. Observe that $\sum_{l=1}^{n} N_l(x) = n$. To account for the contribution of clauses of various lengths, we define the weight, $w(x)$, of $x$ as $w(x) = \sum_{i=1}^{n} 1/|C_{(x,i)}| = \sum_{l=1}^{n} N_l(x)/l$. We show that, as a consequence of the satisfiability coding lemma, the number of isolated solutions of $F$ with weight greater than or equal to $\mu$ is at most $2^{n-\mu}$.

**Lemma 5** *If $F$ is a CNF, then the number of its isolated solutions with weight greater than or equal to $\mu$ is at most $2^{n-\mu}$.*

**Proof** We show that the average description length of an isolated solution of weight $\mu$ or greater is at most $n-\mu$ under the encoding $\Phi$. Let $x$ be an isolated solution of weight $w(x) \geq \mu$. Since the bit in $x$ corresponding to variable $i$ is deleted with probability at least $1/|C_{(x,i)}|$ in computing the encoding $\Phi_\sigma$ for a random $\sigma$, the expected number of bits deleted is at least $\sum_{i=1}^{n} 1/|C_{(x,i)}| \geq \mu$. Hence, there exists a permutation $\sigma$ such that the average (over all isolated solutions of weight greater than or equal to $\mu$) of the description lengths under the encoding $\Phi_\sigma$ is at most $n - \mu$. Therefore, the number of isolated solutions is at most $2^{n-\mu}$. □

We are now ready to prove our lower bound.

**Theorem 6** $\Omega(n^{1/4}2^{\sqrt{n}})$ *gates are required for any depth-3 circuit computing parity.*

**Proof** We only consider $\Sigma_3$ circuits, circuits that can be expressed as an OR of CNFs. Since the complement of a parity function is also a parity function, the proof applies to $\Pi_3$ circuits as well. Let $S$ be the set of inputs accepted by the circuit. By definition, $S$ is the set of inputs with an even number of 1's and $|S| = 2^{n-1}$. Since the top gate is an OR, for each $x \in S$, there exists a CNF (a depth-2 subcircuit), say, $F_x$, that accepts $x$. Moreover, $x$ is an isolated solution of $F_x$. We define the weight of $x$ with respect to the CNF $F_x$. We classify the inputs in $S$ based on their weight. Let $\mu = \sqrt{n}+(\log n)/4$. Let $S_1$ be the set of inputs in $S$ whose weight is greater than or equal to $\mu$. Let $S_2$ be the set of inputs in $S$ whose weight is less than $\mu$. Observe that $|S_1|+|S_2| = 2^{n-1}$. Since no CNF can accept more than $2^{n-\mu}$ isolated solutions whose weight is at least $\mu$, we get that the number of CNFs in the circuit is at least $|S_1|2^{\mu-n}$.

We now argue that many clauses are needed to accept low-weight isolated solutions. Since a clause of length $l$ can only be critical for at most $l2^{n-l}$ pairs $(x, i)$ of solution $x$ and direction $i$, there must be at least

$$\sum_{l=1}^{n} \sum_{x \in S_2} N_l(x)/(l2^{n-l}) = \sum_{x \in S_2} \sum_{l=1}^{n} N_l(x)/(l2^{n-l}) = \sum_{x \in S_2} n2^{-n} \sum_{l=1}^{n} \frac{N_l(x)}{n} \frac{2^l}{l} \quad (1)$$

clauses (that is, level-1 OR gates) to account for all the $|S_2|$ isolated solutions of weight less than $\mu$. Let $T_x = \sum_{l=1}^{n}((N_l(x))/n)(2^l/l)$ denote the inner summation, and define $\delta_x = \sum_{l=1}^{n}((lN_l(x))/n)$. Now, $T_x \geq 2^{\delta_x}/\delta_x$ by the convexity of the function $2^l/l$ and the fact $\sum_{l=1}^{n} N_l(x) = n$. From the constraint $\sum_{l=1}^{n} N_l(x)/l = w(x) < \mu$, we obtain $n\delta_x = \sum_{l=1}^{n} lN_l(x) \geq n^2/\mu$ by applying the Cauchy-Schwarz-Buniakowski inequality. Since the function $2^l/l$ is monotone for $l \geq 2$, we get that the inner sum $T_x \geq \mu 2^{n/\mu}/n$ for sufficiently large $n$. Thus, the the number of level-1 OR gates must be at least $|S_2|\mu 2^{-n+n/\mu}$.

Thus the total number of gates is at least $|S_1|2^{\mu-n} + |S_2|\mu 2^{-n+n/\mu}$. Minimizing this expression subject to the constraint $|S_1| + |S_2| = 2^{n-1}$, we get the desired lower bound. $\square$

In some cases, Theorem 6 can be generalized to provide a lower bound in terms of the *average sensitivity* of a function. The sensitivity of $f$ at $x$, $s(f, x)$, is the number of distance-1 neighbors $y$ of $x$ such that $f(x) \neq f(y)$. The average sensitivity of $f$, $S(f)$, is the expected sensitivity of $f$ at a random assignment: $S(f) = (1/2^n) \sum_{x \in \{0,1\}^n} s(f, x)$. For example, the parity function has an average sensitivity $n$. Boppana [Bop97] proved a $2^{O(\sqrt{S(f)})}$ lower bound on the number of gates required to compute $f$ with a depth-3 AND-OR circuit with literals as inputs.

For balanced functions $f$ (that is, functions with $|f^{-1}(1)| = |f^{-1}(0)|$), the constant factor in the exponent can be made tight using techniques similar to those above. Unlike in the case of parity, such a function need not be equivalent to its complement, so we restrict our attention to depth-3 AND-OR circuits whose output gate is an OR, which we refer to as $\Sigma_3$ circuits.

**Theorem 7** *Let $f$ be a boolean function such that $|\{x|f(x) = 1\}| = |\{x|f(x) = 0\}|$. Any $\Sigma_3$ circuit requires $\Omega(2^{\sqrt{S(f)}}/n)$ gates to compute the boolean function $f$.*

**Proof** Let $A = \{x|f(x) = 1\}$. We make use of the following standard fact about sensitivity (see [Bop97]):

$$\sum_{x \in A} s(f, x) = 2^{n-1} S(f).$$

Since $f$ is balanced, this implies that the average sensitivity of the points in $A$ and the overall average sensitivity are the same.

We now claim that there must be some $B \subseteq A$ with $|B| \geq |A|/2n$ such that for all $x \in B, s(f, x) \geq S(f)$. This follows from the fact that $s(f, x)$ is an integer between 1 and $n$, since otherwise we have that

$$S(f) \leq \frac{1}{2n} n + (1 - 1/2n)(S(f) - 1) < S(f).$$

Note that for $x \in A$, the sensitivity $s(f, x)$ is simply the isolation $I(x)$ with respect to $A$. We now follow the proof of Theorem 6, taking $S = B$ and $\mu = \sqrt{S(f)}$. The inputs in $S_1$ are handled exactly as before, giving a lower bound size of $|S_1| 2^{\mu - n}$. For the inputs in $S_2$, we must account for the fact that, unlike in the case of parity, $x \in A$ need not be isolated with respect to all variables and, thus, may have less than $n$ critical clauses. Instead, we know that $\sum_{l=1}^{n} N_l(x) = s(f, x)$, and thus for any $x \in B$, $\delta_x \geq S(f)/\mu$, since for such $x$'s, $s(f, x) \geq S(f)$.

Because of this distinction, we replace the last expression in equation (1) by $\sum_{x \in S_2} S(f) 2^{-n} \sum_{l=1}^{n} ((N_l(x))/S(f))(2^l/l)$ and define

$$\delta_x = \sum_{l=1}^{n} \frac{l N_l(x)}{S(f)}.$$

By repeating the arguments above, we obtain that $S(f)\delta_x \geq S(f)^2/\mu$ and thus the number of level-1 OR gates must be at least $|S_2|2^{-n+S(f)/\mu}$.

The theorem follows by combining the bounds from the two cases and minimizing subject to $|S| = |S_1| + |S_2| \geq 2^{n-1}/2n$. □

# 4 k-SAT algorithm

The satisfiability coding lemma can also be used to find satisfying assignments of $k$-CNF formulae in less than $2^n$ steps. Suppose the $k$-CNF formula $F$ has some solution that is isolated or nearly isolated. By the satisfiability coding lemma, with respect to many permutations $\sigma$, such a solution has an encoding of short length. By searching this space of encodings for one that encodes a satisfying assignment, we would be assured of finding a satisfying assignment if one exists. If no solution is isolated or nearly isolated, we may not be able to guarantee the existence of such short encodings. However, in such a case, if there is any satisfying solution, there must be many of them; thus the chance of the randomly guessing one of them is higher. In this section, we present a randomized algorithm that (with probability approaching 1) finds a satisfying assignment of a $k$-CNF in $O(n^2|F|2^{n-n/k})$ steps, as well as a somewhat less efficient deterministic algorithm that uses similar techniques. These results are summarized in Table 1.

| | Previous (deterministic) | New (randomized) | New (deterministic) |
|---|---|---|---|
| $k=3$ | $2^{0.582n}$ [Sch96] | $2^{0.667n}$ | $2^{0.896n}$ |
| $k=4$ | $2^{0.879n}$ [MS85] | $2^{0.750n}$ | $2^{0.917n}$ |
| $k=5$ | $2^{0.947n}$ [MS85] | $2^{0.800n}$ | $2^{0.931n}$ |
| $k=6$ | $2^{0.975n}$ [MS85] | $2^{0.833n}$ | $2^{0.941n}$ |

Table 1:

## 4.1 Randomized algorithm

The randomized algorithm we analyze here is extremely simple:

**Algorithm A**

**repeat** $n^2 2^{n-n/k}$ times
  **while** there exists an unassigned variable
    select an unassigned variable $y$ at random
    **if** there is a clause of length 1 involving $y$ or $\bar{y}$
      **then** set $y$ to make that clause true
      **else** set $y$ to **true** or **false** at random
  **if** the formula is satisfied, **then** output the assignment

Before we analyze the probability that this algorithm succeeds in finding a satisfying assignment, we first establish a lemma that we need in the analysis. This lemma relates the density of a set and the isolation of the members of the set. While this lemma can also be proved by using the well-known edge isoperimetric inequality for the cube [Bol85], we include a simple, self-contained proof for completeness.

**Lemma 8** *Let $S \subseteq \{0,1\}^n$, $S$ nonempty, and for $x \in S$, define $I(x)$ as the number of distance-1 neighbors of $x$ that are not in $S$. Define value$(x) = 2^{I(x)-n}$. Then $\sum_{x \in S}$ value$(x) \geq 1$.*

**Proof** The proof is by induction on $n$. If $n = 0$, it is trivially true. If $n > 0$, consider the two subcubes $B_i^{n-1}$ $(i = 0, 1)$ generated by fixing the value of the last coordinate. Let $S_i = S \cap B_i^{n-1}$. If the $S_i$ are nonempty, the induction hypothesis guarantees that the sum of the values of elements in each $S_i$ is at least 1 when $S_i$ are considered as subsets of the $(n-1)$-dimensional cube. If one of the $S_i$ is empty, then the other one is nonempty, and moreover when it is viewed as a subset of the $n$-dimensional cube the value of $I(x)$ increases by one for each of its elements. Thus the sum of the values remains greater than or equal to 1. If both $S_i$ are nonempty, then the sum of the values of the elements in each $S_i$ is at least $1/2$ when $S_i$ are considered as subsets of the $n$-dimensional cube. Thus the total value is at least 1, as desired. $\square$

With this lemma, we now show that the algorithm described above finds satisfying assignments of $k$-CNFs quickly.

**Theorem 9** *Algorithm A runs in time $O(n^2|F|2^{n-n/k})$ and finds a satisfying assignment of a satisfiable $k$-CNF $F$ with probability approaching 1.*

**Proof** Suppose that $F$ is satisfiable, and that $x$ is a $j$-isolated solution of $F$ for some $j \in \{1, \ldots, n\}$. For each of the $j$ directions in which $x$ is isolated, fix a critical clause. We obtain a lower bound for the probability that $x$ is output by Algorithm A during an iteration of the repeat loop.

Consider one iteration of the **repeat** loop. Let $\sigma$ be the random permutation determined by the order in which variables are assigned in the **while** loop. Let $E_1$ be the event that for at least $j/k$ critical clauses, the critical variables occur last among the variables in the critical clause with respect to the random permutation $\sigma$. Let $E_2$ be the event that the values assigned to the variables in the **while** loop agree with the assignment $x$. We use the probability of the event $E_1 \wedge E_2$ as a lower bound on the probability that the algorithm outputs $x$.

Since $x$ is a $j$-isolated solution, the average number (over all permutations) of critical variables that appear last among the variables in their critical clauses is at least $j/k$. Since the maximum number of critical variables is $n$, it follows that for at least a $1/n$ fraction of permutations, the number of such critical variables is at least $j/k$. Thus the probability of $E_1$ is at least $1/n$.

Assuming $E_1$, we lower-bound the probability for $E_2$. It is clear that if the random assignments made in the **else** branch agree with the satisfying assignment $x$, then the assignments made in the **then** branch must agree as well. Since $E_1$ holds, the **else** branch is taken at most $n - j/k$ times. Thus the probability that $E_2$ holds given $E_1$ is at least $2^{-n+j/k}$, and the probability that the $j$-isolated satisfying assignment $x$ is output by the algorithm is at least $2^{-n+j/k}/n$.

The probability that the algorithm generates some satisfying solution can then be obtained by summing over all satisfying assignments. Let $S$ be the set of satisfying assignments of $F$. We have

$$\begin{aligned}
\sum_{x \in S} \Pr[x \text{ is output by the algorithm}] &\geq \sum_{x \in S} \frac{1}{n} 2^{-n+I(x)/k} \\
&= \frac{1}{n} 2^{-n+n/k} \sum_{x \in S} 2^{(-n+I(x))/k} \\
&\geq \frac{1}{n} 2^{-n+n/k} \sum_{x \in S} 2^{-n+I(x)} \\
&\geq \frac{1}{n} 2^{-n+n/k},
\end{aligned}$$

where the last inequality follows from Lemma 8. By repeating the **while** loop $n^2 2^{n-n/k}$ times, we find a satisfying assignment with probability approaching 1. $\square$

There is a simple $k$-CNF, on which the expected number of times the algorithm executes this loop is $2^{n-n/k}$. Experiments suggest, however, that on random $k$-CNF the algorithm runs much faster.

In order to obtain a deterministic algorithm, we first observe that the problem of finding a good ordering of variables requires only limited independence. Specifically, to conclude that the average number of bits saved in the encoding of a $j$-isolated solution of a $k$-CNF is at least $j/k$, it is sufficient that each of the $k$ variables in a clause is equally likely to occur last with respect to a random ordering. However, it is not clear how to come up with an efficient algorithm for selecting a satisfying assignment deterministically when there are a large number of satisfying assignments, none of which are sufficiently isolated. We now present a somewhat less efficient deterministic algorithm for finding a satisfying assignment.

We construct a small space $S$ of permutations of $\{1, 2, \ldots, n\}$ with the following property: For any set $Y$ of up to $k$ variables, any variable $y$ in $Y$, and for a randomly chosen permutation from $S$, the probability that $y$ appears last among the variables in $Y$ is at least $1/|Y| - 1/n$. The following construction of such a family was suggested to us by Russell Impagliazzo.

Let $G = \{1, 2, \ldots, m\}$, where $m$ is a prime power larger than $n^3$. Let $S'$ be a probability space over which $n$ $k$-wise independent random variables, each taking values in $G$, are defined. Using known techniques [ASE92], such a probability space $S'$ can be constructed such that $|S'| \leq O(n^{3k})$. Let $Y$ be a set of at most $k$ variables. If all the values assigned to the variables in $Y$ are distinct, then the assignment induces an ordering of the variables in $Y$. Assuming the variables in $Y$ take distinct values, $k$-wise independence guarantees that all orderings of the variables in $Y$ have the same probability. In particular, each variable in $Y$ occurs last among the variables in $Y$ with probability $1/|Y|$.

Let $S \subseteq S'$ correspond to the event that all the $n$ variables take distinct values. Each element of $S$ can be interpreted as a permutation of $\{1, 2, \ldots, n\}$, which is given by the ordering of the variables by their values. Since $k \geq 2$, it follows that the probability of $S$ is at least $1 - 1/n$. It also follows that over the space $S$, for any set $Y$ of at most $k$ variables and a variable in $Y$, the probability that the variable occurs last is at least $1/|Y| - 1/n$.

Thus $S$ has the desired property.

Our deterministic algorithm makes use of two ideas. First, observe that there is either a satisfying assignment having few 1's or any minimal solution having many 1's, where minimality is defined with respect to the number of 1's in the assignment. This dichotomy is useful, because a minimal solution must be isolated in all the directions where a variable has the value 1. The second observation is that by using permutations from the family $S$ rather than truly random permutations to order the variables, a $j$-isolated solution can still be encoded using at most $n-j/k+1$ bits. The following deterministic algorithm incorporates these ideas. Let the parameter $\varepsilon$ be such that $0 \leq \varepsilon \leq 1/2$ and $\varepsilon$ satisfies $(1-\varepsilon/k) = H(\varepsilon)$, where $H(x)$ is the binary entropy function.

**Algorithm B**
**for** all inputs $x$ with at most $\varepsilon n$ 1's
  **if** the formula is satisfied by $x$, **then** output $x$
**for** all permutations $\sigma$ in $S$,
  and for all strings $s$ of $n(1-\varepsilon/k)+1$ bits
  /* apply the decoding algorithm $\Phi_\sigma^{-1}$ to $s$ */
  **for** i=1 **to** n
    let $y$ be the $i$th variable according to $\sigma$
    **if** there is a clause of length 1 involving $y$ or $\bar{y}$
    **then** set $y$ to make that clause true
    **else** set $y$ equal to the next unused bit from $s$
  **if** the formula is satisfied, **then** output the assignment

**Theorem 10** *Algorithm B finds a satisfying assignment of a satisfiable k-CNF F in* $O(|F|n^{3k}2^{\varepsilon n})$ *time.*

**Proof** The first **for** loop of the algorithm checks whether any input with at most $\varepsilon n$ 1's is a satisfying assignment of $F$. If there is such a satisfying assignment, then the algorithm succeeds. Otherwise, any minimal satisfying assignment of $F$ must be at least $\varepsilon n$-isolated, since a minimal solution must be isolated in any direction where the value of a variable is 1.

Fix such a nearly isolated solution $x$, and fix $\varepsilon n$ critical clauses for $x$. If $\sigma$ is chosen randomly from $S$, then for each of the $\varepsilon n$ critical clauses at $x$, the probability that the critical variable occurs last among the variables of the clause is at least $1/k - 1/n$. Thus the expected number of times this event occurs is at least $\varepsilon n/k - \varepsilon/n$, and there is some $\sigma$ in $S$ that achieves at least the expectation. With respect to that $\sigma$, there is an encoding of $x$ using at most $n(1-\varepsilon/k)+1$ bits. As $s$ runs over all strings of that length, this encoding is encountered and decoded to produce $x$.

The time taken to check all solutions with at most $\varepsilon n$ 1's is $|F|2^{H(\varepsilon)n}$. If no solution is found, then at most $O(n^{3k})$ different permutations are examined, and each permutation requires at most $O(|F|2^{n(1-\varepsilon/k)+1})$ steps for a total of $O(|F|n^{3k}2^{n(1-\varepsilon/k)+1})$. To optimize the algorithm, choose $\varepsilon$ to minimize the sum of these terms. $\square$

The approximate values in the exponent of this running time for small $k$ are given Table 1. For large values of both $n$ and $k$, the running time approaches $2^{n-n/2k}$.

# 5 Conclusion

An obvious open problem is to find a deterministic algorithm for $k$-Sat that runs in time $O(\text{poly}(n)2^{n-n/k})$. It seems that additional insight into the structure of large sets accepted by $k$-CNFs is necessary.

Our satisfiability coding lemma gives information only about the number of length-$n$ minterms of a $k$-CNF, while the switching lemma gives information about the lengths of minterms after a restriction is applied. It seems likely that one could prove a stronger version of the switching lemma if one had good bounds on the number of other minterms.

In an insightful remark, Valiant [Val77] comments that the pursuit to understand the reasons for the inherent computational difficulty of problems has two complementary facets, the positive one of finding fast algorithms and the negative one of proving lower bounds on the inherent complexity. In fact, our question regarding the number of isolated points of a $k$-CNF led us to the discovery of the satisfiability coding lemma, which not only gives precise lower bounds on parity but yields somewhat unexpected insight into the satisfiability problem. It seems that progress can be made in both directions if one can further relate the syntactic properties of the formula to the structure of its solution space.

# Acknowledgments

The authors would like to thank Russell Impagliazzo, Sam Buss, and Vojtěch Rödl for helpful discussions. The authors would also like to thank the reviewer for helpful comments and for suggesting the generalization in Theorem 7.

## Acknowledgment of support

Pudlák was supported by grant number A1019602 of the Academy of Sciences of the Czech Republic and grant number INT-9600919/ME 103(1997) under the cooperation of MŠMT, the Czech Republic, and the National Science Foundation.

# References

[ASE92] N. Alon, J. Spencer, and P. Erdös. *The Probabilistic Method.* Wiley, New York, 1992.

[Bol85] B. Bollobás. *Random Graphs.* Academic Press, London, 1985.

[Bop97] R. Boppana. The average sensitivity of bounded-depth circuits. *Information Processing Letters*, 63:257–261, 1997.

[BS90] R. B. Boppana and M. Sipser. The complexity of finite functions. In *Handbook of Theoretical Computer Science, Vol. A*, pages 759–804. MIT Press, Cambridge, 1990.

[Coo71] S. A. Cook. The complexity of theorem-proving procedures. In *Conference Record of Third Annual ACM Symposium on Theory of Computing*, pages 151–158, Shaker Heights, Ohio, 3–5 May 1971. ACM.

[Hås86] J. Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, pages 6–20, Berkeley, Calif., 28–30 May 1986. ACM.

[HJP93] J. Håstad, S. Jukna, and P. Pudlák. Top-down lower bounds for depth 3 circuits. In *Thirty-Fourth Annual Symposium on Foundations of Computer Science*, pages 124–129, Palo Alto, Calif., 3–5 November 1993. IEEE Computer Society.

[Lev73] L. A. Levin. Universal sorting problems. *Problemy Peredaci Informacii*, 9:115–116, 1973.

[MS85] B. Monien and E Speckenmeyer. Solving satisfiability in less than $2^n$ steps. *Discrete Applied Mathematics*, 10:287–295, 1985.

[PPSZ98] R. Paturi, P. Pudlák, M. E. Saks, and F. Zane. An improved exponential-time algorithm for $k$-sat. In *1998 Annual IEEE Symposium on Foundations of Computer Science*, pages 628–637. IEEE Computer Society, 1998.

[PSZ97] R. Paturi, M. E. Saks, and F. Zane. Exponential lower bounds for depth 3 Boolean circuits. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 86–91, El Paso, Tex., 4–6 May 1997. ACM.

[Raz86] A. A. Razborov. Lower bounds on the size of bounded depth networks over a complete basis with logical addition. *Mathematische Zametki*, 41:598–607, 1986. English Translation in *Mathematical Notes of the Academy of Sciences of the USSR*.

[Raz95] A. A Razborov. Bounded arithmetic and lower bounds in boolean complexity. In *Feasible Mathematics II*, pages 344–386. Birkhäuser, Boston, 1995.

[Sch93] I. Schiermeyer. Solving 3-satisfiability in less than $1.579^n$ steps. In *Selected Papers from CSL 1992*, volume 702 of *Lecture Notes in Computer Science*, pages 379–394. Springer-Verlag, 1993.

[Sch96] I. Schiermeyer. Pure literal look ahead: An $O(1.497^n)$ 3-satisfiability algorithm. Preprint, 1996.

[Sch99] U. Schöning. A probabilistic algorithm for $k$-sat and constraint satisfaction problems. In *1999 Annual IEEE Symposium on Foundations of Computer Science*. IEEE Computer Society, 1999.

[Smo87] R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, pages 77–82, New York, 25–27 May 1987. ACM.

[Val77] L. G. Valiant. Graph-theoretic arguments in low-level complexity. In *Proceedings of the 6th Symposium on Mathematical Foundations of Computer Science*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176. Springer-Verlag, 1977.

[Zha96] W. Zhang. Number of models and satisfiability of sets of clauses. *Theoretical Computer Science*, 155(1):277–288, 26 February 1996.