# Satisfiability Certificates Verifiable in Subexponential Time *

Evgeny Dantsin[1] and Edward A. Hirsch[2]**

[1] Department of Computer Science, Roosevelt University, USA
[2] Steklov Institute of Mathematics at St.Petersburg, Russia

**Abstract.** It is common to classify satisfiability problems by their time complexity. We consider another complexity measure, namely the length of certificates (witnesses). Our results show that there is a similarity between these two types of complexity if we deal with certificates verifiable in subexponential time. In particular, the well-known result by Impagliazzo and Paturi [IP01] on the dependence of the time complexity of $k$-`SAT` on $k$ has its counterpart for the certificate complexity: we show that, assuming the exponential time hypothesis (ETH), the certificate complexity of $k$-`SAT` increases infinitely often as $k$ grows. Another example of time-complexity results that can be translated into the certificate-complexity setting is the results of [CIP06] on the relationship between the complexity of $k$-`SAT` and the complexity of `SAT` restricted to formulas of constant clause density. We also consider the certificate complexity of `CircuitSAT` and observe that if `CircuitSAT` has subexponential-time verifiable certificates of length $cn$, where $c < 1$ is a constant and $n$ is the number of inputs, then an unlikely collapse happens (in particular, ETH fails).

## 1 Introduction

If we assume $\mathbf{P} \neq \mathbf{NP}$, the question of refined complexity classification of **NP**-complete problems remains open. For example, what is the best possible running time for deciding $k$-`SAT`, `SAT`, or `CircuitSAT`? Is it possible to solve $k$-`SAT` in subexponential time? Is it possible to solve `SAT` or even `CircuitSAT` faster than using the trivial enumeration of all assignments? Although the questions like those above seem far enough from being resolved, many interesting results shedding more light on such questions have been appeared for the past two decades, see surveys in [DH09, PP10].

In this paper, we compare a time-complexity classification of problems in **NP** with a classification based on the length of certificates (witnesses). Note an asymmetry between these complexity measures. Any problem in **NP** can be

* Nominated as Best Paper candidate.

** Supported in part by Federal Target Programme "Scientific and scientific-pedagogical personnel of the innovative Russia" 2009-2013, by the grant NSh-5282.2010.1 from the President of RF for Leading Scientific Schools, by the Programme of Fundamental Research of RAS, and by RFBR grant 11-01-00760.

trivially solved by enumerating all possible candidates for a certificate. Therefore, if the certificate length is upper bounded by a function $\ell$ then the running time is upper bounded by $2^\ell$ up to the time needed for verifying a candidate. On the other hand, if the running time is upper bounded by a function $t$ then it is not necessarily true that the certificate length is upper bounded by $\lg t$ (unless $\mathbf{E} \subseteq \mathbf{NP}$, where $\mathbf{E}$ is the complexity class for exponential time with linear exponent).

We observe a similarity between the two types of complexity classifications for satisfiability problems. More specifically, we show that many known results on the time complexity of $k$-SAT, $\mathtt{SAT}_\Delta$ (the restriction of SAT to formulas whose clause density is at most $\Delta$), and CircuitSAT have their counterparts for the certificate complexity. It is important to note that this similarity holds for certificates defined as certificates *verifiable in subexponential time* (although the polynomial-time verification suffices for some cases). Precise definitions for the subexponential-time verification are given in Sect. 2. Our main results can be summarized as follows.

*Certificate complexity of* $k$-SAT*.* It is well known that $k$-SAT can be solved in time $O(2^{cn})$ where $n$ is the number of variables and $c < 1$ is a constant depending on $k$. This bound was obtained using different approaches: critical clauses [PPZ97, PPSZ98], local search [Sch99], covering codes [DGH+02]. The proof based on covering codes can be adapted to show that $k$-SAT has certificates of length $cn$ (we include this adapted proof for self-containedness).

Another known result on $k$-SAT is the result by Impagliazzo and Paturi [IP01] on increasing the time complexity of $k$-SAT as $k$ grows. They defined the sequence $\{s_k\}_{k\geq 3}$ where

$$s_k = \inf\{s \mid k\text{-}\mathtt{SAT} \text{ can be solved by an } O(2^{sn})\text{-time algorithm}\}.$$

The conjecture that $s_k > 0$ for all $k \geq 3$ is called the *Exponential Time Hypothesis* (ETH). Note that ETH is stronger than the $\mathbf{P} \neq \mathbf{NP}$ conjecture. It is shown in [IP01] that if ETH is true then $\{s_k\}$ increases infinitely often. We define the sequence $\{c_k\}_{k\geq 3}$ by

$$c_k = \inf\{c \mid k\text{-}\mathtt{SAT} \text{ has certificates of length } cn\}$$

and we show that if ETH is true then $\{c_k\}$ increases infinitely often too. To index the search space appearing in the proof of [IP01] by certificates of appropriate length, we use the *combinatorial* (also called *binomial*) *number system*, see e.g. [Knu05].

It is an intriguing open question whether $s_k = c_k$.

*Certificate complexity of* $\mathtt{SAT}_\Delta$*.* Using Schuler's reduction from $\mathtt{SAT}_\Delta$ to $k$-SAT [Sch05], it was shown that $\mathtt{SAT}_\Delta$ can be solved in time $O(2^{cn})$ with $c < 1$ [CIP06]. We translate this result into the certificate settings: $\mathtt{SAT}_\Delta$ has certificates of length $cn$. The combinatorial number system is again used in our proof.

The time complexity of $\texttt{SAT}_{\Delta}$ is characterized by the sequence $\{d_{\Delta}\}$ where

$$d_{\Delta} = \inf\{d \mid \texttt{SAT}_{\Delta} \text{ can be solved by an } O(2^{dn})\text{-time algorithm}\}.$$

It was shown in [CIP06] that this sequence is interwoven with $\{s_k\}$ and thus $s_\infty = d_\infty$, where $s_\infty = \lim_{k\to\infty} s_k$ and $d_\infty = \lim_{\Delta\to\infty} d_{\Delta}$. We characterize the certificate complexity of $\texttt{SAT}_{\Delta}$ by the sequence $\{b_{\Delta}\}$, where

$$b_{\Delta} = \inf\{b \mid \texttt{SAT}_{\Delta} \text{ has certificates of length } bn\},$$

and we show that the relationship between the certificate complexities $\{c_k\}$ and $\{b_{\Delta}\}$ is similar to the relationship between the time complexities $\{s_k\}$ and $\{d_{\Delta}\}$. In particular, $\lim_{k\to\infty} c_k = \lim_{\Delta\to\infty} b_{\Delta}$.

*Nondeterministic subexponential time and* `CircuitSAT`. The class **SE** consists of all parameterized problems that can be solved in time subexponential in the parameter [IPZ01]. In Sect. 5, we define the class **NSE** to be the class of all parameterized problems that have subexponential-time verifiable certificates of length bounded by the parameter. Note that there is an analogy between the pair **P** versus **NP** and the pair **SE** versus **NSE**. We also define a subexponential-time reducibility that preserves the certificate length and we observe that

- **NSE** is closed under this reducibility;
- `CircuitSAT` with the number of inputs as the parameter is complete for **NSE** under this reducibility.

It follows from the completeness of `CircuitSAT` that if `CircuitSAT` has certificates of length $cn$, where $n$ is the number of inputs and $c < 1$ is a constant, then **NSE** collapses to **SE**. Therefore, since ETH is a stronger assumption than **SE** $\neq$ **NSE**, ETH also implies that `CircuitSAT` has no certificates shorter than the number of inputs.

This observation can be viewed as a certificate offset of recent results on the time complexity of `CircuitSAT`. For example, it is shown by Paturi and Pudlák [PP10] that `CircuitSAT` cannot be solved by a one-sided probabilistic polynomial-time algorithm with success probability better than $2^{-n+o(n)}$ unless some unlikely complexity containments hold. On the other hand, Williams [Wil10] shows that even a slight improvement in the running time over exhaustive search for `CircuitSAT` implies a proof of $\mathbf{NEXP} \not\subseteq \mathbf{P/poly}$.

## 2 Definitions

**Definition 1 (parameterized problem, [FG06]).** *A* parameterized problem *is a pair $(L, p)$ consisting of a language $L \in \{0,1\}^*$ and a polynomial-time computable parameterization function $p : \{0,1\}^* \to \mathbb{N}$.*

**Definition 2 (verifier and certificate).** *A* verifier *for a parameterized problem $(L, p)$ is an algorithm $V$ such that*

$$x \in L \iff \exists w \in \{0,1\}^* \, (|w| \leq p(x) \text{ and } V \text{ accepts the pair } (x, w))$$

*where the string $w$ is called a* certificate *for $x$.*

*Remark 1.* In the definition above and throughout the paper, we use the word "algorithm" to denote a deterministic algorithm. However, all results of the paper hold if "algorithm" means a randomized algorithm.

**Definition 3 (subexponential verification scheme).** *A* subexponential verification scheme *for a parameterized problem $(L, p)$ is a family $\{V_t\}_{t \in \mathbb{N}}$ of verifiers for $(L, p)$ such that for each verifier $V_t$, the running time of $V_t$ on $(x, w)$ is*

$$|x|^{O(1)}\, 2^{p(x)/t}$$

*where the polynomial $|x|^{O(1)}$ may depend on $t$. If $(L, p)$ has a subexponential verification scheme, we also say that $L$ has* subexponential-time verifiable certificates of length *$p$.*

*Remark 2.* It would be more common if we defined subexponential verification schemes as a family of verifiers $V_\epsilon(x, w)$ like, for example, the definition of a family of subexponential reductions (SERF) in [IPZ01]. These two versions are equivalent, however we prefer the version with $1/t \to 0$ instead of $\epsilon \to 0$ to avoid discussions on the representation of $\epsilon$ (especially when it is given as a function of other parameters).

*Remark 3.* An important special case of subexponential verification schemes is the case where all verifiers $V_t$ are the same and each of them runs in time polynomial in both $p$ and $|x|$. If so, we say that $L$ has *polynomial-time verifiable* certificates of length $p$. An obvious example of this special case is the polynomial-time verification for $(\mathtt{SAT}, n)$: a certificate for a satisfiable formula is an $n$-bit string that encodes a satisfying assignment. Less obvious examples are given in Theorems 1 and 3 below.

*Remark 4.* All certificates considered in this paper are verifiable in subexponential time. To simplify the terminology, we omit the words "subexponential-time verifiable". Thus, throughout the paper, when we write "$L$ has certificates of length $p$", this means "$L$ has subexponential-time verifiable certificates of length $p$".

## 3 Shortest Certificates for $k$-SAT

The time complexity of $k$-SAT for $k \geq 3$ is characterized by the sequence $\{s_k\}_{k \geq 3}$ where

$$s_k = \inf\{s \mid k\text{-}\mathtt{SAT} \text{ can be solved by an } O(2^{sn})\text{-time algorithm}\}.$$

The current knowledge and open questions about this sequence can be described as follows:

– We know that $s_k < 1$. More exactly, $s_k \leq (1 - \mu/k)$ for some constant $\mu > 0$. This bound is obtained using critical clauses [PPZ97, PPSZ98], local search [Sch99], covering codes [DGH+02, MS11].

– We do not know whether $s_k = 0$. The conjecture that $s_k > 0$ for all $k \geq 3$ is called the *Exponential Time Hypothesis* (ETH).
– If ETH holds then $\{s_k\}$ increases infinitely often [IP01].
– Let $s_\infty = \lim_{k\to\infty} s_k$. The conjecture that $s_\infty = 1$ is called the *Strong Exponential Time Hypothesis* (SETH). The relationship between $s_\infty$ and the complexity of `SAT` is also unknown, where the complexity of `SAT` is the minimum number $s$ such that `SAT` can be solved in time $2^{sn}$ up to a polynomial in the input size.

The certificate complexity of $k$-`SAT` is defined below through a sequence similar to $\{s_k\}$.

**Definition 4 (certificate complexity for $k$-`SAT`).** *For each $k \geq 3$, let*

$$c_k = \inf\{c \mid k\text{-}\mathtt{SAT} \textit{ has certificates of length } cn\}.$$

*The limit of $\{c_k\}$ as $k \to \infty$ is denoted $c_\infty$.*

Note that $s_k \leq c_k$ for all $k \geq 3$ and $s_\infty \leq c_\infty$.

### 3.1 Upper bound on certificate length for $k$-`SAT`

The following theorem shows that $c_k < 1$ and, moreover, this inequality holds even for polynomial-time verifiable certificates.

**Theorem 1.** *For each $k \geq 3$ and for each $\epsilon > 0$, $k$-`SAT` has polynomial-time verifiable certificates of length $\left(1 - \lg \frac{k+1}{k} + \epsilon\right) n$.*

Certificates of the claimed length can be extracted from the algorithm that solves $k$-`SAT` in time $O\left(2^{\left(1-\lg \frac{k+1}{k}+\epsilon\right)n}\right)$ using covering codes [DGH+02]. Such a certificate includes the number of the ball containing a satisfying assignment and the index of this assignment in a search tree inside the ball. Although the proof essentially repeats that of [DGH+02], we include it here for the sake of self-containedness.

*Proof.* Let $F$ be a satisfiable $k$-CNF formula over $n$ variables. We show that a satisfying assignment for $F$ can be encoded using less than $n$ bits. Each assignment for $F$ is identified with a point in the Boolean cube $\{0,1\}^n$. The first step of the encoding is to cover the cube with Hamming balls of radius $\rho n$, where a value for $\rho$ will be chosen later. It is known that any such covering must contain at least $2^{(1-H(\rho))n}$ balls, where $H$ is the binary entropy function. An "almost" optimal covering (with at most $2^{(1-H(\rho)+\epsilon)n}$ balls for any $\epsilon > 0$) is constructed in [DGH+02] as follows.

The centers of the balls are viewed as a covering code for the cube. For any $\epsilon > 0$, we need a covering code of radius $\rho n$ that contains at most $2^{(1-H(\rho)+\epsilon)n}$ codewords. Consider a partition of $n$ bits into $n/b$ blocks of size $b$, where $b$ is a constant (without loss of generality, we can assume that $n$ is divisible by $b$ and $n$ is sufficiently large). Using a brute-force enumeration, we can find an optimal covering code of radius $\rho b$ for each block. Let $\mathcal{C} = \{w_1, \ldots, w_M\}$ be such a code, where $M$ is at most $2^{(1-H(\rho))b}$ up to a polynomial in $b$. The direct sum of $n/b$ copies of $\mathcal{C}$ is a covering code of radius $\rho n$ for the cube. It is easy to see that given $\rho$ and $\epsilon$, a value for $b$ can chosen such that this direct sum (denoted $\mathcal{C}^{n/b}$) has at most $2^{(1-H(\rho)+\epsilon)n}$ codewords. We encode each codeword $w_i \in \mathcal{C}$ by an integer $i$. Then each codeword in $\mathcal{C}^{n/b}$ can be encoded by a concatenation of $n/b$ integers from 1 to $M$ each. The length of this encoding is at most $(1-H(\rho)+\epsilon)n$. Moreover, given such a concatenation, the corresponding codeword (or, equivalently, the corresponding ball center) can be computed in time polynomial in $n$.

Assume that $F$ has a satisfying assignment in a ball of radius $\rho n$ centered at an assignment $A$. Then the encoding of $A$ (with at most $(1-H(\rho)+\epsilon)n$ bits) is the first part of a certificate for $F$. To construct the second part, we again refer to [DGH+02] where it is shown how to search for a satisfying assignment inside a ball. This search is essentially a recursive procedure that modifies $F$ and $A$ using the following approach: if the current assignment $\alpha$ does not satisfy the current formula $\phi$, take the first unsatisfied clause $l_1 \vee \ldots \vee l_h$ in $\phi$ and consider pairs $(\phi_1, \alpha_1), \ldots, (\phi_h, \alpha_h)$ where each $\alpha_i$ is obtained from $\alpha$ by flipping the value of the literal $l_i$ and each $\phi_i$ is obtained from $\phi$ by substituting the new value for $l_i$ in $\phi$. This procedure starts with $(F, A)$ and builds a recursion tree $T$ of depth at most $\rho n$. Since $F$ is a $k$-CNF formula, the degree of each node in $T$ is at most $k$. At least one leaf in $T$ is a pair $(\phi, \alpha)$ where $\alpha$ satisfies $\phi$. Hence, $\alpha$ satisfies $F$.

Thus, a satisfying assignment $\alpha$ in a ball of radius $\rho n$ centered at $A$ can be encoded by a path from the root to a leaf in $T$. Such a path is determined by a sequence of literals chosen in unsatisfied clauses. If we choose a literal $l_i$ in a clause $l_1 \vee \ldots \vee l_h$, we encode this choice by the integer $i$. The entire path can thus be encoded by a sequence of integers $i_1, \ldots, i_{\lfloor \rho n \rfloor}$ where $1 \le i_j \le k$ for each $j$. Removing the leading 1s in binary representation of these integers, we encode the path by a concatenation of $\lfloor \rho n \rfloor$ bit strings of length $\lfloor \lg k \rfloor$ each.

Finally, a certificate for $F$ is a pair, where the first element encodes the center of a ball containing a satisfying assignment and the second element encodes a path in $T$. For any $\epsilon$, the total length of this certificate is at most $(1 - H(\rho) + \epsilon)n + \rho n \lg k$. Taking $\rho = 1/(k+1)$, we have:

$$(1 - H(\rho) + \epsilon)n + \rho n \lg k \;=\; \left(1 - \lg \frac{k+1}{k} + \epsilon\right) n.$$

To verify it polynomial time, just compute the center $A$ of the ball from a given index and use a given path to modify $A$ to a satisfying assignment. □

### 3.2 The growth of certificate lengths for $k$-SAT

It is proved in [IP01] that ETH implies the following relationship between $s_k$ and $s_\infty$:

$$s_k \le s_\infty(1 - \sigma/(ek)), \tag{1}$$

where $\sigma$ is the solution of $H(\sigma) = s_\infty/2$ on $(0; 1/2]$. Therefore, if ETH holds then $\{s_k\}$ increases infinitely often. We prove a similar result for $\{c_k\}$.

**Theorem 2.** *If ETH holds then*

$$c_k \leq c_\infty(1 - \gamma/(ek)) \tag{2}$$

*where* $\gamma$ *is the solution of* $H(\gamma) = c_\infty/2$ *on* $(0; 1/2]$.

This theorem is proved using the following lemma from [IP01]:

**Lemma 1 ([IP01]).** *Let* $F$ *be a formula in* $k$*-CNF such that* $F$ *is not satisfiable by any assignment of weight*[3] *at most* $\delta n$*. For any* $\epsilon > 0$*, there exists* $k'$ *such that the following holds: The satisfiability of* $F$ *is equivalent to the satisfiability of the disjunction* $F_1 \vee \ldots \vee F_N$*, where* $N \leq 2^{\epsilon n}$ *and each* $F_i$ *is a formula in* $k'$*-CNF on at most* $n(1 - \delta/(ek))$ *variables. Moreover, this disjunction can be computed from* $F$ *in time* $n^{O(1)} 2^{\epsilon n}$.

*Proof (of Theorem 2).* We mimic the proof of inequality (1) in [IP01]. The proof shows how to construct an $O(2^{cn})$-time algorithm for $k$-SAT using an $O(2^{c'n})$-time algorithm for $k'$-SAT for certain $k' > k$ and $c' > c$. We must make sure that the decrease in the running time is accompanied by the decrease in the length of a certificate verifiable in subexponential time.

The algorithm constructed in [IP01] tests satisfiability of a given $k$-CNF formula $F$ as follows (here $\epsilon > 0$ and $w = \lfloor \sigma n \rfloor$):

1. Use exhaustive search to check all assignments of weight at most $w$. If at least one of them satisfies $F$, return "satisfiable".
2. Apply Lemma 1 (with $\delta = w/n$) to obtain $k'$-CNF formulas $F_1, \ldots, F_N$ on at most $n(1 - w/(ekn))$ variables each, where $N \leq 2^{\epsilon n}$.
3. Apply a $k'$-SAT algorithm to $F_i$'s; if at least one of them is satisfiable, return "satisfiable"; otherwise return "unsatisfiable".

In the certificate settings, we take $w = \lfloor \gamma n \rfloor$ and we bound the length of certificates considering two cases: the case of a satisfying assignment of low weight ($\leq w$), and the case of application of Lemma 1.

1. If $F$ is satisfied by an assignment of weight at most $w$ then $F$ has a certificate of length
$$\left\lceil \lg \binom{n}{w} \right\rceil + O(\lg n).$$
Such a certificate can be obtained using the *combinatorial* (also called *binomial*) *number system*, see e.g. [Knu05].
   (a) Consider the lexicographic order of all assignments ($n$-bit strings) of weight exactly $w$ and consider the numbering of assignments in this list by numbers from 0 to $\binom{n}{w} - 1$. Let $A$ be an assignment with 1s on positions $n > a_w > \ldots > a_1 \geq 0$ and 0s on all other positions. We encode $A$ by

[3] An assignment is identified with a bit string; the *weight* of an assignment is the number of 1s in the string.

its number $N_A$ in the lexicographic order, where $N_A$ can be computed as the following sum:

$$N_A = \binom{a_w}{w} + \ldots + \binom{a_1}{1}.$$

Obviously, the decoding can be done efficiently: first, find $a_w$, then proceed to lower terms.

(b) To encode an assignment of weight $w - i$, we first encode $i$ and then append the number

$$\binom{a_{w-i}}{w-i} + \ldots + \binom{a_1}{1}.$$

The encoding of $i$ has length $O(\lg n)$ if we encode $i$ as follows: $1 \ldots 10\langle i \rangle$ where $\langle i \rangle$ is $i$ written in binary and the number of 1s is equal to the length of the binary representation of $i$.

2. In the case of application of Lemma 1, we specify the index $i$ of the first satisfiable formula $F_i$ by $\lceil \epsilon n \rceil$ bits. The formula itself can be found by running the procedure in Lemma 1, which takes time $2^{\epsilon n} n^{O(1)}$. These $\lceil \epsilon n \rceil$ bits are appended with the the certificate for $F_i$. By definition of $c_{k'}$, its length is bounded by $(c_{k'} + \epsilon)$ times the number of variables in $F_i$. Finally, we put leading 0 on top of all that to indicate that this is the case of application of Lemma 1.

In total, we have the following upper bound on the certificate length:

$$\begin{aligned}
&\max\{\lceil \lg \tbinom{n}{w} \rceil + O(\lg n), 1 + \lceil \epsilon n \rceil + (c_{k'} + \epsilon)\lceil n(1 - w/(ekn)) \rceil\} &=\\
&n \cdot \max\{H(w/n), c_{k'}(1 - w/(ekn)) + 2\epsilon\} + O(1) &=\\
&n \cdot \max\{c_\infty/2, c_\infty(1 - \gamma/(ek)) + 2\epsilon\} + O(1) &=\\
&n \cdot (c_\infty(1 - \gamma/(ek)) + 2\epsilon) + O(1).
\end{aligned}$$

□

**Corollary 1.** *If ETH holds then the sequence $\{c_k\}$ increases infinitely often as $k$ grows.*

*Proof.* Straightforwardly follows from (2). □

## 4 Shortest Certificates for $\mathtt{SAT}_\Delta$

The *clause density* of a CNF formula with $m$ clauses over $n$ variables is the ratio $m/n$. For any positive constant $\Delta$, we write $\mathtt{SAT}_\Delta$ to denote the restriction of SAT to formulas whose clause density is at most $\Delta$.

**Lemma 2.** *For each $\Delta > 0$, $k \geq 3$, and $c > 0$, if $k$-SAT has (polynomial-time verifiable) certificates of length $cn$ then $\mathtt{SAT}_\Delta$ has (polynomial-time verifiable) certificates of length*

$$\left(c + \frac{(\Delta + 1/k)\lg e}{2^{ck}}\right) n + o(n).$$

*Proof.* Let $F$ be a satisfiable formula in CNF with $m/n \le \Delta$. We build a certificate for $F$ using Schuler's reduction [Sch05] which transforms any CNF formula into an equivalent disjunction of an exponential number of $k$-CNF formulas. This reduction can be represented as a labeled binary tree in which the root is labeled by $F$ and the leaves are labeled by $k$-CNF formulas [CIP06]. Any path from the root to a leaf is given by a sequence of choices:

- either choose a left branch where a clause is reduced to a $k$-clause;
- or choose a right branch where the number of variables is decreased by $k$ variables.

The maximum number of branchings to the left is $m$; the maximum number of branchings to the right is $n/k$ (without loss of generality we can assume that $n$ is divisible by $k$).

Consider a path from the root to a leaf such that the path contains exactly $r$ branchings to the right. Then the $k$-CNF formula at the leaf has $n-kr$ variables. Let $P_r$ be the set of all such paths. Any path in $P_r$ can be identified with a bit string of length $m+n/k$ that has exactly $r$ ones. We encode these strings using the combinatorial number system [Knu05], like we encoded assignments of fixed weight in the proof of Theorem 2. Then any path in $P_r$ is encoded by a bit string of length

$$\left\lfloor \lg \binom{m+n/k}{r} \right\rfloor + 1$$

and the decoding can be done in polynomial time.

Given a path from the root to a leaf, the $k$-CNF formula at this leaf can be computed in time polynomial in the size of $F$. Therefore, a certificate for $F$ is a path to a leaf $L$ labeled by a satisfiable $k$-CNF formula $F_L$ plus a certificate for $F_L$. If the path to $L$ has $r$ branchings to the right then a certificate for $F$ can be defined as the concatenation of the following three strings:

- the encoding of the integer $r$ with $\lfloor \lg(n/k) \rfloor + 1$ bits;
- the encoding of the path to $L$ with $\lfloor \lg \binom{m+n/k}{r} \rfloor + 1$ bits;
- the encoding of a certificate for $F_L$ with $\lfloor c(n-kr) \rfloor + 1$ bits.

Now we show

$$\lg(n/k) + \lg \binom{m+n/k}{r} + c(n-kr) \;\le\; \left( c + \frac{(\Delta + 1/k)\lg e}{2^{ck}} \right) n + o(n).$$

Since the first term in the left-hand side is sublinear, it suffices to upper bound the sum of the other two terms. We estimate it using the same way as in [CIP06]:

$$\begin{aligned}
\lg \tbinom{m+n/k}{r} + c(n-kr) &\le \lg \left( \textstyle\sum_{r=0}^{m+n/k} \tbinom{m+n/k}{r} 2^{c(n-kr)} \right) \\
&\le \lg \left( 2^{cn} \left(1 + 2^{-ck}\right)^{m+n/k} \right) \\
&\le cn + (m+n/k) \lg \left( e^{2^{-ck}} \right) \\
&\le cn + \tfrac{(m+n/k)\lg e}{2^{ck}} \\
&\le \left( c + \tfrac{(\Delta+1/k)\lg e}{2^{ck}} \right) n.
\end{aligned}$$

Given a certificate described above, the verification of satisfiability of $F$ consists of two steps. The first step is to decode the certificate into a $k$-CNF formula $G$ and a certificate of satisfiability of $G$. This can be done in polynomial time. The second step is to verify satisfiability of $G$. If a certificate for $G$ is verifiable in polynomial time then this step can also be done in polynomial time. □

**Theorem 3.** *For each $\Delta > 0$, there exists $b < 1$ such that $\mathtt{SAT}_\Delta$ has polynomial-time verifiable certificates of length $bn$.*

*Proof.* We apply Lemma 2 choosing $k$ and $c$ such that

$$c + \frac{(\Delta + 1/k)\lg e}{2^{ck}} < 1.$$

Namely, if $c = 1 - \lg(1 + 1/k) + \epsilon$ for some $\epsilon > 0$ (Theorem 1) then

$$\frac{(\Delta + 1/k)\lg e}{2^{ck}} \le \frac{O(\Delta)}{2^k}.$$

Now if we take $k = r\lg(\Delta + 2)$, where $r$ is a sufficiently large constant, we have

$$\begin{aligned} c + \tfrac{(\Delta+1/k)\lg e}{2^{ck}} &\le 1 - \lg\left(1 + \tfrac{1}{r\lg(\Delta+2)}\right) + \epsilon + \tfrac{O(\Delta)}{2^{r\lg(\Delta+2)}} \\ &\le 1 - \tfrac{O(1)}{r\lg(\Delta+2)} + \epsilon + \tfrac{O(1)}{(\Delta+2)^{r-1}} < 1. \end{aligned}$$

□

Without loss of generality, we can assume that the clause density $\Delta$ is a positive integer. Then, similarly to the case of $k$-$\mathtt{SAT}$, the time complexity of $\mathtt{SAT}_\Delta$ is characterized by the sequence $\{d_\Delta\}_{\Delta \ge 1}$ where

$$d_\Delta = \inf\{d \mid \mathtt{SAT}_\Delta \text{ can be solved by an } O(2^{dn})\text{-time algorithm}\}.$$

It is known that $d_\Delta < 1$ for all $\Delta$. More exactly, $\mathtt{SAT}$ can be solved in time $2^{(1-1/O(\lg\Delta))n}$ up to a factor polynomial in the size of the input formula [CIP06, DH09]. It is also known that $\{d_\Delta\}$ is interwoven with $\{s_k\}$. Namely, as shown in [CIP06],

- for any $k$ and for any $\epsilon > 0$, there exists $\Delta$ such that $s_k \le d_\Delta + \epsilon$;
- for any $\Delta$ and for any $\epsilon > 0$, there exists $k$ such that $d_\Delta \le s_k + \epsilon$.

Therefore, $s_\infty = d_\infty$ where $d_\infty = \lim_{\Delta\to\infty} d_\Delta$.

We define an analog of $\{d_\Delta\}$ in the certificate settings and show a similarity between the two sequences.

**Definition 5 (certificate complexities for $\mathtt{SAT}_\Delta$).** *For each $\Delta \ge 1$, let*

$$b_\Delta = \inf\{b \mid \mathtt{SAT}_\Delta \textit{ has certificates of length } bn\}.$$

*Similarly to $d_\infty$, we define $b_\infty = \lim_{\Delta\to\infty} b_\Delta$.*

**Lemma 3.** *For each $\Delta > 0$ and $\epsilon > 0$, there exists $k$ such that $b_\Delta \leq c_k + \epsilon$.*

*Proof.* Consider two cases: $c_\infty > 0$ and $c_\infty = 0$. In the case of $c_\infty > 0$, we apply Lemma 2 with $k$ such that $c_k > 0$. Then we have

$$b_\Delta \;\leq\; c_k + \frac{(\Delta + 1/k)\lg e}{2^{c_k k}} + o(1)$$

for each $\Delta > 0$. Taking $k$ sufficiently large, we can make the fraction in the right-hand side arbitrarily small. If $c_\infty = 0$, we can apply Lemma 2 with arbitrarily small $c > 0$. In particular, if we take $c$ as a function of $k$ such that $ck \to \infty$ as $k \to \infty$, we can make the right-hand side arbitrarily small. Hence $b_\Delta = 0$ in this case. □

**Corollary 2.** $b_\infty \leq c_\infty$

*Proof.* Take $\Delta, k \to \infty$ and $\epsilon \to 0$. □

**Lemma 4 (Sparsification Lemma, [IPZ01]).** *Let $F$ be a formula in $k$-CNF. There is a function $f(k,\epsilon)$ upper bounded by a polynomial in $\frac{1}{\epsilon}$ such that for any $\epsilon > 0$, the satisfiability of $F$ is equivalent to the satisfiability of the disjunction $F_1 \vee \ldots \vee F_N$ over the same set of variables, where $N \leq 2^{\epsilon n}$ and each $F_i$ is a $k$-CNF formula in which every variable occurs at most $f(k,\epsilon)$ times. Moreover, this disjunction can be computed from $F$ in time $n^{O(1)}\,2^{\epsilon n}$.*

**Lemma 5.** *For any $k \geq 3$ and for any $\epsilon > 0$, we have $c_k \leq b_\infty + \epsilon$.*

*Proof.* Similarly to [CIP06, Corollary 2], the proof proceeds by application of Lemma 4. Given $k \geq 3$ and $\epsilon > 0$, we show that $k$-`SAT` has certificates of length $(b_\infty + \epsilon)n$. Namely, we construct a subexponential verification scheme $\{V_t\}$, where each verifier $V_t$ runs in time

$$|F|^{O(1)}\,2^{(b_\infty+\epsilon)n/t} \tag{3}$$

where $|F|$ is the size of the input $k$-CNF formula $F$.

Each $V_t$ starts with sparsifying $F$ by Lemma 4. The parameter of the sparsification procedure is chosen so that the procedure runs in time

$$|F|^{O(1)}\,2^{(b_\infty+\epsilon)n/2t}.$$

Let $\Delta = \Delta(k,\epsilon)$ be the maximum clause density of the formulas $F_1, \ldots, F_s$ returned by the sparsification procedure. The input string $w$ for $V_t$ is interpreted as a certificate of satisfiability for some $F_j$. Therefore, $V_t$ tests each formula $F_i$: whether $w$ is a certificate for $F_i$. This test is done using a subexponential verification scheme $\{U_t\}$ for $(\mathtt{SAT}_\Delta, b_\Delta + \epsilon)$. More exactly, the verifier $V_t$ uses $U_{2t}$ and, thus, the test of $F_i$ runs in time

$$|F|^{O(1)}\,2^{(b_\Delta+\epsilon)n/2t}.$$

Since $b_\Delta \leq b_\infty$, the overall running time of $V_t$ is (3). □

**Corollary 3.** $c_\infty \le b_\infty$

*Proof.* Take $k \to \infty$ and $\epsilon \to 0$. □

**Theorem 4.** $c_\infty = b_\infty$

*Proof.* Corollaries 2 and 3. □

**Theorem 5.** *If ETH holds then the sequence $\{b_\Delta\}$ of certificate complexities for $\mathtt{SAT}_\Delta$ increases infinitely often.*

*Proof.* Suppose that $b_{\Delta_0} = b_\infty$ for some $\Delta_0$. Then, by Lemma 3, there exists $k_0$ such $c_{k_0} \ge b_\infty$. Since $b_\infty = c_\infty$ and $\{c_k\}$ is nondecreasing, we have $c_k = c_\infty$ for all $k \ge k_0$, which contradicts Theorem 2. □

## 5 Shortest Certificates for `CircuitSAT`

**Definition 6 (subexponential time).** *We say that a parameterized problem $(L, p)$ can be solved in* subexponential time *if for any $t \in \mathbb{N}$, there exists an algorithm that decides $L$ in time $|x|^{O(1)}\, 2^{p(x)/t}$, where $x$ is an instance. The class* **SE** *consists of all parameterized problems $(L, p)$ that can be solved in subexponential time.*

**Definition 7 (nondeterministic subexponential time).** *The class* **NSE** *consists of all parameterized problems $(L, p)$ that have subexponential verification schemes.*

*Remark 5.* Note that **NSE** is to **SE** as **NP** is to **P**: the larger class requires a verifiable certificate to accept a "yes" instance. There are two differences:

- subexponential time versus polynomial time;
- the bound $|w| \le p(x)$ on the certificate length in the case of parameterized problems $(L, p) \in$ **NSE** versus the bound $|w| \le |x|^{O(1)}$ in the case of problems in **NP**.

The class **SE** is closed under reducibility defined in [IPZ01] and called *subexponential reduction families* (SERFs for short). Informally, a SERF from $(L, p)$ to $(L', p')$ is a collection of Turing reductions $R_t$ from $L$ to $L'$ such that each reduction runs in time $|x|^{O(1)}\, 2^{p(x)/t}$ and allows at most a linear increase of the parameter. We define a "strict" version of SERFs under which **NSE** is closed.

**Definition 8 (strict SERF).** *We say that $R$ is a* strict subexponential reduction family *(strict SERF) from a parameterized problem $(L, p)$ to a parameterized problem $(L', p')$ if $R$ is a sequence of algorithms $R_t$ such that*

- *each algorithm $R_t$ takes a string $x \in \{0, 1\}^*$ as input and outputs strings $y_1, \ldots, y_m$, where $m \le 2^{p(x)/t}$;*
- *each $R_t$ runs in time $|x|^{O(1)}\, 2^{p(x)/t}$;*
- *$p'(y_i) \le p(x)$ for all $1 \le i \le m$;*

– *for every* $x \in \{0,1\}^*$*, we have*

$$x \in L \iff \bigvee_{1 \le i \le m} (y_i \in L').$$

*Remark 6.* A strict SERF is a special case of a SERF, where the word "strict" alludes to two refinements:

- a strict SERF is a disjunctive truth table reduction, while a SERF is a Turing reduction;
- a strict SERF does not increase the parameter, while a SERF allows multiplying the parameter by an arbitrary constant.

Note also that if we allowed a slight increase of the parameter

$$p'(y_i) \le p(x) + o(p(x)),$$

we would have an equivalent definition.

**Theorem 6.** **NSE** *is closed under strict SERFs: if* $(L,p)$ *has a strict SERF to* $(L',p') \in \mathbf{NSE}$*, then* $(L,p) \in \mathbf{NSE}$.

*Proof.* A certificate for $x$ is a certificate for a $y_i$ such that $y_i \in L'$. The verification of this certificate includes generating $y_1, \dots, y_m$ with checking each of them: whether the given certificate is a certificate for $y_j$. □

**Theorem 7.** `CircuitSAT` *with the number of inputs as the parameter is complete for* **NSE** *under strict SERFs.*

*Proof.* Consider $(L,p) \in \mathbf{NSE}$. Let $t \in \mathbb{N}$. Consider a Turing machine that verifies certificates of length $p(x)$ in time $|x|^{O(1)}\, 2^{p(x)/2t}$. It is well-known that the machine can be transformed into a circuit with $p(x)$ inputs (after hardwiring a specific $x$) and size polynomial in the length of the machine's input and quadratic in the running time. The reduction $R_t$ outputs this circuit. □

**Corollary 4.** *If* `CircuitSAT` *has certificates of length* $cn$*, where* $n$ *is the number of inputs and* $c < 1$ *is a constant, then* $\mathbf{SE} = \mathbf{NSE}$.

*Proof.* Suppose that `CircuitSAT` has certificates of length $cn$. We show that if $(L,p) \in \mathbf{NSE}$ then $(L,p) \in \mathbf{SE}$. Since $(L,p)$ has a strict SERF to `CircuitSAT` with $p$ inputs, $L$ has certificates of length $cp$. That is, $(L,cp) \in \mathbf{NSE}$ and therefore $(L,cp)$ has a strict SERF to `CircuitSAT` with $cp$ inputs. Using the supposition again, we obtain $(L,c^2p) \in \mathbf{NSE}$. Continuing, we can conclude that $L$ has certificates of arbitrarily small length. Hence, $L$ can be solved in subexponential time. □

*Remark 7.* It follows from Corollary 4 that if ETH is true then there is no constant $c < 1$ such that `CircuitSAT` has certificates of length $cn$. Indeed, $(3\text{-}\mathtt{SAT}, n) \in \mathbf{NSE}$ where $n$ is the number of variables. However, if ETH is true then $(3\text{-}\mathtt{SAT}, n) \notin \mathbf{SE}$, i.e., ETH implies $\mathbf{SE} \ne \mathbf{NSE}$.

## Acknowledgements

This work was done in part when the authors attended Dagstuhl Seminar 10441, Exact Complexity of NP-Hard Problems, November 2010. The authors are grateful to its participants for numerous enlightening discussions.

## References

[CIP06] Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. A duality between clause width and clause density for SAT. In *Proceedings of the 21st Annual IEEE Conference on Computational Complexity, CCC 2006*, pages 252–260. IEEE Computer Society, 2006.

[DGH+02] Evgeny Dantsin, Andreas Goerdt, Edward A. Hirsch, Ravindran Kannan, Jon Kleinberg, Christos H. Papadimitriou, Prabhakar Raghavan, and Uwe Schöning. A deterministic $(2 - 2/(k+1))^n$ algorithm for $k$-SAT based on local search. *Theoretical Computer Science*, 289(1):69–83, 2002.

[DH09] Evgeny Dantsin and Edward A. Hirsch. Worst-case upper bounds. In *Handbook of Satisfiability*, chapter 12, pages 403–424. IOS Press, 2009.

[FG06] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006.

[IP01] Russell Impagliazzo and Ramamohan Paturi. On the complexity of $k$-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity. *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

[Knu05] Donald E. Knuth. *Generating All Combinations and Partitions*, volume 4 of *The Art of Computer Programming*, fascicle 3, pages 1–6. Addison-Wesley, 2005.

[MS11] Robin A. Moser and Dominik Scheder. A full derandomization of Schöning's $k$-SAT algorithm. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing, STOC 2011*. ACM, 2011. To appear.

[PP10] Ramamohan Paturi and Pavel Pudlák. On the complexity of circuit satisfiability. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing, STOC 2010*, pages 241–250. ACM, 2010.

[PPSZ98] Ramamohan Paturi, Pavel Pudlák, Michael E. Saks, and Francis Zane. An improved exponential-time algorithm for $k$-SAT. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science, FOCS 1998*, pages 628–637, 1998.

[PPZ97] Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science, FOCS 1997*, pages 566–574, 1997.

[Sch99] Uwe Schöning. A probabilistic algorithm for $k$-SAT and constraint satisfaction problems. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science, FOCS 1999*, pages 410–414, 1999.

[Sch05] Rainer Schuler. An algorithm for the satisfiability problem of formulas in conjunctive normal form. *Journal of Algorithms*, 54(1):40–44, 2005.

[Wil10] Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing, STOC 2010*, pages 231–240. ACM, 2010.