On Complexity gaps for Resolution-based proof systems

Stefan Dantchev[1] and Søren Riis[2]
[1]Dept. of Mathematics and Computer Science, University of Leicester
dantchev@mcs.le.ac.uk
[2]Dept. of Computer Science, Queen Mary, University of London
smriis@dcs.qmw.ac.uk

26th November 2002

Abstract

We study the proof complexity of $Taut$, the class of Second-Order Existential (SO$\exists$) logical sentences which fail in all finite models. The Complexity-Gap theorem for Tree-like Resolution says that the shortest Tree-like Resolution refutation of any such sentence Φ is either fully exponential, $2^{\Omega(n)}$, or polynomial, $n^{O(1)}$, where n is the size of the finite model. Moreover, there is a very simple model-theoretics criteria which separates the two cases: the exponential lower bound holds if and only if Φ holds in some infinite model.

In the present paper we prove several generalisations and extensions of the Complexity-Gap theorem.

1. The gap for stronger systems, Res$^*(k)$, is between polynomial and $\exp\left(\Omega\left(\frac{\log k}{k}n\right)\right)$ for every k, $1 \leq k \leq n$. Res$^*(k)$ is an extension of Tree-like Resolution, in which literals are replaced by terms (i.e. conjunctions of literals) of size at most k. The lower bound is tight.

2. For a natural subclass of $Taut$, $Rel\,(Taut)$, there is a gap between polynomial Tree-like Resolution proofs and sub-exponential, $2^{\Omega(n^{\varepsilon})}$, general (DAG-like) Resolution proofs, whilst the separating criteria is the same as before. $Rel\,(Taut)$ is the set of all sentences in $Taut$, relativised with respect to a unary predicate.

3. There is (as expected) no gap for any propositional proof system (including Tree-like Resolution) if we enrich the language of SO logic by a built-in order.

1 Introduction

In [10] a new kind of results for propositional logic was introduced. Expressed somewhat informally, it was shown that any sequence ψ_n of tautologies which expresses the validity of a fixed combinatorial principle either is "easy" i.e. has polynomial size tree-resolution proofs or is "difficult" i.e requires full exponential size tree-resolution proofs. It was shown that the class of tautologies which are hard (for tree-resolution) is identical to the class of tautologies which are based on combinatorial principles which are violated for infinite sets.

According to this result the proof complexity of a combinatorial principle never have intermediate growth rates like for example $2^{\log^k(n)}, k = 2, 3, \ldots$. In this paper we extend this result to a number of related resolution-based systems.

A central question in the theory of proof complexity concerns to the amount of resources (usually proof length) which is needed to prove a certain sequence of tautologies. Usually, the sequence of tautologies consists of Tautologies which are *similar* except for their size. The paper is organised as follows:

Firstly we consider $Res(k)$ which is similar but stronger than resolution. In $Res(k)$ clauses i.e. disjunctions, are replaced by disjunctions of conjunctions of $\leq k$ literals. The rules are strengthened so one can resolve not just a single variable (like in resolution), but also conjunctions of up to k variables. An easy extension of [10] gives a complexity jump from polynomial to $\exp(\Omega(n/k))$ for these problems (when proofs are represented as trees). This lower bound is also implicit in [8] by Krajicek. We improve this, and show that the jump is from polynomial to $\exp(\Omega(n \log(k)/k))$.

Secondly, we consider the resolution proof system in a setting of DAG-like proofs rather than tree-like structure. In a DAG-like proof a once derived disjunction can be used any number of times later in the proof. This cannot happen in the tree like cases. Thus a given tautology might have a DAG-like proof which is substantially shorter than the shortest tree-like proofs [2], [3].

The class of combinatorial problems which are hard for

DAG-like resolution differs from the class of combinatorial problems which are hard for tree like resolution. "Minimal element" is a principle separates the two systems [2], [3].

In this paper we show the class of combinatorial problems which are hard for DAG resolution is identical to the class which is hard for tree resolution *provided the combinatorial principles are being relativised.* It is already known that minimal element, which has polynomial size DAG-like proofs, require exponential size proofs when this problem is relativised [4].

Finally we show that there is no complexity gap if we enrich the language of SO logic by a built-in order. Even though this result is expected it has a number of less obvious consequences. It allows us to answer an open question from [5] by showing that there is no complexity gap for tree resolution above $2^{n\log(n)}$. We expect that there is a complexity gap above 2^n, but are not sure if the gap jumps the whole way up to $2^{nlog(n)}$. If we consider the "cost" of a proof to be the length of the longest branch (rather than the number of symbols/steps in the proof) it also follows that there is no branch-complexity gap above n^3. There is a rather obvious gap in branch length complexity from constant to n. We expect there to be a gap in branch length complexity above n, but are not sure if the gap jumps the whole way up to n^3.

2 Preliminaries

2.1 $Res(k)$

In this section we recall some of the basic concepts related to resolution proofs.

A *literal* is a propositional variable or the negation of a propositional variable. A k-conjunction $\wedge_j l_j$ is a conjunction of at most k literals. A k DNF is a disjunction of k-conjunctions.

The k-Resolution proof system is a refutation system designed to provide certificates (i.e. proofs) that a system of k DNF's is unsatisfiable. This is done by means of the following four derivation rules. The $\wedge$-*introduction rule* is

$$\frac{C_1 \vee \bigwedge_{j\in J_1} l_j \quad C_2 \vee \bigwedge_{j\in J_2} l_j}{C_1 \vee C_2 \bigwedge_{j\in J_1\cup J_2} l_j},$$

provided that $|J_1 \cup J_2| \leq k$. The *cut (or resolution) rule* is

$$\frac{C_1 \vee \bigvee_{j\in J} l_j \quad C_2 \vee \bigwedge_{j\in J} \neg l_j}{C_1 \cup C_2}.$$

The two *weakening rules* are

$$\frac{C}{C \vee \bigwedge_{j\in J} l_j},$$

provided that $|J| \leq k$, and

$$\frac{C \vee \bigwedge_{j\in J_1\cup J_2} l_j}{C \vee \bigwedge_{j\in J_1} l_j}.$$

Here C's are d-DNFs, and l's are literals.

The given clauses are often referred to as *axioms*, and the task is to derive the empty clause (the contradiction) from the axioms. In *tree-resolution* the proof is organised as a binary tree with the axioms in the leaves and the empty clause in the root. In *DAG-resolution* (or just *resolution*) the proof is given as a linear sequence $C_1, C_2, \dots, C_u$ of clauses, where each clause either is an axiom or can be obtained by means of the resolution rule (applied to two already derived clauses). In a resolution proof, clauses can be reused more the once. A tree-resolution proof do not allow this.

2.2 Translating formulas from predicate logic into propositional logic

Let L denote a first order language. To keep things slightly simplified we assume L contains only relations (constants and function symbols can be defined using relations anyway). We use the standard definition from model theory where a model of L is a set M_{set} on which each r-ary relation R is interpreted as a subset $\tilde{R} \subseteq M^k$. The size of the model is given by the cardinality of M_{set}. If for example $R(x,y)$ is a binary relation, a model of size n is a digraph on n vertex. In general a model is a hyper-graph with a finite set of edge types. The size n of the model (hyper-graph) might be infinite. Given n points (vertex). For each potential hyper-edge we introduce a boolean variable. Assume for example we consider a relational language which contains two binary relations (say, $R(x,y)$ and $S(x,y)$) and one ternary relation (say $T(x,y,z)$). This language is the language for hyper-graphs with two types (colours) for each ordinary edge as well one type for each 3-ary hyper-edge. In this example we introduce boolean variables r_{ij}, s_{ij} and t_{ijk} where $i,j,k \in \{1,2,\dots,n\}$. Let $\mathrm{Var}_n(L)$ denote the collection of boolean variables introduced this way. Notice that any concrete model of L (i.e. any concrete hyper-graph) corresponds to a specific assignment of the boolean variables.

There is a straightforward way of translating a purely relational CNF formula into a propositional CNF (one for each n). The formula $T(x,y,z) \wedge (R(x,y) \vee \neg S(y,z))$ translate into $\bigwedge_{i,j,k}^n t_{ijk} \wedge \bigwedge_{i,j,k}^n (r_{ij} \vee \neg s_{jk})$. Pure existential formulas can also be translated into a propositional CNF by replacing the existential quantifier with a disjunction. Thus for example $\exists y R(x,y)$ becomes $\bigwedge_i^n \bigvee_j^n r_{ij}$.

A tautology Ψ in the variables $\mathrm{Var}_n(L)$ express a fact about hyper-graphs of size n. We will mainly consider properties of hyper graphs, which are independent of the labelling of the hyper-graph. A *combinatorial principle* of hyper-graphs is a property, which hold for all hyper-graphs (of a specific type). Now since any such property is preserved under re-labelling of the vertex, it is natural to focus on properties, which are defined explicitly so they remain invariant under re-labelling of the vertex. In other words it is natural to focus on properties which are (explicitly) invariant under the natural action of the symmetric group S_n. In this paper we consider resolution based propositional systems. We say that a collection $C_1, C_2, \ldots, C_u$ of disjunctions (clauses) is invariant under the action of S_n if for each C_j with $j \in \{1, 2, \ldots, u\}$ and each $\pi \in S_n$ we have $\pi(C_j) = C_k$ for some $k \in \{1, 2, \ldots, u\}$. We say a CNF is invariant under the action of S_n if its collection of disjunctions (clauses) is invariant under S_n.

For $A \subseteq \{1, 2, \ldots, n\}$ we say the disjunction C is invariant under S_A, if $\pi(C) = C$ for each permutation $\pi \in S_A$ (i.e. C is invariant under permutations which only moves points in A). We need the following lemma:

Lemma 2.1 *Assume C is invariant under S_A as well as invariant under S_B. Assume $A \cap B \neq \emptyset$. Then C is invariant under $S_{A \cup B}$.*

Proof: To see this it suffice to show $(a, b)C = C$ for $a \in A$ and $b \in B$. Pick $c \in A \cap B$. Then since $(ab) = (ac)(bc)(ac)$ we have $(ab)C = (ac)(bc)(ac)C = (ac)(bc)C = (ac)C = C$ since C is invariant under $(ac) \in S_A$ as well as $(bc) \in S_B$. □

Consider a disjunction C. Assume C is invariant under A. In that case we say C has support contained in A^c. We define the *support size* of C as $n - |A|$ where A is a set of maximal size such that C is S_A invariant.

Lemma 2.2 *Let L be a fixed relational language. Assume Ψ_n is a sequence of CNFs in the variables $\mathrm{Var}_n(L)$. Assume there is a polynomial P such that the number of distinct disjunctions in Ψ_n is bounded by $P(n)$. Then there exists a constant v such that for any n and any clause $C \in \Psi_n$, C has support size bounded by v.*

Proof: Let C be any clause from the CNF. We want to show that C must have constant size support (i.e. support size bound by a number independent of n). The strategy to proof this is to show that if C has non-constant support size, then $\{\pi(C) : \pi \in S_n\}$ contains to many distinct clauses. Without loss of generality assume C has support $\{1, 2, \ldots, w\}$ for some w minimal with this property. We consider two distinct cases:

case 1: $w < n/2$. Consider two distinct sets $A = \{a_1, a_2, \ldots, a_w\}$ and $B = \{b_1, b_2, \ldots, b_w\}$ each containing w elements. Let $C_A := (1, a_1)(2, a_2) \ldots (w, a_w)C$ and let $C_B := (1, b_1)(2, b_2) \ldots (w, b_w)C$. We claim $C_A \neq C_B$. To see this assume $C_A = C_B$. Now C_A is A^c invariant and C_B is B^c invariant. Now since $w < n/2$ clearly $A^c \cap B^c \neq \emptyset$. According the lemma 2.1 C_A (=C_B) is invariant under $A^c \cup B^c$ and thus C_A has support contained in $A \cap B$. Thus support size of C_A is $w \leq |A \cup B| < w$ which is a contradiction. The number of elements in $\{\pi(C) : \pi \in S_n\}$ is thus at least as large as the number of distinct w-element subsets of n.

case 2: $w \geq n/2$. We want to show that the set $\{\pi(C) : \pi \in S_n\}$ is too large. The idea is to map the disjunction C into a clause $C?$ with support in $\{1, 2, \ldots, n/2 - 1\}$. This is done by taking the union of all disjunctions $\pi(C)$ where $\pi \in S_{\{n/2, n/2+1, \ldots, n\}}$. This map naturally extend to a map, mapping the collection $\{\pi(C) : \pi \in S_n\}$ of clauses onto the collection $\{\pi(C?) : \pi \in S_n\}$. The map is usually not one-to-one, but since it is onto the number of distinct elements in $\{\pi(C) : \pi \in S_n\}$ is at least as big as the number of distinct elements in $\{\pi(C?) : \pi \in S_n\}$. This latter number equals the number of distinct $n/2 - 1$ element subsets of $\{1, 2, \ldots, n\}$. □

Proposition 2.3 *For each relational language L and for each polynomial P there is a constant $c = c_{L,P}$ such that for any n the number of distinct S_n-invariant CNF's of size $\leq P(n)$ is at most c.*

Proof: According the lemma 2.2 each CNF is generated by a finite number of clauses with support in $\{1, 2, \ldots, w\}$ for some constant w which can be chosen independent of n. The number of distinct clauses with support in $\{1, 2, \ldots, w\}$ is bound by a constant $c_{L,P}$. □

Proposition 2.4 *Let Ψ be a S_n invariant CNF in the variables $\mathrm{Var}_n(L)$ and assume each clause has support size $\leq w < n - c$ (were c the maximal arity of a relation in L). Then there is a universal existential relational which translate into Ψ.*

Proof:(outline) Replace each big conjunction with an universal quantifier, and each big disjunction by an existential quantifier. □

Let us illustrate the proposition with an example. Let $\Psi :\equiv \bigwedge_{i,j}(r_{ij} \vee \neg r_{ji}) \wedge \bigwedge_j \bigvee_k r_{jk}$. Each clause has support size ≤ 2. Since Ψ is on CNF form as is invariant under S_n according to the proposition the sentence appears as translation of a relational sentence. The sentence is obviously: $\forall x \forall y(R(x,y) \vee \neg R(y,x)) \wedge \forall x \exists y R(x,y)$.

Up to now one might have got the impression that the power of expression was rather pure. After all we have showed that for any given type of hyper-graph, there are only a constant number of distinct properties which can be expressed by CNFs bounded in size by a fixed polynomial. The situation is in some sense quite different. Any tautology belongs to co-NP so only co-NP properties have a chance to be translated into tautologies. Now any co-NP property of hyper-graphs (of a given fixed type), can be expressed as a $SO\exists$ (i.e. a universal second order sentence) $\forall \vec{R} \eta(\vec{R}, \vec{G})$. This sentence holds for all hyper-graphs of size n and type given by $\vec{G}$ if and only if $\eta(\vec{R}, \vec{G})$ holds for all hyper-graphs of type given by $\vec{R}, \vec{G}$. Thus any co-NP tautology of hyper-graphs of a given type holds if and only if a certain CNF tautology hold for hyper-graphs of a richer type.

A *strict* $SO\forall$ sentence is a universal second order sentence of the form $\forall \vec{R} \vee_i \exists \vec{y} \wedge_j \forall \vec{z} \psi_{ij}(\vec{y}, \vec{z}, \vec{R})$ where each ψ_{ij} is an atomic formula or the negation of an atomic formula. A *strict* $SO\exists$ sentence is an existential second order sentence of the form $\exists \vec{R} \wedge_i \forall \vec{y} \vee_j \exists \vec{z} \psi_{ij}(\vec{y}, \vec{z}, \vec{R})$ where each ψ_{ij} is an atomic formula or the negation of an atomic formula.

We have shown:

Theorem 2.5 *There is a one-to-one correspondence between strict $SO\forall$-sentences and polynomial size S_n-invariant DNFs.*

There is a one-to-one correspondence between strict $SO\exists$-sentences and polynomial size S_n-invariant CNFs.

Notice that a sequence of polynomial size S_n-invariant DNF (CNFs) need not be uniform, by the correspondence appears for each fixed n when n is sufficiently large.

2.3 Relativising combinatorial principles

Let Ψ be a combinatorial principle for hyper-graphs (of a certain type). Informally, the statement Ψ states that some property holds for all graphs on the universe of vertex. Informally, the relativised sentence $\mathrm{rel}(\Psi)$ say that the principle Ψ holds for any subset U of vertex. Since relativised principle not only state the validity of the principle for models of size n but implies that the principles holds for all models of size $\leq n$ the relativised principle is in general harder to prove than its non-relativised counter part.

Formally, we relativise a sentence $\psi \in L$ as follows by adding a new unary relation symbol to the language L and by modifying ψ by systematically replacing each universal quantifier $\forall x$.... with $\forall x \neg U(x) \vee$ ... and by replacing each existential quantifier $\exists x$... with $\exists x U(x) \wedge$..... A second order sentence $Q_1 R_1 Q_2 R_2 \ldots Q_s R_s \Psi$, where each Q_j is a second order quantifier, and where Ψ is a first order sentence, relativise to $\forall U Q_1 R_1 Q_2 R_2 \ldots Q_s R_s \Psi_U$.

Notice that a strict SO$\forall$ (SO$\exists$) sentence relativise to a strict SO$\forall$ (SO$\exists$) sentence.

Notice also that a purely universal sentence of the form

$$\forall x_1 \forall x_2 \ldots \forall x_k R(x_1, x_2, \ldots x_k)$$

relativises into the set of clauses

$$\neg u_{i_1} \vee \neg u_{i_2} \vee \ldots \neg u_{i_k} \vee r_{i_1, i_2, \ldots i_k}$$

for all possible combinations of $i_1, i_2, \ldots i_k$.

For every Skolem relation $s_{i_1, i_2, \ldots i_k, i_0}$ (where i_0 is the witness) we have the clause

$$\neg u_{i_1} \vee \neg u_{i_2} \vee \ldots \neg u_{i_k} \vee \neg s_{i_1, i_2, \ldots i_k, i_0} \vee u_{i_0}$$

i.e. if all the arguments are in U, so is the witness.

2.4 Proving lower bounds for Resolution and Tree-like Res(k)

We will first describe the *search problem*, associated to an *inconsistent set of clauses*, as defined in [7]: Given a truth assignment, find a clause, falsified under the assignment.

We can use a refutation of the given clause to solve the search problem as follows. We first turn around all the edges of the graph of the proof. The contradiction now becomes the only root (source) of the new graph, and the axioms and the initial formulae become the leaves (sinks). We perform a search in the new graph, starting from the root, which is falsified by any assignment, and always going to a vertex which is falsified under the given assignment. Such a vertex always exists as the inference rules are sound. We end up at a leaf, which is one of the initial clauses.

Thus, if we want to prove *existence of a particular kind of clauses in any proof*, we can use an *adversary argument in solving the search problem*. The argument is particularly nice for Resolution as developed by Pudlak in [9]. There are two players, named *Prover* and *Adversary*. An unsatisfiable set of clauses is given. Adversary claims wrongly that there is a satisfying assignment. *Prover* holds a Resolution refutation, and uses it to solve the search problem. A *position* in the game is a partial assignment of the propositional variables. The positions can be viewed as conjunctions of literals. All the possible positions in the game are exactly negations of all the clauses in the Prover's refutation. The game start from the empty position (which corresponds to $\top$, the negation of the empty clause). Prover has two kind of moves:

1. She queries a variable, whose value is unknown in the current position. Adversary answers, and the position then is extended with the answer.

2. She forgets a value of a variable, which is known. The current position is then reduced, i.e., the variable value becomes unknown.

The game is over, when the current partial assignment falsifies one of the clauses. Prover then wins, having shown a contradiction.

We will be interested in *deterministic Adversary's strategies* which allows to prove the existence of certain kind of clauses in a Resolution refutation.

In order to prove *lower bounds on Resolution proofs*, we will use the known technique, "bottleneck counting". It has been introduced by Haken in his seminal paper [6] (for the modern treatment see [1]). We first define the concept of *big clause*. We then design *random restrictions*, so that they "kill" (i.e. evaluate to $\top$) any big clause with high probability (whp). By the union bound, If there are few big clause, there is a restriction which kills them all. We now consider the *restricted set of clauses*, and using *Prover-Adversary game*, show that there has to be *at least one big clause in the restricted proof*, which is a contradiction and completes the argument.

The case of Tree-like proofs, either Resolution or Res(d), is much simpler as a *tree-like proof* of a given set of clause *is equivalent* to a *decision tree, solving the search problem*. We can use pretty straightforward adversary argument against a decision tree, in order to show that it has to have many nodes.

3 Complexity gap for Res* (k)

Theorem 3.1 *A SO$\exists$ sentence Φ is given which fails in all finite models, but holds in some infinite model. Let us denote by φ_n the translation of Φ into propositional logic, assuming a finite model of size n. Then for any k, $2 \le k \le n$, any $R^*(k)$ refutation of φ_n is of size* $\exp\left(\Omega\left(\frac{\log k}{k} n\right)\right)$.

Proof We will describe the modifications of the original proof, Section 4 of [10]. Recall that the proof goes as follows. Prover holds the Tree-like Resolution refutation which is equivalent to a decision tree solving the search problem for φ_n. Adversary's task to force a big subtree in the Prover's tree. In doing so, he uses an infinite model M in which Φ holds. At any stage in the game some elements of the universe are interpreted in M, and it is clear that no contradiction can be achieved by Prover, unless she forces Adversary to interpret all the n elements. When a variable is queried by Prover, Adversary answer as follows.

1. If the truth value can be derived from the current interpretation, i.e. the partial assignment, he replies with the value. The current interpretation is not extended as this was a forced question.

2. If the value cannot be derived from the current interpretation, it follows that both $\top$ and $\bot$ answers are consistent with some extension of it. Thus Adversary is free to choose an answer, and it both possible cases there is a consistent extension of the current interpretation by at most r new elements, where r is a constant, the maximal arity of relation symbols in φ_n.

This shows that Prover's decision tree has to contain a complete binary subtree of height n/r which proves a $2^{n/r}$ lower bound.

Let us now consider the case of $Res^*(k)$ instead of Tree-like Resolution. A Prover's query is a k-disjunction instead of a single variable. Adversary first simplifies the query, using the current interpretation. That is, if a literal evaluates to $\bot$ it vanishes; if all the literals vanish, the query itself is forced, and the answer is $\bot$. If a literal evaluates to $\top$ so does the entire disjunction; the query is forced, and the answer is $\top$.

The non-trivial case is when the query is not forced, i.e. after having been simplified, it can still be answered both $\top$ and $\bot$. For the positive answer it is enough to force a single literal to $\top$,and therefore to interpret at most r new elements. For the negative answer Adversary should force all the literals to $\bot$, and therefore he has to interpret all the mentioned elements which are at most kr.

We will show that a subtree rooted at a given node can be lower-bounded by a function S in the number of free elements at the node. If the number of free elements at the current node is u, the $\top$ successor has at least $u-r$ such elements whilst the $\bot$ successor has at most $u-kr$. Thus we have

$$S(u) \ge S(u-r) + S(u-kr) + 1.$$

We will prove that $S(u) \ge x_k^{u/r} - 1$ where x_k is the biggest positive real root of the equation

$$x^k - x^{k-1} - 1 = 0.$$

The induction step is trivial. By the induction hypothesis we get

$$\begin{aligned} S(u) &\geq \left(x_k^{u/r-1} - 1\right) + \left(x_k^{u/r-k} - 1\right) + 1 \\ &= x_k^{u/r} - 1. \end{aligned}$$

Let us now observe that there are positive constants a and b such that for every $k \geq 1$, x_k is in the interval $\left(1 + a\frac{\ln k}{k}, 1 + b\frac{\ln k}{k}\right)$. Indeed, let us denote $f(x) = x^k - x^{k-1} - 1$. We have $f\left(1 + c\frac{\ln k}{k}\right) = \left(1 + \frac{c \ln k}{k}\right)^{k-1} \frac{c \ln k}{k} - 1$ where c is a constant. Since $\lim_{k \to \infty} \left(1 + \frac{c \ln k}{k}\right)^{k-1} k^{-c} = 1$ we can conclude that for every constant $\varepsilon > 0$ there is k_ε so that for every $k \geq k_\varepsilon$, $f\left(1 + (1-\varepsilon)\frac{\ln k}{k}\right) < 0$ and $f\left(1 + (1+\varepsilon)\frac{\ln k}{k}\right) > 0$.

It is now clear how to get the desired lower bound. At the root of every decision tree we have all the n elements free, so that the decision tree has to be of size at least $\left(1 + a\frac{\ln k}{k}\right)^{n/r} - 1$ which is $\sim e^{\frac{an \ln k}{rk}}$ and therefore $\exp\left(\Omega\left(\frac{\log k}{k} n\right)\right)$. What remains to be checked is the basis case, $n < kr$. In this case the size of the tree is at least n/r (as this is the minimal number of queries required to force interpretation of all the elements), whilst the lower bound expression is $\leq e^{a \ln k} = k^a$. Clearly $n/r > k^a$ for big enough n as $k \leq n$ and r, a, $a < 1$ are constants independent from n. This completes the proof. □

It is important to note that the lower bound, we have proven, is tight. The SO∃ sentence, which shows this, is *Minimal Element Principle* (MEP_n), saying that *a finite* (partially) *ordered* n-element *set* has a *minimal element*. Its negation is

$$\begin{aligned} \exists P \quad & ((\forall x \, \neg P(x,x)) \wedge \\ & (\forall x, y, z \ (P(x,y) \wedge P(y,z)) \to P(x,z)) \wedge \\ & (\forall x \exists y \, P(y,x))). \end{aligned} \quad (1)$$

It is not hard to see that there is a $Res*(k)$ refutation of MEP_n of size $\exp\left(\Omega\left(\frac{\log k}{k} n\right)\right)$ for any k, $2 \leq k \leq n-1$. Note also that the $Res*(n-1)$ proof of MEP_n is essentially the same as the (DAG-like) Resolution proof of the principle, so that our result is consistent with the known fact that MEP_n is hard for Tree-like Resolution, but easy for Resolution.

4 Complexity gap for $Rel\,(Taut)$

Let us first introduce some conventions. Recall that φ_n is built upon two kinds of variables, r's and s's. *r-variables correspond to the relation symbols* in the original sentence Φ. Suppose there are m such variables, $r^1, r^2, \ldots r^m$, having arities $p_1, p_2, \ldots p_m$, and let us denote $p = \max_i p_i$. Given the r-variable $r^i_{j_1, j_2, \ldots j_{p_i}}$, we say it *mentions* the elements $j_1, j_2, \ldots j_{p_i}$. *s-variables correspond to the Skolem relations* we use to encode Φ as a set of clauses. Suppose there are l such variables $s^1, s^2, \ldots s^l$. Given the s-variable $s^i_{j_1, j_2, \ldots j_{q_i}, j_0}$, we say it *mentions only its arguments* $j_1, j_2, \ldots j_{q_i}$, *but not the witness*, j_0. We also define its arity to be q_i, not $q_i + 1$. Let us denote $q = \max_i q_i$ as well as $t = \max\{p, q\}$.

We will first prove that we need big clauses to refute Φ.

Lemma 4.1 *Any Resolution refutation of φ_n contains a clause which mentions at least $n^{1/q} / \left(2l^{1/q}\right) - t$ elements.*

Proof We will describe a deterministic Adversary's strategy which enforces a big clause. As usual Adversary holds an infinite model M in which Φ holds. We say that an element is *busy* in the current position (i.e. clause) iff it is mentioned by the clause. We say an element is *hidden* iff it is not busy, but is the witness of a Skolem relation, having all its arguments busy. An element, which is neither busy nor hidden, is said to be *free*. At each stage in the Prover-Adversary game Adversary maintains two disjoint sets B and H. B is the set of all the busy elements, and H is the set of the hidden elements.

The Adversary's strategy is now clear. At any stage in the game all the elements from $B \uplus H$ have interpretations in M. Initially $B = H = \emptyset$. There are two kinds of Prover's moves:

1. She queries a new propositional variable. The easier case is when the variable can be evaluated under the current interpretation. Adversary replies with the value, and does not change B and H. If the variable cannot be evaluated, Adversary needs to enlarge B with all the new elements mentioned by the variable. Note that there is a constant number of such elements, namely at most t. H then has to be enlarged as well with the witnesses of all the new tuples in B.

2. She forgets a propositional variable. Some elements from B may then become non-busy. Adversary removes these from B. Some of them may become hidden, i.e.

they are witnesses of a Skolem relation with all its arguments from B, and go to H. The rest become free. Some elements from H, namely the witnesses of the tuples which contained at least one of the just forgotten elements, become free as well.

In any case no contradiction can be achieved as far as $|B| + |H| \leq n$. On the other hand there are l Skolem relation, each with arity at most q, so at any time $|H| \leq l\,|B|^q$. Thus at any stage in the game, before a contradiction is achieved, we have

$$|B| + l\,|B|^q \leq n.$$

Since $|B| = n^{1/q}/\left(2l^{1/q}\right)$ satisfies the inequality, and $|B|$ increases by at most t after any stage, there should be a point where $|B| > n^{1/q}/\left(2l^{1/q}\right) - t$ as claimed. □

Let us now consider $Rel\left(\varphi_n\right)$. We will first describe a distribution of *random restrictions* which kills any big clause in any Resolution refutation of $Rel\left(\varphi_n\right)$ with high probability (whp). The idea is to randomly divide the universe $U = [n]$ into two approximately equal parts. One of them, R, will represent the predicate, we relativise by, R^0; all the variables within it will remain unset. The rest, C, will be the "chaotic" part; all the variables within C and most of the variables between C and R will be set entirely at random.

More precisely, the random restrictions are as follows.

1. We first set all the variables r^0_j to either $\top$ or $\bot$ independently at random with equal probabilities, $1/2$. Let us denote the set of variables with $r^0_j = \top$ by R, and the set of variables with $r^0_j = \bot$ by C, $C = U \setminus R$.

2. We now set all the variables $r^i_{j_1,j_2,\ldots j_{p_i}}$, $i > 0$, which mention at least one element of C, i.e. $\{j_1, j_2, \ldots j_{p_i}\} \cap C \neq \emptyset$, to either $\top$ or $\bot$ independently at random with equal probabilities, $1/2$.

3. We set all the variables $s^i_{j_1,j_2,\ldots j_{q_i},j_0}$, which mention at least one element of C, i.e. $\{j_1, j_2, \ldots j_{q_i}\} \cap C \neq \emptyset$, to either $\top$ or $\bot$ independently at random with equal probabilities, $1/2$.

4. We finally set to $\bot$ all the variables $s^i_{j_1,j_2,\ldots j_{q_i},j_0}$ which mention only elements from R, but the witness is in C, i.e. $\{j_1, j_2, \ldots j_{q_i}\} \subseteq R$ and $j_0 \in C$.

It is very important to note that the variables and clauses, which survive the random restriction, define exactly Φ on R, i.e. $\varphi_{|R|}$.

There are few minor problems. The third case of the above description may violate an axiom as well as the first case may make R very small. We can however see that these bad events happen with exponentially small probability.

Lemma 4.2 *The probability that the random restrictions are inconsistent with the axioms or $|R| \leq n/4$ is at most $ln^q/2^n + 1/e^{n/16}$.*

Proof Indeed, an axiom is violated iff in the third case there is a q_i-tuple $j_1, j_2, \ldots j_{q_i}$ such that for every j_0, $s^i_{j_1,j_2,\ldots j_{q_i},j_0} = \bot$, i.e. there is no witness for the tuple. The probability for this is $1/2^n$ and there are at most ln^q such tuples, so that the union-bound gives $ln^q/2^n$. By the Chernoff bound the probability that $|R| \leq n/4$ is at most $e^{-n/16}$. □

The next step is to show that the random restrictions kill any clause with exponential probability in the number of the elements mentioned.

Lemma 4.3 *Given a clause, which mentions at least k elements, the probability it does not evaluate to $\top$ under the random restrictions is at most $(3/4)^{k/t}$.*

Proof Let us denote the clause by A, and perform the following experiment. We pick up a literal l_1 from A. The probability that at least one of the elements, mentioned by l_1, is in the chaotic set C is at least $1/2$. Given such an element, the probability that l_1 evaluates to $\top$ under the random restrictions is $1/2$. Thus the probability l_1 does not evaluate to $\top$ is at most $3/4$. We now take all the elements, mentioned by l_1 and mark them.

We pick another literal l_2 from A which mentions at least one unmarked element, and proceed as we have done with l_1. We then pick yet another literal l_3 and so on.

The clause A mentions at least k elements, whilst after having considered a literal we mark at most t elements, so that we can repeat the above procedure at least k/t times. These trials have been independent by the construction of the random restrictions. Therefore the probability that A does not evaluate to $\top$ is at most $(3/4)^{k/t}$ as claimed. □

We can now prove the main result of the section.

Theorem 4.4 *A SO$\exists$ sentence Φ is given which fails in all finite models, but holds in some infinite model. Let us denote $Rel\left(\Phi\right)$ the relativisation of Φ with respect to a unary predicate. Let $Rel\left(\varphi_n\right)$ be the translation of the latter into set of clauses, assuming a finite model of size n. Then there is a constant ε, depending only on Φ, such that any Resolution refutation of $Rel\left(\varphi_n\right)$ is of size* $\exp\left(\Omega\left(n^\varepsilon\right)\right)$.

Proof Let us denote $k = \frac{n^{1/p}}{2(4l)^{1/p}} - t$, and say a clause is *big*, iff it mentions at least k elements. We will prove that any Resolution refutation of $Rel\left(\varphi_n\right)$ contains exponentially many in k big clauses, which would give the desired lower bound.

Assume, for the sake of contradiction, there is a refutation Γ which contains at most $(4/3)^{k/2}$ big clauses. We hit Γ by random restrictions. By the lemma 4.2 and the lemma 4.3 + union-bound, the probability that the restrictions are "bad" is at most

$$\frac{ln^q}{2^n} + \frac{1}{e^{n/16}} + \left(\frac{3}{4}\right)^{n^{1/p}/\left(2(4l)^{1/p}\right)-t}.$$

As this quantity is smaller than 1 for big enough n (recall that l, p, q and t depend on Φ, but not on n), there is a set of good restrictions, "good" meaning that they kill all the big clauses, and moreover what survives is φ_m for some $m > n/4$. But now by the lemma 4.1 there has to be at least one big clause in the restricted refutation which is a contradiction. □

Note that we do not claim that the bound proven, $\exp\left(\Omega\left(n^{1/p}\right)\right)$ is tight. As a matter of fact, we believe the right lower bound is $\exp\left(\Omega\left(n\right)\right)$, but we also believe this might be very hard to prove, say as hard as proving the Complexity Gap theorem for (DAG-like) Resolution.

5 Built-in order "kills" the gap

We will first show that there is no Complexity gap for Tree-like Resolution if we enrich the SO∃-language with a built-in order predicate.

Theorem 5.1 *There is no tree-resolution complexity gap for the logical sentences in the language $SO\exists$ + built-in order.*

Proof Let us first describe the argument very informally. Assume a finite model of size n. There are know tautologies which requires size $2^{\Omega(n)}$ to refute. The most natural of them is *Minimal Element Principle (MEP)* which has already been mentioned (note that the partial order, defined by MEP, is entirely independent from the built-in total order predicate).

As we have built-in order, we can interpret the elements of the universe as the first n natural numbers. We can define in the SO∃ language a broad class of functions, such as $\log x$, $x^{p/q}$ (p and q integers) and so on. We will show that it is easy, i.e. polynomially size doable, to verify that a given element k of the universe is $f\left(n\right)$, where f is a function from the class. Then we will restrict Minimal Element Principle to the first k elements (in the built-in order) to get a sentence in the SO∃ language + built-in order predicate whose optimal refutation is of size $2^{\Omega(f(n))}$, provided $f\left(n\right) = \Omega\left(\log n\right)$.

What remains is to show how to define and verify $f\left(n\right)$ in polynomial in n size. As an example, we will give the definitions of $\sqrt{\cdot}$ and $\log\left(\cdot\right)$.

In what follows, all the relation symbols are quantified existentially and the free variables are quantified universally. Moreover, the definitions are nested in the order they appear, starting with the above definition of a total order, being outermost one. We denote by L the built-in order predicate, i.e. $L\left(x,y\right)$ stands for $x < y$.

We first define the successor function as the relation $S\left(x,y\right)$ standing for y is the successor of x.

$$S\left(x,y\right) \equiv \left(L\left(x,y\right) \wedge \left(\forall z \neg\left(L\left(x,z\right) \wedge L\left(z,y\right)\right)\right)\right)$$

We can define any constants, by the relations $C_a\left(x\right)$, meaning $x = a$.

$$C_0\left(x\right) \equiv \left(\forall y\ \neg S\left(y,x\right)\right)$$

$$C_a\left(x\right) \equiv \left(\exists y\ \left(C_{a-1}\left(y\right) \wedge S\left(y,x\right)\right)\right)$$

We are now ready to define addition and multiplication recursively by the following relations, $A\left(x,y,z\right)$ and $M\left(x,y,z\right)$ standing for $x + y = z$ and $x \times y = z$, respectively.

$$\begin{aligned} A\left(x,y,z\right) \quad &\equiv \quad \left(\left(C_0\left(x\right) \wedge C_0\left(y\right) \wedge C_0\left(z\right)\right)\right. \\ &\vee \quad \exists u,v\ \left(S\left(v,z\right)\right) \\ &\quad \wedge \left(\left(S\left(u,x\right) \wedge A\left(u,y,v\right)\right)\right. \\ &\quad \left.\left.\vee \left(S\left(u,y\right) \wedge A\left(x,u,v\right)\right)\right)\right) \end{aligned}$$

$$\begin{aligned} M\left(x,y,z\right) \quad &\equiv \quad \left(\left(\left(C_0\left(x\right) \vee C_0\left(y\right)\right) \wedge C_0\left(z\right)\right)\right. \\ &\vee \quad \exists u,v\ \left(\left(S\left(u,x\right) \wedge A\left(y,v,z\right) \wedge M\left(u,y,v\right)\right)\right. \\ &\quad \left.\left.\vee \left(S\left(u,y\right) \wedge A\left(x,v,z\right) \wedge M\left(x,u,v\right)\right)\right)\right) \end{aligned}$$

We can now define $y = \lfloor\sqrt{x}\rfloor$ by $y = \lfloor\sqrt{x}\rfloor$ if and only if either $y^2 \leq x < (y+1)^2$ or $y^2 \leq x$, but $(y+1)^2 > n$.

$$\begin{aligned} R_2\left(x,y\right) \quad &\equiv \quad \left(M\left(y,y,x\right)\right. \\ &\vee \quad \exists u\ \left(M\left(y,y,u\right) \wedge L\left(u,x\right)\right. \\ &\wedge \quad \left(\exists z,w\ \left(S\left(y,z\right) \wedge M\left(z,z,w\right) \wedge L\left(x,w\right)\right)\right. \\ &\quad \left.\left.\left.\vee \forall z,w\ \left(S\left(y,z\right) \rightarrow \neg M\left(z,z,w\right)\right)\right)\right)\right) \end{aligned}$$

We also define $y = \lfloor\frac{x}{2}\rfloor$ as

$$\begin{aligned} D_2(x,y) \quad \equiv \quad & \exists u,v,w \;\; (M(u,x,v) \wedge A(v,w,u) \\ & \wedge C_2(u) \wedge (C_0(v) \vee C_1(v))) \, , \end{aligned}$$

and finally $y = \lfloor \log x \rfloor$, using the recursion

$$\lfloor \log x \rfloor = \begin{cases} 0 & x = 1 \\ 1 + \lfloor \log \lfloor \frac{x}{2} \rfloor \rfloor & x > 1 \end{cases}$$

$$\begin{aligned} L_2(x,y) \quad \equiv \quad & (C_1(x) \wedge C_0(y) \\ \vee \quad & (\exists u,v \;\; (S(v,y) \wedge D_2(x,u) \wedge L_2(u,v)))) \, . \end{aligned}$$

We shall not formally prove that it is straightforward to verify all the functions defined above within the decision tree computational model. The sketch of the proof is, however, pretty clear: as all the definitions are inductive and there are no mutual recursive relations, we check the functions in the order they are defined, starting from the basis cases. Thus all the queries are forced in the sense that if the assignment does not satisfy the definition of the given relation then the "wrong" answer leads to an immediate contradiction. Thus the size of the initial part of the tree, verifying the definitions of these functions, is polynomial in n as claimed. □

Of course, the result holds for many other propositional proof systems as Tree-like Resolution is a very weak system, and can be polynomially-simulated by them. As an important consequence we get the following

Theorem 5.2 *There is no Tree-like Resolution Complexity gap above* $2^{\theta(n \log n)}$.

Proof Let us first observe that there are $SO\exists$ statements with optimal proofs of size $2^{\theta(n^p)}$ for every $p \geq 1$. It is enough to take MEP_{n^p} where x, y and z in 1 are p-tuples instead of single elements of the universe. To get the intermediate complexities, we could restrict the last element of any such p-tuple to be less than or equal to $f(n)$ where f is some function and n is the size of the universe (model). The optimal tree resolution proof of the obtained in this way statement would be $2^{\theta\left(n^{p-1} f(n)\right)}$. However we have not any predefined functions in our language. The definition of f therefore should be a part of the sentence and the proof should "verify" the definition of the function f. We shall use the same argument as we have done in the previous proof. However we have not total order either, so we have to define it within the SO∃ language, and to verify it by a decision tree. □

The total order can be defined as

$$\begin{aligned} \exists L \quad & (\forall x \, \neg L(x,x) \\ \wedge \quad & \forall x,y \;\; ((x=y) \vee L(x,y) \vee L(y,x)) \\ \wedge \quad & \forall x,y,z \;\; ((L(x,y) \wedge L(y,z)) \rightarrow L(x,z))) \, , \end{aligned}$$

and we can complete the argument by proving the following lemma.

Lemma 5.3 *The relation L can be* optimally *verified by a decision tree of size* $2^{\theta(n \log n)}$.

Proof The sentence translates into the following set of clauses:

$$\begin{array}{rl} \neg l_{ii} & i \\ l_{ij} \vee l_{ji} & i,j \\ \neg l_{ij} \vee \neg l_{jk} \vee l_{ik} & i,j,k. \end{array}$$

It is clear that every permutation π of $\{1, 2, \ldots n\}$ defines a satisfying assignment by setting l_{ij} to $\top$ if and only if $\pi(i) < \pi(j)$ and vice versa. This observation immediately implies a lower bound of $n! = 2^{\Omega(n \log n)}$.

A decision tree which verifies L can be constructed by incrementally ordering the elements of the universe. Suppose we already have a decision tree which orders the first j elements, i.e. each leaf corresponds either to a contradiction or to a permutation of $\{1, 2, \ldots j\}$ as explained above. We can now expand the latter leaves by finding the place of the $j+1$-th element. In doing so, we use binary search which uses $O(\log(j+1))$ *free-choice* queries. Once the place of the $j+1$-th element has been found, all the queries involving it and some of the previous j elements are *forced*, i.e. one of the answers leads to an immediate contradiction. Thus the forced queries contribute a polynomial factor to the size of the subtree consisting of the free-choice queries only. The depth of this subtree is $\sum_{j=1}^{n-1} O(\log(j+1)) = O(n \log n)$, and therefore its size and the size of the entire decision tree is $2^{O(n \log n)}$. □

References

[1] P. Beame and T. Pitassi. Simplified and improved resolution lower bounds. In *Proceedings of the 37th annual IEEE symposium on Foundation Of Computer Science*, pages 274–282, 1996.

[2] E. Ben-Sasson, R. Impagliazzo, and A. Wigderson. Near-optimal separation of general and tree-like resolution. *Combinatorica*. to appear.

[3] M. Bonet and N. Galesi. A study of proof search algorithms for resolution and polynomial calculus. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science*. IEEE, 1999.

[4] S. Dantchev. Relativisation makes tautologies harder for resolution. submitted, October 2002.

[5] S. Dantchev and S. Riis. Tree resolution proofs of the weak pigeon-hole principle. In *In proceedings of the 16th annual IEEE Conference on Comutational Complexity*. IEEE, June 2001.

[6] A. Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297–308, 1985.

[7] J. Krajíĉek. *Bounded Arithmetic, Propositional Logic, and Complexity Theory*. Cambridge University Press, 1995.

[8] J. Krajicek. Combinatorics of first order structures and propositional proof systems. July 2001.

[9] P. Pudlák. Proofs as games. *American Mathematical Monthly*, pages 541–550, June-July 2000.

[10] S. Riis. A complexity gap for tree-resolution. *Computational Complexity*, 10:179–209, 2001.