On the Exact Complexity of Evaluating Quantified k-CNF

Chris Calabro[1], Russell Impagliazzo[2,⋆], and Ramamohan Paturi[2,⋆⋆]

[1] Google Inc.
[2] Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093-0404, USA

Abstract. We relate the exponential complexities $2^{s(k)n}$ of k-SAT and the exponential complexity $2^{s(\Pi_2 3\text{-SAT})n}$ of $\Pi_2$3-SAT (evaluating formulas of the form $\forall \boldsymbol{x} \exists \boldsymbol{y} \phi(\boldsymbol{x}, \boldsymbol{y})$ where ϕ is a 3-CNF in $\boldsymbol{x}$ variables and $\boldsymbol{y}$ variables) and show that $s(\infty)$ (the limit of $s(k)$ as $k \to \infty$) is at most $s(\Pi_2 3\text{-SAT})$. Therefore, if we assume the Strong Exponential-Time Hypothesis, then there is no algorithm for $\Pi_2$3-SAT running in time 2^{cn} with $c < 1$. On the other hand, a nontrivial exponential-time algorithm for $\Pi_2$3-SAT would provide a k-SAT solver with better exponent than all current algorithms for sufficiently large k. We also show several syntactic restrictions of $\Pi_2$3-SAT have nontrivial algorithms, and provide strong evidence that the hardest cases of $\Pi_2$3-SAT must have a mixture of clauses of two types: one universal literal and two existential literals, or only existential literals. Moreover, the hardest cases must have at least $n - o(n)$ universal variables, and hence only $o(n)$ existential variables. Our proofs involve the construction of efficient minimally unsatisfiable k-CNFs and the application of the Sparsification Lemma.

1 Introduction

From the viewpoint of exponential complexity of **NP**-complete problems, the most studied and best understood problems are probably the restricted versions of the general satisfiability problem (SAT), in particular, k-SAT, the restriction of SAT to k-CNFs, and CNF-SAT, the restriction to general CNFs. There has been a sequence of highly nontrivial and interesting algorithmic approaches to these problems [Sch99, PPZ99, PPSZ05, Sch05, DW05, Rol05], where the best known constant factor improvements in the exponent are of the form $1 - 1/\Theta(k)$ for k-SAT and $1 - 1/\Theta(\lg c)$ for CNF-SAT with at most cn clauses. More recently,

⋆ This research is supported by supported by the Simonyi Fund, the Bell Company Fellowship and the Fund for Math, and NSF grants DMS-083573, CNS-0716790 and CCF-0832797.

⋆⋆ This research is supported by NSF grant CCF-0947262 from the Division of Computing and Communication Foundations. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

V. Raman and S. Saurabh (Eds.): IPEC 2010, LNCS 6478, pp. 50–59, 2010.

the algorithmic approaches have been generalized to obtain improved exponential time algorithms for bounded-depth linear size circuits [CIP09]. Also, a sequence of papers ([IPZ98, IP01, CIKP08, CIP06]) has shown many nontrivial relationships between the exponential complexities of these problems, and helped characterize their hardest instances (under the assumption that they are indeed exponentially hard).

We consider the complexity of evaluating a quantified k-CNF ϕ. A quantified k-CNF is a quantified expression of the following form $Q_1\boldsymbol{x_1}\cdots Q_i\boldsymbol{x_i}\ \phi$ where $\phi \in$ k-CNF, each $\boldsymbol{x_j}$ is a tuple of Boolean variables, and each $Q_j \in \{\forall, \exists\}$. Similarly, we define a quantified circuit to be a quantified expression $Q_1\boldsymbol{x_1}\cdots Q_i\boldsymbol{x_i}\ \phi$ where ϕ is a Boolean circuit. For $k = 2$, the problem of evaluating quantified k-CNF can be solved in linear time [APT79], but for $k \geq 3$, it is PSPACE-complete. Note that the exponential complexity of evaluating arbitrary quantified circuits C, which is at least as hard, is at most $2^n\text{poly}(|C|)$: evaluation of a circuit can be carried out in time polynomial in its size, and backtracking need explore at most 2^n paths, each of length at most n, where n is the number of variables of the input (and this only requires polynomial space as well). We ask whether there exist nontrivial exponential-time algorithms for evaluating quantified k-CNF for $k \geq 3$.

Since we cannot hope for a nontrivial, unconditional result about the exponential complexity of qantified k-CNF, we will relate the exponential complexity of evaluating quantified k-CNF to the exponential complexities of k-SAT. As in [IP01], we define the *exponential complexity* of an evaluation problem Φ to be $s(\Phi) = \inf\{c|\exists$ a randomized algorithm for evaluating an instance ϕ of Φ with time complexity $\text{poly}(m)2^{cn}$ where m is the size of ϕ and n is the number of variables$\}$. While [IP01] did not explicitly use the words 'randomized algorithm' in its definition of $s(k)$ (it uses the word 'algorithm' without qualification), it meant to employ a broader class of algorithms (randmozed algorithms) so as to define the constants $s(k)$ robustly.

Let **ETH** denote the Exponential-Time Hypothesis: $s(3) > 0$. Let $s(\infty)$ denote the limit of the sequence $\{s(k)\}$. [IP01] proposed the open question whether $s(\infty) = 1$, which we will call the *Strong Exponential-Time Hypothesis* (**SETH**). Since the best known upper bounds for $s(k)$ are all of the form $1 - 1/\Theta(k)$, **SETH** is plausible.

As with CNF-SAT, we will consider various syntactic restrictions of quantified k-SAT to arrive at a *minimally complex* set of formulas which are as hard as the standard existentially quantified 3-CNFs as far as exponential complexity is concerned. However, merely bounding k does not seem to be enough. Our main restriction is to bound the number of alternations of quantifier type. Define $\Pi_i k$-SAT ($\Sigma_i k$-SAT) as the problem of evaluating a quantified k-CNF $Q_1\boldsymbol{x_1}\cdots Q_i\boldsymbol{x_i}\ \phi$ where $\phi \in k$-CNF, each $\boldsymbol{x_j}$ is a tuple of Boolean variables, and each $Q_j \in \{\forall, \exists\}$ is $\forall$ iff j is odd (even). Using the earlier notation, $s(\Pi_i k\text{-SAT})$ and $s(\Sigma_i k\text{-SAT})$ denote the exponential comlexities of the evaluation problems $\Pi_i k$-SAT and $\Sigma_i k$-SAT respectively. Note that k-SAT is the same problem as

$\Sigma_1 k$-SAT (also, $s(k) = s(\Sigma_1 k\text{-SAT})$) and that $\Pi_1 k$-SAT is the problem of evaluating universally quantified k-CNF, which can be done in polynomial time.

Our main result is $s(\infty) \leq s(\Pi_2 3\text{-SAT})$. Thus **SETH** would imply that for $\Pi_2 3$-SAT, virtually no improvement over the 2^n exhaustive search algorithm is possible. Since $\Pi_i k$-SAT is at least as hard when $i \geq 2$, $k \geq 3$, the complexity of problems in the polynomial hierarchy would seem to top off rather early. Of course, SETH is a very strong assumption, so this evidence should be considered weak. Conversely, a single nontrivial algorithm for $\Pi_2 3$-SAT would provide a k-SAT solver better than all current algorithms for sufficiently large k.

We also show several syntactic restrictions of $\Pi_2 3$-SAT that have nontrivial algorithms, concluding that the hardest cases of $\Pi_2 3$-SAT must have a mixture of clauses of two types under the assumption **SETH**: one universal literal and two existential literals, or only existential literals. Algorithmic design may benefit by concentrating on the hard cases, in much the same way that much of the progress in k-SAT solvers has been driven by focusing on the hard cases – when there are a linear number of clauses, only one solution, expanding variable-dependence graph, etc.

Lastly, we relate the exponential complexities $s(\Pi_i k\text{-SAT})$ $(s(\Sigma_i k\text{-SAT}))$ for various i, k when the clause density is not too high, although it may well be that these are all simply 1.

[Wil02] explored similar problems, namely exponential upper bounds for solving quantified constraint satisfaction problems (QCSP) with domain size d and cn constraints, each of width $\leq k$. When $d = 2$ and the formula is in k-CNF, we'll call this k-QCNF. [Wil02] solves QCSP for $d = 3$ in time $O(2^{.669cn})$, QCNF in $O(2^{.536cn})$ time and space, 3-QCNF in time $O(2^{.610cn})$, and the special case of 3-QCNF with $c = 1$ in time $O(2^{.482n})$. Each of these techniques outperforms exhaustive search only when c is very small, certainly no more than 2. This is probably not an accident: our results suggest that finding a substantially improved algorithm for moderate c, even for low quantifier depth, is unlikely.

The main step of our lower bound proof is to encode a k-CNF as a $\Pi_2 3$-CNF using only $o(n)$ additional variables. For this encoding, we construct a minimally unsatisfiable CNF formula with many clauses while using few additional variables. Another key technique in our proofs is the Sparsification Lemma [IPZ98, CIP06]:

Lemma 1 (Sparsification Lemma). *$\exists$ algorithm A $\forall k \geq 2, \epsilon \in (0,1], \phi \in$ k-CNF with n variables, $A_{k,\epsilon}(\phi)$ outputs a sequence $\phi_1, \ldots, \phi_s \in$ k-CNF with $n^{O(k)}$ delay such that*

1. $s \leq 2^{\epsilon n}$
2. $\mathrm{sol}(\phi) = \bigcup_i \mathrm{sol}(\phi_i)$, *where* $\mathrm{sol}(\phi)$ *is the set of satisfying assignments of* ϕ
3. $\forall i \in [s]$ *each disjunction of h literals occurs* $\leq O(\frac{k}{\epsilon})^{3(k-h)}$ *times in* ϕ_i, *which also implies*
   (a) *each literal occurs* $\leq O(\frac{k}{\epsilon})^{3k}$ *times, or* $\leq O(\frac{1}{\epsilon}\lg\frac{1}{\epsilon})$ *times for* $k = 2$, *in* ϕ_i
   (b) $|\phi_i| \leq O(\frac{k}{\epsilon})^{3k} n$.

1.1 Notation

A CNF is a set of clauses and a clause is a set of literals. A CNF F is regarded as a formula at times and as a set of clauses at other times. Intended meaning should be clear from the context. For example, $F \subseteq G$ indicates that the CNFs F and G should ve viewed as sets of clauses. $|F|$ denotes the number of clauses in the CNF F. A k-clause contains at most k literals. A k-CNF is a CNF which contains only k-clauses. We also use CNF to denote the set of all CNFs. If ϕ is a formula, then $\mathrm{var}(\phi)$ denotes set of variables occurring in ϕ. If a is a partial assignment to the variables of ϕ, $\phi|a$ denotes the restriction of ϕ to a. We use SAT to denote the set of satisfiable formulas.

2 $\Pi_2$3-SAT Lower Bound

Our proof uses a construction similar to [Che09], proposition 3.2, that allows us to reduce the clause width of a k-CNF using a small number of new quantified variables. For this, we need for all k' a "minimally unsatisfiable" k'-CNF which includes a large number of $k' - 1$ clauses. Moreover, we would like to employ as few additional variables as possible, certainly no more than $o(n)$ additional variables. The following construction suffices for our purposes. However, it is an open question whether there are other constructions that employ fewer variables. Our construction differs from [Che09] primarily in its efficiency, since we are more concerned with exponential complexity rather than hardness for a class closed under polynomial time reductions, such as NP or coNP.

Say a pair (F, G) of formulas in CNF is *minimally unsatisfiable* iff $\forall F' \subseteq F\ ((F' \wedge G) \notin \mathrm{SAT} \leftrightarrow F' = F)$. In other words, pair (F, G) is minimally unsatisfiable iff $F \wedge G$ is unsatisfiable, but for any proper subset $F' \subsetneq F$, $F' \wedge G$ is satisfiable. Say that $\phi \in \mathrm{CNF}$ is *combinable* with (F, G) iff $|\phi| \leq |F|$ and $\mathrm{var}(\phi) \cap \mathrm{var}(F \wedge G) = \emptyset$. In this case, letting $f : \phi \to F$ be an arbitrary injection, we define

$$\mathrm{combine}(\phi, F, G) = G \cup \{\{\bar{l}\} \cup f(C) \mid l \in C \in \phi\} \cup (F - \mathrm{range}(f)),$$

i.e., starting with G, for each clause C of ϕ and literal l of C, we add a clause meaning "l implies the clause of F corresponding to C"; and any clauses of F that are left over get added as-is, without adjoining a literal.

Lemma 2. *Let ϕ be combinable with minimally unsatisfiable (F, G) and $\phi' =$ $\mathrm{combine}(\phi, F, G)$. Then $\forall a \in \{0,1\}^{\mathrm{var}(\phi)}$ $(\phi(a) = 1 \leftrightarrow \phi'|a \notin \mathrm{SAT})$. In other words, for every assignment to the variables of ϕ, $\phi|a$ evaluates to true iff $\phi'|a$ is unsatisfiable.*

Proof. This is simply because in ϕ', after assigning to $\mathrm{var}(\phi)$, each clause $f(C)$ appears in the resulting formula iff some literal $l \in C$ is assigned true. Note that this does not depend on the specific injection $f : \phi \to F$. □

Lemma 3. $\forall k \geq 3$, *given $m \in \mathbb{N}$, one can construct in time* $\mathrm{poly}(m)$, $F \in$ $(k-1)$-CNF *and* $G \in k$-CNF *such that*

– (F, G) *is minimally unsatisfiable*
– $|F| \geq m$
– $|\operatorname{var}(F \wedge G)| \leq \frac{(k-1)^2}{k-2} \lceil m^{\frac{1}{k-1}} \rceil$.

Proof. Let $l = \lceil m^{\frac{1}{k-1}} \rceil$. Consider a $(k-1) \times l$ matrix A of Boolean variables $x_{i,j}$ and the contradictory statements

1. "every row of A contains a 1",
2. "every choice of one entry from each row contains a 0".

We can express 1. by using at most $\frac{k-1}{k-2} l$ new variables $y_{i,j}$ and k-clauses G as follows. For each i, partition $x_{i,1}, \ldots, x_{i,l}$ into $b = \lceil \frac{l}{k-2} \rceil$ blocks of size at most $k-2$. Add to G k-clauses expressing that $y_{i,1}$ is the OR of the variables of the first block; $y_{i,j}$ is the OR of $y_{i,j-1}$ and the variables of the jth block, for $j \in [2, b-1]$; and that the OR of $y_{i,b-1}$ and the variables of the bth block is true. Note that $y = z_1 \vee z_2 \vee \cdots \vee z_l$ can be expressed as an $(l+1)$-CNF since every Boolean function in n variables can be expressed as a CNF where each clause has width at most n. We can express 2. by using a set F of $(k-1)$-clauses where $|F| = l^{k-1} \geq m$. Each clause in F is a disjunction of the negations of the variables in the matrix where each row contributes exactly one variable. F and G have the desired properties. □

Corollary 1. $\forall k, k' \geq 3$, *given* $\phi \in k$-CNF *with* n *variables* $\boldsymbol{x}$ *and* m *clauses, one can construct in time* $\operatorname{poly}(n)$, $\phi' \in k'$-CNF *with* n *variables* $\boldsymbol{x}$ *and* $\frac{(k'-1)^2}{k'-2} \lceil m^{\frac{1}{k'-1}} \rceil$ *variables* $\boldsymbol{y}$ *such that* $\forall a \in \{0,1\}^{\operatorname{var}(\phi)}$ $(\phi(a) = 1 \leftrightarrow \phi'|a \notin \text{SAT})$. *In particular,* $\phi \in \text{SAT}$ *iff* $\forall \boldsymbol{x} \exists \boldsymbol{y}\, \phi'$ *is false.*

Proof. Use lemma 3 to construct minimally unsatisfiable (F, G) with $F \in (k'-1)$-CNF, $G \in k'$-CNF, $|F| = m$, $|\operatorname{var}(F \wedge G)| \leq \frac{(k'-1)^2}{k'-2} \lceil m^{\frac{1}{k'-1}} \rceil$, and $\operatorname{var}(\phi) \cap \operatorname{var}(F \wedge G) = \emptyset$. Let $\phi' = \operatorname{combine}(\phi, F, G)$ and apply lemma 2. □

Theorem 1. $s(\infty) \leq s(\Pi_2 3\text{-SAT})$.

Proof. Fix k and $\epsilon > 0$. We will show that $s(k) \leq s(\Pi_2 3\text{-SAT}) + 4\epsilon$. Let ϕ be a k-CNF with n variables. Use Lemma 1 to sparsify ϕ into at most $2^{\epsilon n}$ many subformulas ϕ_i so that each has at most $m = \lfloor \frac{\epsilon n}{4} \rfloor^2$ clauses for all sufficiently large n. By corollary 1, with $k' = 3$, for each ϕ_i, we can construct in time $\operatorname{poly}(n)$ a $\psi_i \in \Pi_2 3$-CNF with $(1+\epsilon)n$ variables such that $\phi_i \in \text{SAT}$ iff ψ_i is false. So $\phi \in \text{SAT}$ iff ψ_i is false for some i. Evaluating each ψ_i with some $\Pi_2 3$-SAT solver with exponential complexity $\leq s(\Pi_2 3\text{-SAT}) + \epsilon$, we see that

$$s(k) \leq (s(\Pi_2 3\text{-SAT}) + \epsilon)(1+\epsilon) + \epsilon \leq s(\Pi_2 3\text{-SAT}) + 4\epsilon. \qquad \square$$

The previous result puts us in an interesting situation. [IP01] showed that **ETH**, namely that $s(3) > 0$, implies $s(k)$ increases infinitely often as a function of k. Consider the following much weaker analogue.

Conjecture 1. **ETH** $\Rightarrow \exists i \geq 2, k \geq 3 \qquad s(\Pi_i k\text{-SAT}) < s(\Pi_{i+1} k+1\text{-SAT})$.

Conjecture 1, together with Theorem 1, would imply that $s(\infty) < 1$. To see this, assume that for some i and k, $s(\Pi_i k\text{-SAT}) < s(\Pi_{i+1} k+1\text{-SAT})$. Since $s(\Pi_{i+1} k+1\text{-SAT}) \leq 1$, it follows that $s(\Pi_2 3\text{-SAT}) \leq s(\Pi_i k\text{-SAT}) < 1$, which implies $s(\infty) < 1$ by Theorem 1.

3 Algorithms for Two Special Cases

In this section, we show that Quantified CNF-SAT has a poly time algorithm if each clause has at most one existential literal and that $\Pi_2 3$-SAT has a nontrivial algorithm if each clause has at least one universal literal. The purpose of such theorems is to find where the hard cases of $\Pi_i k$-SAT lie.

In a quantified CNF, we say that two clauses A, B *disagree* on variable x iff one of A, B contains x and the other contains $\neg x$. A, B are *resolvable* iff they differ in exactly 1 variable x, and the *resolvent* is $\text{resolve}(A, B) = A \cup B - \{x, \neg x\}$. Also, define $\text{eliminate}(A)$ to be A after removing any universal literal l of A for which no existential literal l' of A occurs after l in the order of quantification. A, B are *Q-resolvable* iff they are resolvable on an existential variable, and the *Q-resolution* operation is $\text{qresolve}(A, B) = \text{eliminate}(\text{resolve}(A, B))$. Büning, Karpinski and Flögel [BKF95] introduced the Q-resolution proof system and showed that it is sound and complete, in the sense that a quantified CNF is false iff the empty clause can be derived by first replacing each clause C by $\text{eliminate}(C)$ and then repeatedly applying qresolve to generate new clauses.

Theorem 2. *Let $\psi = \forall \boldsymbol{x_1} \, \exists \boldsymbol{x_2} \, \cdots \forall \boldsymbol{x_n} \, \phi$ be a quantified Boolean formula where each clause of $\phi \in$ CNF has at most one existential literal. Then ψ can be evaluated in polynomial time.*

Proof. Since each clause contains at most one existential literal, any application of the Q-resolution operator will produce the empty clause. Letting ψ' be ψ after replacing each clause C by $\text{eliminate}(C)$, ψ is true iff ψ' does not contain the empty clause and no two clauses are Q-resolvable in ψ'. □

Theorem 3. *Let $\psi = \forall \boldsymbol{x} \, \exists \boldsymbol{y} \, \phi$, where $\phi \in$ 3-CNF. If each clause of ϕ has at least one universal literal, then ψ can be evaluated in time 2^{cn} for some $c < 1$.*

Proof. Let $G = (V, E)$ where

$$V = \{\text{existential literals of } \phi\}$$
$$E = \{(a, b) \text{ labeled with } c \mid (\bar{a} \vee b \vee \bar{c}) \text{ is a clause of } \phi \text{ and } a, b \text{ are existential and } c \text{ is universal}\}.$$

While we define the graph G assuming that each clause has exactly two existential literals and one universal literal, other cases can be handled as well. For example, if $(a \vee b \vee c)$ is a clause with a as the sole existential variable and the remaining two literals $\bar{b}$ and $\bar{c}$ are universal, we introduce the edge $(\bar{a}, a)$ with the label $b \wedge c$. 2-clauses can be treated similarly. Branching rules used in

[Wil02] can also be applied to reduce the fomulas with these special types of clauses (clauses with more than one universal literal or 2-clauses) to formulas with clauses containing exactly one universal literal and two existential literals.

A *consistent path* in G is a path such that no two edge labels disagree, a *consistent cycle* is a consistent path that is a cycle. Consistency of a path can be checked in polynomial time since all labels are products of literals. Then ψ is false iff there is an existential variable z such that there is a consistent cycle in G containing nodes $z, \bar{z}$.

Notice that a simple cycle containing nodes $z, \bar{z}$ cannot use more edge labels than $4\epsilon n$, where ϵn is the number of existential variables. So we can test whether there is a consistent cycle in G containing $z, \bar{z}$ in time at most $\text{poly}(|\psi|)\binom{2n}{4\epsilon n} \leq \text{poly}(|\psi|)2^{2h(2\epsilon)n}$ where h is the binary entropy function.

If ϵ is large, we can exhaustively search over all settings of the universal variables and then use a 3-SAT algorithm on the rest. If our 3-SAT algorithm runs in time $2^{s(3)n}$, then the combined algorithm runs in time at most $\min\{2^{(1-\epsilon)n+\epsilon s(3)n}, 2^{2h(2\epsilon)n}\}$, which is maximized at about $\epsilon = .05$ for $s(3) = .403$ and yields a run time of at most $2^{.97n}$. If $s(3)$ is smaller, this value would be even smaller. Also, there is already an algorithm for 3-SAT with $s(3) \leq .403$ [Rol05]. □

It is unlikely that we can find a polynomial time algorithm for the variant where each clause has at least one universal literal because of the following.

Theorem 4. *The language of true* $\Pi_2$3-CNF *formulas where each clause has at least one universal literal is* coNP*-complete.*

Proof. The language is in coNP because a witness for falsehood is a consistent cycle containing some existential variable z and its negation $\bar{z}$. To show coNP-hardness, we reduce from 3-UNSAT. Let $\phi \in$ 3-CNF have n variables $\boldsymbol{x}$ and m clauses. Assume without loss of generality that m is even, otherwise, just repeat some clause twice. Let $n' = \frac{m}{2}$, $\boldsymbol{y} = (y_1, \ldots, y_{n'})$ be new variables, and consider the following contradictory 2-CNF:

$$F = \{(y_i \to y_{i+1}), (y_{i+1} \to y_i) \mid i \in [n'-1]\} \cup \{(y_{n'} \to \bar{y}_1), (\bar{y}_1 \to y_{n'})\}.$$

Then the pair of formulas $(F, \emptyset)$ is minimally unsatisfiable and $|F| = m$. By Lemma 2, $\psi = \forall \boldsymbol{x} \exists \boldsymbol{y}$ combine$(\phi, F, \emptyset)$ is a $\Pi_2$3-CNF with at least one universal variable in each clause and such that $\phi \in$ SAT iff ψ is false. □

The coNP-hardness of theorem 4 also follows from [Che09], proposition 3.2, if one observes that the number of universal variables in each constraint constructed there is 1.

4 Reduction to a Canonical Form

Theorems 1 and 3 together suggest that pure existential clauses (those with only existentially quantified literals) and clauses with a mixture of universal and existential literals are both essential to make an instance of $\Pi_2$3-SAT hard. Below we present further evidence of this.

Let $\Pi_2$3-SAT′ be the special case of $\Pi_2$3-SAT where each clause is one of just two types:

(1) one universal literal and two existential literals
(2) pure existential.

We then obtain the following theorem, which uses the same idea as in [Wil02] and follows from the algorithm therein.

Theorem 5. $s(\Pi_2 3\text{-SAT}') < 1 \Rightarrow s(\Pi_2 3\text{-SAT}) < 1$.

Proof. Let A be an algorithm with run time $\leq O(2^{cn})$ with $c < 1$ for $\Pi_2$3-SAT′. Given a $\Pi_2$3-CNF, do this: while there is a clause other than types (1) or (2), i.e., a clause C with at least one universal variable and at most one existential variable, reject if there is no existential variable in C, but otherwise branch on the universal variables. One of the branches will force the existential variable. Each formula at a leaf has only clauses of type (1) or (2), so apply algorithm A. This solves the general case in time at most $O(2^{dn})$ where $d = \max\{c, \frac{\lg 7}{3}\} < 1$. □

We may also assume, without loss of generality, that in a Π_2k-SAT the number of existential variables is $o(n)$, since otherwise we could branch on every possible setting of the universal variables and then invoke a k-SAT solver to obtain a nontrivial algorithm.

5 Parameter Trade-Off at Higher Levels of the Hierarchy

In the next theorem, we show that when confronted with a $\Pi_i k$-SAT instance, one may be able to reduce k to $k' < k$ if one is willing to increase i by 1, and the input clause density is not too high. If m is a function, let $\Sigma_i k\text{-SAT}_m$ ($\Pi_i k\text{-SAT}_m$) be $\Sigma_i k$-SAT ($\Pi_i k$-SAT) but where the number of clauses is promised to be at most $m(n)$.

Theorem 6. *Let $k, k' \geq 3, m \in o(n^{k'-1})$. $\forall$ odd $i \geq 1$,*

$$s(\Sigma_i k\text{-SAT}_m) \leq s(\Pi_{i+1} k'\text{-SAT})$$
$$s(\Pi_i k\text{-SAT}_m) \leq s(\Sigma_i k'\text{-SAT})$$

(which is uninteresting for $i = 1$ since $\Pi_1 k$-SAT is in P). $\forall$ even $i \geq 2$,

$$s(\Sigma_i k\text{-SAT}_m) \leq s(\Pi_i k'\text{-SAT})$$
$$s(\Pi_i k\text{-SAT}_m) \leq s(\Sigma_{i+1} k'\text{-SAT})$$

Proof. Consider a quantified formula $Q_1\boldsymbol{x_1} \cdots Q_i\boldsymbol{x_i}\ \phi$ where each Q_j is a quantifier, each $\boldsymbol{x_j}$ is a tuple of Boolean variables, and ϕ is a k-CNF with n variables $\boldsymbol{x}$ and $\leq m(n)$ clauses. Using corollary 1, $\forall \epsilon > 0$ and for sufficiently large n, we can construct in polynomial time a k'-CNF ϕ' such that

- ϕ' has $(1+\epsilon)n$ vars: $\boldsymbol{x}$ and ϵn new ones $\boldsymbol{y}$
- for each assignment a to $\boldsymbol{x}$, $\phi(a) = 1$ iff $\phi'|a \notin$ SAT.

So

$$\begin{aligned} &Q_1\boldsymbol{x_1}\cdots Q_i\boldsymbol{x_i}\ \phi \\ \Leftrightarrow &Q_1\boldsymbol{x_1}\cdots Q_i\boldsymbol{x_i}\ \neg\exists\boldsymbol{y}\ \phi' \\ \Leftrightarrow &\neg Q_1'\boldsymbol{x_1}\cdots Q_i'\boldsymbol{x_i}\exists\boldsymbol{y}\ \phi' \end{aligned}$$

where Q_j' is existential iff Q_j is universal. □

It would be nice to eliminate the requirement that ϕ have at most $o(n^{k'-1})$ clauses. One might try to do this by first sparsifying ϕ, but the possibly complex quantification of the variables prevents this.

Acknowledgments. We thank the reviewers for their helpful comments.

References

[APT79] Aspvall, B., Plass, M.F., Tarjan, R.E.: A linear-time algorithm for testing the truth of certain quantified boolean formulas. Inf. Process. Lett. 8(3), 121–123 (1979)

[BKF95] Büning, H.K., Karpinski, M., Flögel, A.: Resolution for quantified boolean formulas. Information and Computation 117(1), 12–18 (1995)

[Che09] Chen, H.: Existentially restricted quantified constraint satisfaction. Information and Computation 207(3), 369–388 (2009)

[CIKP08] Calabro, C., Impagliazzo, R., Kabanets, V., Paturi, R.: The complexity of unique k-SAT: An isolation lemma for k-CNFs. Journal of Computer and Systems Sciences 74(3), 386–393 (2008); Preliminary version in Proceedings of the Eighteenth IEEE Conference on Computational Complexity, pp. 135–144 (2003)

[CIP06] Calabro, C., Impagliazzo, R., Paturi, R.: A duality between clause width and clause density for SAT. In: CCC 2006: Proceedings of the 21st Annual IEEE Conference on Computational Complexity, Washington, DC, USA, pp. 252–260. IEEE Computer Society, Los Alamitos (2006)

[CIP09] Calabro, C., Impagliazzo, R., Paturi, R.: The complexity of satisfiability of small depth circuits. In: Parameterized and Exact Computation: 4th International Workshop, IWPEC 2009, Revised Selected Papers, Copenhagen, Denmark, September 10-11, pp. 75–85. Springer, Heidelberg (2009)

[DW05] Dantsin, E., Wolpert, A.: An improved upper bound for SAT. In: Bacchus, F., Walsh, T. (eds.) SAT 2005. LNCS, vol. 3569, pp. 400–407. Springer, Heidelberg (2005)

[IP01] Impagliazzo, R., Paturi, R.: The complexity of k-SAT. Journal of Computer and Systems Sciences 62(2), 367–375 (2001); Preliminary version in 14th Annual IEEE Conference on Computational Complexity, pp. 237–240 (1999)

[IPZ98] Impagliazzo, R., Paturi, R., Zane, F.: Which problems have strongly exponential complexity? Journal of Computer and System Sciences 63, 512–530 (1998); Preliminary version in 39th Annual IEEE Symposium on Foundations of Computer Science, pp 653–662 (1998)

[PPSZ05] Paturi, R., Pudlák, P., Saks, M.E., Zane, F.: An improved exponential-time algorithm for k-SAT. Journal of the ACM 52(3), 337–364 (2005); Preliminary version in 39th Annual IEEE Symposium on Foundations of Computer Science, pp. 628–637 (1998)

[PPZ99] Paturi, R., Pudlák, P., Zane, F.: Satisfiability coding lemma. Chicago Journal of Theoretical Computer Science (December 1999); Preliminary version in 38th Annual Symposium on Foundations of Computer Science, pp. 566–574 (1997)

[Rol05] Rolf, D.: Improved bound for the PPSZ/Schöning-algorithm for 3-SAT. Electronic Colloquium on Computational Complexity (ECCC) 12(159) (2005)

[Sch99] Schöning, U.: A probabilistic algorithm for k-SAT and constraint satisfaction problems. In: FOCS, pp. 410–414 (1999)

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

[Wil02] Williams, R.: Algorithms for quantified boolean formulas. In: Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 299–307 (2002)

Lecture Notes in Computer Science 6478

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Venkatesh Raman   Saket Saurabh (Eds.)

Parameterized and Exact Computation

5th International Symposium, IPEC 2010
Chennai, India, December 13-15, 2010
Proceedings

Springer

Volume Editors

Venkatesh Raman
The Institute of Mathematical Sciences, Chennai, 600113, India
E-mail: vraman@imsc.res.in

Saket Saurabh
The Institute of Mathematical Sciences, Chennai 600113, India
E-mail: saket@imsc.res.in

Library of Congress Control Number: 2010939810

CR Subject Classification (1998): F.2, G.1-2, I.3.5, F.1, G.4, E.1, I.2.8

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-642-17492-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-17492-6 Springer Berlin Heidelberg New York

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Lecture Notes in Computer Science 6478

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Venkatesh Raman   Saket Saurabh (Eds.)

Parameterized and Exact Computation

5th International Symposium, IPEC 2010
Chennai, India, December 13-15, 2010
Proceedings

Springer

Volume Editors

Venkatesh Raman
The Institute of Mathematical Sciences, Chennai, 600113, India
E-mail: vraman@imsc.res.in

Saket Saurabh
The Institute of Mathematical Sciences, Chennai 600113, India
E-mail: saket@imsc.res.in

Library of Congress Control Number: 2010939810

CR Subject Classification (1998): F.2, G.1-2, I.3.5, F.1, G.4, E.1, I.2.8

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-642-17492-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-17492-6 Springer Berlin Heidelberg New York

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

The International Symposium on Parameterized and Exact Computation (IPEC, formerly IWPEC) is an international symposium series that covers research in all aspects of parameterized and exact algorithms and complexity. Started in 2004 as a biennial workshop, it became an annual event from 2008.

The four previous meetings of the IPEC/IWPEC series were held in Bergen, Norway (2004), Zürich, Switzerland (2006), Victoria, Canada (2008) and Copenhagen, Denmark (2009). On recommendations of the Steering Committee, from this year, the word 'symposium' replaces the word 'workshop' in the name, and it gets a new abbreviation IPEC (rhyming with the old one IWPEC).

IPEC 2010 was the fifth symposium in the series, held in Chennai, India, during December 13-15, 2010. The symposium was co-located with the 30th Foundations of Software Technology and Theoretical Computer Science conference (FSTTCS 2010), a premier theory conference in India.

At IPEC 2010, we had three plenary speakers: Anuj Dawar (University of Cambridge, UK), Fedor V. Fomin (University of Bergen, Norway) and Toby Walsh (NICTA and University of New South Wales, Australia). The abstracts accompanying their talks are included in these proceedings. We thank the speakers for accepting our invitation and for their abstracts.

In response to the call for papers, 32 papers were submitted. Each submission was reviewed by at least four reviewers. The reviewers were either Program Committee members or invited external reviewers. The Program Committee held electronic meetings using the EasyChair system, went through extensive discussions, and selected 19 of the submissions for presentation at the symposium and inclusion in this LNCS volume.

From this year IPEC also started the tradition of awarding excellent student paper awards. This year the award was shared by two papers, "Small Vertex Cover Makes Petri Net Coverability and Boundedness Easier" by M. Praveen and "Inclusion/Exclusion Branching for Partial Dominating Set and Set Splitting" by Jesper Nederlof and Johan M. M. van Rooij. We thank Frances Rosamond for sponsoring the award.

We are very grateful to the Program Committee, and the external reviewers they called on, for the hard work and expertise which they brought to the difficult selection process. We also wish to thank all the authors who submitted their work for our consideration. Special thanks go to Meena Mahajan and her team for the local organization of IPEC 2010. We thank, in particular, Neeldhara Misra and Geevarghese Philip for help in maintaining the website of IPEC 2010.

Finally, we thank the Institute of Mathematical Sciences for the financial support, its staff for the organizational support, and the editors at Springer for their cooperation throughout the preparation of these proceedings.

December 2010

Venkatesh Raman
Saket Saurabh

Conference Organization

Programe Committee

Rod Downey	Victoria University of Wellington, New Zealand
Mike Fellows	University of Newcastle, Australia
Martin Fürer	The Pennsylvania State University, USA
Serge Gaspers	CMM Universidad de Chile, Chile
Jiong Guo	Max Planck Institute, Germany
Gregory Gutin	Royal Holloway University of London, UK
Petr Hlinený	Masaryk University Brno, Czech Republic
Falk Hüffner	Humboldt University of Berlin, Germany
Petteri Kaski	Helsinki Institute for Information Technology, Finland
Dieter Kratsch	University of Metz, France
Daniel Lokshtanov	University of Bergen, Norway
Dániel Marx	Tel Aviv University, Israel
Ramamohan Paturi	University of California San Diego, USA
Christophe Paul	LIRMM, University of Montpellier, France
Venkatesh Raman	Institute of Mathematical Sciences, India (Co-chair)
Peter Rossmanith	RWTH Aachen University, Germany
Saket Saurabh	Institute of Mathematical Sciences, India (Co-chair)
C. R. Subramanian	Institute of Mathematical Sciences, India
Dimitrios M. Thilikos	National and Kapodistrian University of Athens, Greece
Ryan Williams	IBM Almaden Research Center, USA

Steering Committee

Hans Bodlaender	Utrecht University, The Netherlands (Chair)
Jianer Chen	Texas A&M University, USA
Mike Fellows	University of Newcastle, Australia
Fedor V. Fomin	University of Bergen, Norway
Martin Grohe	Humboldt University, Germany
Michael Langston	University of Tennessee, USA
Dániel Marx	Humboldt University, Germany
Venkatesh Raman	Institute of Mathematical Sciences, India
Saket Saurabh	Institute of Mathematical Sciences, India

IPEC Publicity Chair

Frances Rosamond	University of Newcastle, Australia

Local Organization

K. Narayan Kumar	Chennai Mathematical Institute, India
Meena Mahajan	Institute of Mathematical Sciences, India (Chair)
R. Ramanujam	Institute of Mathematical Sciences, India
Saket Saurabh	Institute of Mathematical Sciences, India
Vikram Sharama	Institute of Mathematical Sciences, India

External Reviewers

Anders Yeo
Anthony Perez
Archontia Giannopoulou
Catherine Mccartin
Christian Komusiewicz
David Bryant
Dimitris Zoros
Gabor Erdelyi
Geevarghese Philip
Geoff Whittle
George B. Mertzios
Holger Dell
Hubie Chen
Ignasi Sau
Ildikó Schlotter
Isolde Adler
Iyad Kanj
Jan Arne Telle
Jeremiah Blocki
Jianer Chen
Klaus Reinhardt
Magnus Wahlström
Marcin Kaminski
Marek Cygan
Mark Jones
Mathieu Liedloff
Matthias Mnich
Matthias Killat
Meena Mahajan
Michael Lampis
N.R. Aravind
N.S. Narayanaswamy
Pascal Schweitzer
Petr Golovach
Piotr Faliszewski
René van Bevern
Robert Bredereck
Robert Ganian
Roland Meyer
Sharon Bruckner
Somnath Sikdar
Stefan Kratsch
Thore Husfeldt
Valia Mitsou
Virginia Vassilevska Williams
William Gasarch
Yixin Cao
Yngve Villanger

Table of Contents