48th Annual IEEE Symposium on Foundations of Computer Science

Parameterized Proof Complexity*

Stefan Dantchev, Barnaby Martin, and Stefan Szeider

Department of Computer Science
Durham University, Durham, England, UK
[s.s.dantchev,b.d.martin,stefan.szeider]@durham.ac.uk

Abstract

We propose a proof-theoretic approach for gaining evidence that certain parameterized problems are not fixed-parameter tractable. We consider proofs that witness that a given propositional CNF formula cannot be satisfied by a truth assignment that sets at most k variables to true*, considering k as the parameter (we call such a formula a parameterized contradiction). One could separate the parameterized complexity classes* FPT *and* W[2] *by showing that there is no fpt-bounded parameterized proof system, i.e., that there is no proof system that admits proofs of size $f(k)n^{O(1)}$ where f is a computable function and n represents the size of the propositional formula.*

By way of a first step, we introduce the system of parameterized tree-like resolution, and show that this system is not fpt-bounded. Indeed we give a general result on the size of shortest tree-like resolution proofs of parameterized contradictions that uniformly encode first-order principles over a universe of size n. We establish a dichotomy theorem that splits the exponential case of Riis's Complexity Gap Theorem into two sub-cases, one that admits proofs of size $f(k)n^{O(1)}$ and one that does not.

We also discuss how the set of parameterized contradictions may be embedded into the set of (ordinary) contradictions by the addition of new axioms. When embedded into general (DAG-like) resolution, we demonstrate that the pigeonhole principle has a proof of size $2^k n^2$. This contrasts with the case of tree-like resolution where the embedded pigeonhole principle falls into the "non-FPT" category of our dichotomy.

1 Introduction

In recent years parameterized complexity and fixed-parameter algorithms have become an important branch of algorithm design and analysis; hundreds of research papers have been published in the area (see, e.g., the references given in [2, 6, 8, 11]). In parameterized complexity one considers computational problems in a two-dimensional setting: the first dimension is the usual *input size* n, the second dimension is a positive integer k, the *parameter*. A problem is fixed-parameter tractable if it can be solved in time $f(k)n^{O(1)}$ where f denotes a computable, possibly exponential, function. Several NP-hard problems have natural parameterizations that admit fixed-parameter tractability. For example, given a graph with n vertices, one can check in time $O(1.273^k + nk)$ (and polynomial space) whether the graph has a vertex cover of size at most k [3]. On the other hand, several parameterized problems such as CLIQUE (has a given graph a clique of size at least k?) are believed to be *not* fixed-parameter tractable. BOUNDED CNF SATISFIABILITY is a further problem that is believed to be *not* fixed-parameter tractable (and which will play a special role in the sequel): given a propositional formula in conjunctive normal form, is there a satisfying truth assignment that sets at most k variables to *true*?

Parameterized complexity offers also a completeness theory. Numerous parameterized problems that appear to be not fixed-parameter tractable have been classified as being complete under *fpt-reductions* for complexity classes of the so-called *weft hierarchy* $\mathrm{W}[1] \subseteq \mathrm{W}[2] \subseteq \mathrm{W}[3] \subseteq \cdots$. For example, CLIQUE and BOUNDED CNF SATISFIABILITY are complete for the first two levels of the weft hierarchy, respectively. We will outline the basic notions of parameterized complexity in Section 2.1; for an in-depth treatment of parameterized complexity classes and fpt-reduction we refer the reader to Flum and Grohe's monograph [8].

It is widely believed that problems that are hard for the weft hierarchy are not fixed-parameter tractable. Up to now there are mainly three types of evidence:

1. *Accumulative evidence*: numerous problems are known which are hard or complete for classes of the weft hierarchy, and for which no fixed-parameter algorithm has been found in spite of considerable efforts [2].

*Research supported by Engineering and Physical Sciences Research Council, UK, Projects EP/C526120/1 and EP/E001394/1.

0272-5428/07 $25.00 © 2007 IEEE
DOI 10.1109/FOCS.2007.53

2. k*-step Halting Problems* for non-deterministic Turing machines are complete for the classes W[1] (single-tape) and W[2] (multi-tape) [8]. A Turing machine is such an opaque and generic object that it does not appear reasonable that we should be able to decide if a given Turing machine on a given input has some accepting path without looking at the paths.

3. If a problem that is hard for a class of the weft hierarchy turns out to be fixed-parameter tractable, then the *Exponential Time Hypothesis* (ETH) fails, i.e., there is a $2^{o(n)}$ time algorithm for the n-variable 3-SAT problem [9]. ETH is closely related to the parameterized complexity class M[1] which lies between FPT and W[1] (see [8]).

We propose a new approach for gaining further evidence that certain parameterized problems are not fixed-parameter tractable. We generalize concepts of proof complexity to the two-dimensional setting of parameterized complexity. This allows us to formulate a parameterized version of the program of Cook and Reckhow [4]. Their program attempts to gain evidence for NP $\neq$ co-NP, and in turn for P $\neq$ NP, by showing that propositional proof systems are not polynomially bounded. We introduce the concept of parameterized proof systems; in our program, lower bounds for the length of proofs in these new systems yield evidence that certain parameterized problems are not fixed-parameter tractable.

In propositional proof complexity one usually constructs a sequence of tautologies (or contradictions), and shows that the sequence requires proofs (or refutations) of super-polynomial size in the proof system under consideration. In the scenario of contradictions and refutations, such sequences of propositional formulas frequently encode a first-order (FO) sentence (such as the pigeon hole principle) where the n-th formula of the sequence states that the FO sentence has no model of size n. S. Riis [13] established a meta-theorem that exactly pinpoints under which circumstances a given FO sentence gives rise to a sequence of propositional formulas that have polynomial-sized refutations in the system of tree-like resolution. Namely, if the sequence has not tree-like resolution refutations of polynomial size, then shortest tree-like resolution refutations have size at least $2^{\varepsilon n}$ for a positive constant ε that only depends on the FO sentence. Hence there is a *gap* between two possible proof complexities. The case of exponential size prevails exactly when the FO sentence has no finite but some infinite model.

In this paper we show a meta-theorem regarding the complexity of parameterized tree-like resolution. To this aim we consider *parameterized contradictions* which are pairs $(\mathcal{F}, k)$ where $\mathcal{F}$ is a propositional formula in CNF and k is an integer, such that $\mathcal{F}$ cannot be satisfied by a truth assignment that sets at most k variables to *true*. Parameterized contradictions form a co-W[2]-complete language. Hence FPT $=$ co-W[2] $=$ W[2] implies that there is a proof system that admits proofs of size at most $f(k)n^{O(1)}$ for parameterized contradictions $(\mathcal{F}, k)$ where n represents the size of $\mathcal{F}$. We call such a (hypothetical) proof system *fpt-bounded*.

In this paper we consider the relatively weak system of tree-like resolution. A parameterized tree-like resolution refutation for a parameterized contradiction $(\mathcal{F}, k)$ has built-in access to all clauses with more than k negated variables as additional axioms. We show a meta-theorem that classifies exactly the complexity of parameterized tree-like resolution refutations for parameterized contradictions. Our theorem allows a refined view of the exponential case of Riis's Theorem: Consider the sequence $\langle \mathcal{C}_{\psi,n} \rangle_{n \in \mathbb{N}}$ of propositional formulas generated from a FO sentence ψ that has no finite but some infinite model. For a positive integer k we get a sequence of parameterized contradictions $\langle (\mathcal{C}_{\psi,n}, k) \rangle_{n \in \mathbb{N}}$. We show that exactly one of the following two cases holds (and provide a criterion that decides which one).

2a. $(\mathcal{C}_{\psi,n}, k)$ has a parameterized tree-like resolution refutation of size $\beta^k n^\alpha$ for some constants α and β which depend on ψ only.

2b. There exists a constant γ, $0 < \gamma \leq 1$, such that for every $n > k$, every parameterized tree-like resolution refutation of $(\mathcal{C}_{\psi,n}, k)$ is of size at least n^{k^γ}.

We establish the upper bound $\beta^k n^\alpha$ via certain boolean decision trees. For the lower bound n^{k^γ} we use a game-theoretic argument.

We provide examples of FO sentences for each of the above categories. In particular, the examples for the n^{k^γ} case (Examples 15 and 16) show that parameterized tree-like resolution is not fpt-bounded.

As discussed, a parameterized tree-like resolution refutation for the parameterized contradiction $(\mathcal{F}, k)$ has access to all clauses with more than k negated variables as additional axioms. However, these axioms are not considered to be a part of the input parameterized contradiction; rather they are thought of as belonging to the resolution system itself (whence the "parameterized" in "parameterized tree-like resolution"). In the final section of the paper, we consider how such axioms could be introduced to a parameterized contradiction, thus creating an ordinary contradiction ripe for an ordinary proof system. In this manner, we can embed the set of parameterized contradictions into the set of (ordinary) contradictions. Given a proof system, and considering the parameter to be preserved, this embedding itself gives rise to a parameterized proof system. The embedding we consider is well-behaved, in that it preserves the complexity gap of parameterized tree-like resolution. In particular, the pigeonhole principle remains "hard" – in category (2b) – when embedded in tree-like resolution. However, when considered with general (DAG-like) resolution, the embedded pigeonhole principle has refutations of size $2^k n^2$.

Owing to space limitations some technical proofs are omitted. Full proofs and further examples can be found in a technical report [5].

2 Preliminaries

2.1 Fixed-parameter Tractability

In the following let Σ denote an arbitrary but fixed finite alphabet. A *parameterized language* is a set $L \subseteq \Sigma^* \times \mathbb{N}$ where $\mathbb{N}$ denotes the set of positive integers. If (I, k) is in a parameterized language L, then we call I the *main part* and k *the parameter*. We identify a parameterized language with the decision problem "$(I, k) \in L$?" and will therefore synonymously use the terms *parameterized problem* and parameterized language. A parameterized problem L is called *fixed-parameter tractable* if membership of (I, k) in L can be deterministically decided in time

$$f(k)|I|^{O(1)} \tag{1}$$

where f denotes a computable function. FPT denotes the class of all fixed-parameter tractable decision problems; algorithms that achieve the time complexity (1) are called *fixed-parameter algorithms*. The key point of this definition is that the exponential growth is confined to the parameter only, in contrast to running times of the form

$$|I|^{O(f(k))}. \tag{2}$$

There is theoretical evidence that parameterized problems like CLIQUE are *not* fixed-parameter tractable. This evidence is provided via a completeness theory which is similar to the theory of NP-completeness. This completeness theory is based on the following notion of reductions: Let $L_1 \in \Sigma_1^* \times \mathbb{N}$ and $L_2 \in \Sigma_2^* \times \mathbb{N}$ be parameterized problems. An *fpt-reduction* from L_1 to L_2 is a mapping $R : \Sigma_1^* \times \mathbb{N} \to \Sigma_2^* \times \mathbb{N}$ such that

1. $(I, k) \in L_1$ if and only if $R(I, k) \in L_2$.
2. R is computable by a fixed-parameter algorithm, i.e., there is a computable function f such that $R(I, k)$ can be computed in time $f(k)|I|^{O(1)}$.
3. There is a computable function g such that whenever $R(I, k) = (I', k')$, then $k' \leq g(k)$.

A parameterized complexity class $\mathcal{C}$ is the equivalence class of a parameterized problem under fpt-reductions. It is easy to see that FPT is closed under fpt-reductions, thus FPT is a parameterized complexity class. Parameterized problems appear to have several degrees of intractability, as manifested by the *weft hierarchy*. The classes W[t] of this hierarchy form a chain

$$\text{FPT} \subseteq \text{W[1]} \subseteq \text{W[2]} \subseteq \cdots \subseteq \text{XP}$$

where all inclusions are assumed to be proper. Here XP denotes the class of problems solvable in time $O(|I|^{f(k)})$; it is known that FPT $\neq$ XP [6]. Each class W[t] is defined as the equivalence class of a certain canonical weighted satisfiability problem for decision circuits. For W[2] the canonical problem is equivalent to the following satisfiability problem:

> WEIGHTED CNF SATISFIABILITY
>
> *Instance:* A propositional formula $\mathcal{F}$ in conjunctive normal form (CNF), and a positive integer k.
>
> *Parameter:* k.
>
> *Question:* Can $\mathcal{F}$ be satisfied by a truth assignment τ that sets exactly k variables to *true*? (k is the *weight* of τ.)

Note that if the clauses of the CNF formula are required to contain at most three literals, we get the W[1]-complete problem WEIGHTED 3-CNF SATISFIABILITY. Let BOUNDED CNF SATISFIABILITY denote the problem obtained from WEIGHTED CNF SATISFIABILITY by allowing truth assignments of weight *at most* k. It is easy to see that this relaxation does not change the parameterized complexity of the problem since BOUNDED CNF SATISFIABILITY contains the W[2]-complete problem HITTING SET [6] as a special case.

Lemma 1. BOUNDED CNF SATISFIABILITY *is complete for the class* W[2] *under fpt-reductions.*

The related problem BOUNDED 3-CNF SATISFIABILITY is actually fixed-parameter tractable; this explains why our study concerns W[2] and not W[1].

As in classical complexity theory, we can define for a parameterized complexity class $\mathcal{C}$ the complementary complexity class co-$\mathcal{C} = \{\, \overline{L} : L \in \mathcal{C} \,\}$ where $\overline{L} = (\Sigma^* \times \mathbb{N}) \setminus L$ for a parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$. Clearly FPT = co-FPT. It is easy to see that if $\mathcal{C}$ is closed under fpt-reductions, then so is co-$\mathcal{C}$. Thus, in particular, each class W[t] of the weft hierarchy gives rise to a parameterized complexity class co-W[t].

2.2 Parameterized Proof Systems

Definition 2. *Let $L \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized language. A* parameterized proof system for L *is an onto mapping* $\Gamma : (\Sigma_1^* \times \mathbb{N}) \to L$ *for some alphabet* Σ_1 *where* Γ *can be computed by a fixed-parameter algorithm.*

We say that Γ *is* fpt-bounded *if there exist computable functions* f *and* g *such that for every* $(I, k) \in L$ *there is* $(I', k') \in \Sigma_1^* \times \mathbb{N}$ *with* $\Gamma(I', k') \leq (I, k)$, $|I'| = f(k)|I|^{O(1)}$, *and* $k' \leq g(k)$.

Note that the problems of the classes W[t] of the weft hierarchy have fpt-bounded proof systems since the yes-instances of these problems have small witnesses. Consider, for example, the W[2]-complete problem $L =$ BOUNDED CNF SATISFIABILITY. Let $S_{\mathcal{F},\tau,k}$ denote a string over some alphabet Σ_1 that encodes a CNF formula $\mathcal{F}$ together with a satisfying truth assignment τ of weight $\leq k$ for $\mathcal{F}$. A proof system Γ for L can now be defined by setting $\Gamma(w, k) = (\mathcal{F}, k)$ if $w = S_{\mathcal{F},\tau,k}$, and otherwise $\Gamma(w, k) = (\mathcal{F}_0, k_0)$ for some fixed $(\mathcal{F}_0, k_0) \in L$. Evidently, Γ is fpt-bounded.

However, the situation is different for the classes co-W[t]; specifically, in this case, for co-W[2]. We can witness that a CNF formula with n variables has no satisfying assignment of weight $\leq k$ by listing all $O(k \cdot n^k)$ assignments of weight $\leq k$, then checking that none is satisfying. However, this listing requires too much space and apparently we cannot use it for the construction of an fpt-bounded proof system.

Lemma 3. *Let $\mathcal{C}$ be a parameterized complexity class and let L be a* co-$\mathcal{C}$*-complete parameterized problem. If there is no fpt-bounded parameterized proof system for L, then* $\mathcal{C} \neq$ FPT.

This result follows by a standard argument in which the computation of a Turing machine is considered as a proof. In view of this lemma we suggest a program à la Cook-Reckhow for gaining evidence that the complexity classes from the weft hierarchy are distinct from FPT. This program consists of showing that particular parameterized proof systems are not fpt-bounded. For such an approach we would start with a weak system such as a parameterized version of tree-like resolution. The consideration of stronger systems is left for future research.

2.3 From First-Order to Propositional Logic

Next we describe a translation of a FO sentence to a sequence of propositional CNF formulas. We use the language of FO logic with equality but with neither function nor constant symbols. We omit functions and constants only for the sake of a clearer exposition; note that we may simulate constants in a single FO sentence with added *outermost* existential quantification on new variables replacing those constants. We assume that the FO sentence is given as a conjunction of FO sentences, each of which is in prenex normal form; thus, we need only explain the translation of a single FO sentence in prenex normal form. The case of a purely universal sentence is easy – a sentence ψ of the form

$$\forall x_1, \ldots, x_k \; \mathcal{F}(x_1, \ldots, x_k)$$

where $\mathcal{F}$ is quantifier-free, is translated into a sequence of propositional formulas in CNF $\langle \mathcal{C}_{\psi,n} \rangle_{n \in \mathbb{N}}$, of which the n-th member $\mathcal{C}_{\psi,n}$ is constructed as follows. Let $[n] = \{1, 2, \ldots, n\}$. For instantiations $x_1, \ldots, x_k \in [n]$, we can consider $\mathcal{F}(x_1, \ldots, x_k)$ to be a propositional formula over propositional variables of two different kinds: $R(x_{i_1}, \ldots, x_{i_p})$, where R is a p-ary predicate symbol, and $(x_i = x_j)$. We transform $\mathcal{F}$ into CNF and then take the union of all such CNF formulas for $(x_1, \ldots, x_k)$ ranging over $[n]^k$. The variables of the form $(x_i = x_j)$ evaluate to either true or false, thus we are left with variables of the form $R(x_{i_1}, \ldots, x_{i_p})$ only.

The general case, a sentence ψ of the form

$$\forall x_1 \exists y_1 \forall x_2 \exists y_2 \ldots \forall x_k \exists y_k \; \mathcal{F}(x_1, \ldots, x_k, y_1, \ldots, y_k),$$

can be reduced to the previous case by Skolemization. We introduce *Skolem relations* $S_i(x_1, \ldots, x_i, y_i)$ for $1 \leq i \leq k$. $S_i(x_1, \ldots, x_i, y_i)$ witnesses y_i for any given $x_1, \ldots, x_i$, so we need to add *Skolem clauses* stating that such a witness always exists, i.e.,

$$\bigvee_{y_i=1}^{n} S_i(x_1, \ldots, x_i, y_i) \quad \text{for all } (x_1, \ldots, x_i) \in [n]^i .$$

The original sentence can be transformed into the following purely universal sentence

$$\forall x_1, \ldots x_k, y_1, \ldots y_k \bigvee_{i=1}^{k} \begin{array}{l} \neg S_i(x_1, \ldots x_i, y_i) \vee \\ \mathcal{F}(x_1, \ldots x_k, y_1, \ldots y_k). \end{array}$$

By construction it is clear that, for FO sentences ψ, the CNF formula $\mathcal{C}_{\psi,n}$ is satisfiable if and only if ψ has a model of size n. Thus satisfiability questions on the sequence $\langle \mathcal{C}_{\psi,n} \rangle_{n \in \mathbb{N}}$ relate to questions on the existence of non-empty finite models for ψ.

Remark 4. Note that the size of $\mathcal{C}_{\psi,n}$ with respect to some reasonable encoding is polynomial in n.

Example 5. We consider (the negation of) the pigeonhole principle. Let ψ^{PHP} be the conjunction of the following.

$$\forall x \exists y R(x, y)$$
$$\exists y \forall x \neg R(x, y)$$
$$\forall x \forall w \forall y \; \neg R(x, y) \vee \neg R(w, y) \vee x = w.$$

We translate this to the conjunction of the following universal clauses

$$\forall x \forall y\ \neg S_2(x,y) \vee R(x,y)$$
$$\forall y \forall x\ \neg S_1(y) \vee \neg R(x,y)$$
$$\forall x \forall y \forall w\ \neg R(x,y) \vee \neg R(w,y) \vee x = w$$

together with the Skolem clauses

$$\forall x \exists y S_2(x,y)$$
$$\exists y S_1(y).$$

For $x, y \in [n]$ we now consider $R(x,y)$, $S_2(x,y)$ and $S_1(y)$ to be propositional variables. $\mathcal{C}_{\psi^{\text{PHP}},n}$ is therefore the system of clauses

$$\neg S_2(x,y) \vee R(x,y),\ \neg S_1(y) \vee \neg R(x,y) \text{ and}$$
$$\neg R(x,y) \vee \neg R(w,y), \text{ for } x, y, w \in [n],\ w \neq x,$$

together with the Skolem clauses

$$\bigvee_{i=1}^{n} S_2(x,i), \text{ for } x \in [n], \text{ and } \bigvee_{i=1}^{n} S_1(i).$$

2.4 Parameterized Tree-like Resolution

A *literal* is either a propositional variable or the negation of a propositional variable. A *clause* is a disjunction of literals (and a propositional variable can appear only once in a clause). A set of clauses is a conjunction, i.e., it is *satisfiable* if there exists a truth assignment satisfying simultaneously all the clauses. *Resolution* is a proof system designed to *refute* a given set of clauses, i.e., to prove that it is unsatisfiable. This is done by means of a single derivation rule

$$\frac{C \vee v \quad \neg v \vee D}{C \vee D},$$

which we use to obtain a new clause from two already existing ones. The goal is to derive the empty clause – resolution is known to be sound and complete, i.e., we can derive the empty clause from the initial clauses if and only if the initial set of clauses was unsatisfiable.

In this paper, we shall work with a restricted version of resolution, namely *tree-like resolution*. In tree-like resolution we are not allowed to reuse any clause that has already been derived, i.e., we need to derive a clause as many times as we use it (this, of course, does not apply to the initial clauses). In other words, a tree-like resolution refutation can be viewed as a binary tree whose nodes are labeled with clauses. Every leaf is labeled with one of the original clauses, every clause at an internal node is obtained by a resolution step from the clauses at its two children nodes, and the root of the tree is labeled with the empty clause. We measure the *size* of a tree-like resolution refutation by the number of nodes.

It is not hard to see that a tree-like resolution refutation of a given set of clauses is equivalent to a *boolean decision tree* solving the *search problem* for that set of clauses. The search problem for an unsatisfiable set of clauses is defined as follows (see, e.g., Krajíček's book [10]): given a truth assignment, find a clause which is falsified under the assignment. A boolean decision tree solves the search problem by querying values of propositional variables and then branching on the answer. Without loss of generality, we may assume that no propositional variable is questioned twice on the same branch and that a branch of the tree is closed as soon as a falsified clause is found, under the partial assignment – conjunction of facts – obtained so far along that branch. When a branch is thus closed we say that an *elementary contradiction* has been obtained. Note that we consider a node of the decision tree to be labeled by the conjunction of facts thus far obtained together with the propositional variable there questioned. This is analogous to a node in a tree-like resolution refutation being labeled with its clause together with the variable just resolved. Given the equivalence between tree-like resolution refutations and boolean decision trees, we shall concentrate on the latter. Whenever we need to show that there is a certain tree-like resolution refutation of some unsatisfiable set of clauses, we shall construct a boolean decision tree for the corresponding search problem. On the other hand, whenever we claim a tree-like resolution lower bound, we shall prove it by an adversary argument against any boolean decision tree which solves the search problem.

We give working definitions of parameterized contradiction and parameterized tree-like resolution, which we shall use to state and prove the complexity gap for parameterized tree-like resolution.

Definition 6. *A* parameterized contradiction *is a pair $(\mathcal{F}, k)$ where $\mathcal{F}$ is a propositional CNF formula and k is a positive integer such that $\mathcal{F}$ has no satisfying assignment of weight at most k.*

Example 7. Let us consider an undirected graph $G = (V, E)$ that does not have a vertex cover of size $\leq k$. We introduce a propositional variable p_v for every vertex $v \in V$. Then the pair

$$\left(\bigwedge\nolimits_{\{u,v\} \in E} (p_u \vee p_v), k\right)$$

is a parameterized contradiction.

Let PARAMETERIZED CONTRADICTIONS be the language of parameterized contradictions. Note that PARAMETERIZED CONTRADICTIONS is the complement of BOUNDED CNF SATISFIABILITY and, as such, is co-W[2]-complete under fpt-reductions.

We can now define a parameterized version of tree-like resolution. As we have already explained, we shall give the definition in terms of boolean decision trees.

Definition 8. *Given a parameterized contradiction $\mathcal{P} = (\mathcal{F}, k)$, a* parameterized boolean decision tree *is a decision tree that queries values of propositional variables and branches on the answers; a branch of the tree is closed as soon as (1) or (2) happens:*

(1) an elementary contradiction is reached, i.e., the partial assignment obtained along the branch falsifies $\mathcal{F}$;

(2) the partial assignment obtained along the branch has more than k propositional variables set to true, i.e., has weight $> k$.

The fact that we can close branches by criterion (2) is equivalent to our having, built-in as axioms, all clauses of more than k negated variables. This represents the difference between parameterized boolean decision trees and (ordinary) boolean decision trees; hence also the difference between parameterized tree-like resolution and (ordinary) tree-like resolution.

3 Complexity Gap for Parameterized Tree-like Resolution

We first recall the complexity gap theorem for tree-like resolution proven by Riis [13].

Theorem 9. *Given a FO sentence ψ which fails in all finite models, consider its translation into a sequence of propositional CNF contradictions $\langle \mathcal{C}_{\psi,n} \rangle_{n\in\mathbb{N}}$. Then either 1 or 2 holds:*

1. *$\mathcal{C}_{\psi,n}$ has polynomial-size in n tree-like resolution refutations.*
2. *There exists a positive constant ε such that for every n, every tree-like resolution refutation of $\mathcal{C}_{\psi,n}$ is of size at least $2^{\varepsilon n}$.*

Furthermore, 2 holds if and only if ψ has an infinite model.

In the parameterized setting, one can hope that the second case above, the hard one, splits into two subcases. This is indeed true as we shall prove the following complexity gap theorem for *parameterized tree-like resolution*:

Theorem 10. *Given a FO sentence ψ, which fails in all finite models but holds in some infinite model, consider the sequence of parameterized contradictions $\langle \mathcal{D}_{\psi,n,k} \rangle_{n\in\mathbb{N}} = \langle (\mathcal{C}_{\psi,n}, k) \rangle_{n\in\mathbb{N}}$ where $\langle \mathcal{C}_{\psi,n} \rangle_{n\in\mathbb{N}}$ is the translation of ψ already defined. Then either 2a or 2b holds:*

- 2a. *$\mathcal{D}_{\psi,n,k}$ has a parameterized tree-like resolution refutation of size $\beta^k n^\alpha$ for some constants α and β which depend on ψ only.*
- 2b. *There exists a constant γ, $0 < \gamma \leq 1$, such that for every $n > k$, every parameterized tree-like resolution refutation of $\mathcal{D}_{\psi,n,k}$ is of size at least n^{k^γ}.*

Furthermore, 2b holds if and only if ψ has an infinite model whose induced hypergraph has no finite dominating set.

By proving that Case 2b can be attained (see Examples 15 and 16), and bearing in mind Remark 4, we derive the following as a corollary.

Corollary 11. *Parameterized tree-like resolution is not fpt-bounded.*

If we could prove that no parameterized proof system for PARAMETERIZED CONTRADICTIONS is fpt-bounded, then we would have derived W[2] $\neq$ FPT.

Before we prove Theorem 10, we need to give some definitions. For a model M, let $|M|$ denote the universe of M. Given a model M of a FO sentence ψ, either finite or infinite, the *hypergraph induced by the model M* has the elements of $|M|$ as vertices and as hyperedges those sets $\{y_1, \ldots y_l\}$ such that $(y_1, \ldots, y_l)$ appears as a tuple in some relation. (Recall that there are two kinds of relations – the extensional R relations which are present in the original FO sentence, and the S relations that we introduce when Skolemizing the sentence – both give rise to hyperedges.) A set of vertices is *independent* if it contains no hyperedge as a subset. Given a set X of vertices, a vertex $y \notin X$, and a set A such that $X \cup \{y\} \subseteq A \subseteq |M|$, we say that *$y$ is A-independent from X* if and only if (i) there is no self-loop $\{y\}$ at y, and (ii) there is no hyperedge $E \subseteq A$ which contains y and intersects with X. We say that y is independent from X if y is $|M|$-independent from X; otherwise we say that X *dominates* y. Finally, a *dominating set* is a set X of vertices that dominates every other vertex of the hypergraph.

3.1 Case 2a of Theorem 10

We provide an overview of the proof method. We begin by describing the method involved in the proof of Case 1 of Theorem 9, before suggesting how this can be amended for Case 2a of Theorem 10. Whilst we do not allow constants in our signatures, we do refer to those elements that have been questioned in the decision tree as *constants*.

For Case 1 of Theorem 9, we construct a certain decision tree to refute the FO sentence ψ. The questions of the decision tree fall into two categories: I) boolean questions on the truth of (extensional) relations R on the already witnessed constants, and II) questions that ask for a witness to already witnessed constants in Skolem relations S. In the latter case the potential witness may be one of the already witnessed constants, or it may be a new constant. The important point is that this decision tree is finite – of height h and never involving more than m constants – for, if it were not, it would imply the existence of an infinite model for ψ. It is relatively straightforward to turn this FO decision tree into a boolean decision tree, for each propositional

$\mathcal{C}_{\psi,n}$, of size at most $(\max\{m,n\})^h$, i.e., polynomial in n as claimed.

For Case 2a of Theorem 10, we construct a certain different decision tree to refute the FO sentence ψ in a parameterized setting. This decision tree adds new constants in pairs, under the additional assumption that the second new constant is independent from both the first new constant and the set of constants already witnessed. We are able to demonstrate that this tree is finite – of height h and never involving more than m constants – so long as all models of ψ have a finite dominating set. Again, we are able to turn this into a parameterized boolean decision tree, for each propositional $\mathcal{D}_{\psi,n,k}$, of size at most $(m^{abh})^k n^h$, where a is the maximum arity of any relation of ψ and b is the number of relations of ψ (including Skolem relations in both cases). The result follows.

We conclude this section with an example of Case 2a of Theorem 10. This specimen provides a somewhat trivial instance, having, as it does, parameterized tree-like resolution refutations not just polynomial in n, but actually independent of n. There are examples for Case 2a which are nontrivial insofar as there the size of a smallest parameterized tree-like refutation depends on n (see [5]).

Example 12. We consider the (negation of the) least number principle for total orders. Let ψ^{LNP_1} be the conjunction of the following.

$$\forall x\, \neg R(x,x) \quad \text{(antireflexivity)}$$
$$\forall x \forall y\, \neg R(x,y) \vee \neg R(y,x) \quad \text{(antisymmetry)}$$
$$\forall x \forall y \forall z\, \neg R(x,y) \vee \neg R(y,z) \vee R(x,z) \quad \text{(transitivity)}$$
$$\forall x \forall y R(x,y) \vee R(y,x) \quad \text{(totality)}$$
$$\forall y \exists x\, R(x,y) \quad \text{(no least element)}$$

All models of ψ^{LNP_1} have a dominating set of size 1; moreover, every element of the model constitutes such a dominating set. It is straightforward to verify that $\langle \mathcal{D}_{\psi^{\mathrm{LNP}_1},n,k}\rangle_{n\in\mathbb{N}}$ has parameterized tree-like resolution refutations of size $2k$, independent from n.

3.2 Case 2b of Theorem 10

We now turn our attention to proving Case 2b of Theorem 10. Our argument will be facilitated by a game based on those described by Pudlák [12] and Riis [13] in which *Prover* (female) plays against *Adversary* (male). In this game, a strategy for Prover gives rise to a parameterized boolean decision tree on a set of clauses. Prover questions the propositional variables that label the nodes of the tree and Adversary attempts to answer these so as neither to violate any specific clause nor to have conceded that more than k variables are true ($\top$), for in either of these situations Prover is deemed the winner. Of course, assuming the set of clauses was unsatisfiable, Adversary is destined to lose: the question is how large he can make the tree in the process of losing. Note that each branch of the tree corresponds to a play of this game, hence each parameterized decision tree corresponds to a Prover strategy. We will be concerned with Adversary strategies that perform well over all Prover strategies, and hence induce a lower bound on all parameterized decision trees and, consequently, all parameterized tree-like resolution refutations.

When considering a certain Prover strategy – a parameterized decision tree – we will actually consider only a certain subtree in which the missing branches correspond to places where Adversary has simply given up, already conceding the imminent violation of a clause. In this way, there are two types of non-leaf nodes in this subtree, those of out-degree 1 in which Adversary's decision was *forced* (because he conceded defeat on the alternative valuation) and those of out-degree 2 in which he is happy to continue on either outcome. In the latter case, we may consider that he has given Prover a *free choice* as to the value of the relevant variable. The free choice nodes play a vital role in ensuring the large size of this subtree, which in turn places a lower bound on the size of the parameterized decision tree of which it is a subset.

Let $\mathcal{C}_{\psi,n}$ be the propositional translation of some FO sentence ψ which has no finite models, but holds in some infinite model. We formally define the game $\mathcal{G}(\mathcal{C}_{\psi,n},k)$ as follows. At each turn Prover selects a propositional variable of $\mathcal{C}_{\psi,n}$ that she has not questioned before, and Adversary responds either by answering that the variable is true ($\top$) or that it is false ($\bot$), or by allowing Prover a free choice over those two. The Prover wins if at any point she holds information that contradicts a clause of $\mathcal{C}_{\psi,n}$ or she holds more than k variables evaluated true. In this formalism, given a Prover strategy on her moves, and considering both possibilities on the free choice nodes, we generate a *game tree*, the subtree of the parameterized decision tree alluded to in the previous paragraph.

Henceforth, we consider only the case in which some model of ψ has no finite dominating set. We will give a strategy for Adversary in the game $\mathcal{G}(\mathcal{C}_{\psi,n},k)$ that guarantees a large game tree for all opposing Prover strategies.

Adversary's Strategy At any point in the game – node in the game tree – Adversary will have conceded certain information to Prover. He always has in mind two disjoint sets of already mentioned constants P and Q on which he has conceded certain information: initially these sets are both empty. The set Q is to be a $(P \cup Q)$-independent set whose members are also $(P \cup Q)$-independent from P. In some sense P is the only set of constants for which Adversary has actually conceded an interpretation; all he concedes of Q is that it is a floating set with certain independence properties. If X is a set of constants, let $\mathcal{M}_X$ be the class of models of ψ that are consistent with the information Adversary has conceded on X. At each point Prover will ask Adversary a question of the form $R_i(\overline{c})$ or $S_j(\overline{c})$. The Adversary answers as follows:

I. If all constants of $\overline{c}$ are in P, then Adversary should choose some model in $\mathcal{M}_P$ and answer according to that.

II. If all constants of $\overline{c}$ are in $P \cup Q$, and there is at least one from Q, then Adversary should answer false ($\perp$).

III. If some constant in $\overline{c}$ is not in $P \cup Q$ then

   – if no model in $\mathcal{M}_P$ satisfies the question, then Adversary should answer false ($\perp$), otherwise

   – he should give Prover a free choice on the question.

In all cases the sets P and Q remain the same, except in Case III Part 2. If the Prover chooses true ($\top$), then Adversary places all the constants of $\overline{c}$ in P, possibly removing some from Q in the process. If the Prover chooses false ($\perp$), then Adversary places any constants in $\overline{c}$ that are not already in $P \cup Q$ into Q. It turns out that, in Cases II and III, the situation never arises in which Adversary is forced to answer true. In particular, in Case III, it will never be the case that all models in $\mathcal{M}_P$ satisfy the question. This is vital to the success of Adversary's strategy, and we will return to it later. We must now prove that this strategy leads to a large parameterized decision tree; we will need the following lemma.

Lemma 13. *Consider any path in the game tree of $\mathcal{G}(\mathcal{C}_{\psi,n}, k)$ from the root to a leaf. If there are k or fewer propositional variables evaluated to true by the leaf, then every one of the n constants must have appeared in a free choice node along that path.*

Proof. We give a sketch proof of the lemma; for a fuller explanation, see Riis's paper [13]. It is important to see that Adversary plays faithfully according to some (infinite) models of ψ, because this means that an elementary contradiction can only be reached by the violation of a Skolem clause. In order to see that Adversary plays so, it becomes necessary to explain why in Case II of his strategy he never loses any of his putative models $\mathcal{M}_P$ and why in Case III he is never forced to answer true ($\top$).

In Case II, Adversary never loses a model M in $\mathcal{M}_P$ because Q can always be chosen to be independent, and independent from P. Indeed, if such an interpretation is put on Q in M, then Adversary's answer is forced to be false ($\perp$).

Suppose, in Case III, that Adversary were forced to answer true ($\top$), i.e., all models M in $\mathcal{M}_P$ satisfy the question $R_i(\overline{c})$ or $S_j(\overline{c})$. By the floating nature of all elements that are not in P this would generate a finite dominating set of $P \cup Q$ on M. Let us dwell on this point further. Let $\overline{c}'$ be the subtuple of $\overline{c}$ consisting of those constants of the latter that are not in $P \cup Q$. Some of the constants of $\overline{c}'$ could have been mentioned in questions before, but only in ones for which Adversary's response had been forced false. Suppose that $P \cup Q$ were not a dominating set for M, then there exists an element $x \in M$, independent from $P \cup Q$. But this element is such that it can fill the tuple $\overline{c}'$ and falsify $R_i(\overline{c})$ or $S_j(\overline{c})$ in M (and falsify any questions that previously involved it, which had already been answered false). This contradicts the question having been forced true in the first place.

Recalling that we can only reach an elementary contradiction by the violation of a Skolem clause, we can now complete the proof. Let c' be a constant that never appears in a free choice node in our game tree. In order to violate a Skolem clause, Adversary must have denied some $S(\overline{c}, x)$, for each of the n constants substituted for x. But that his denial of $S(\overline{c}, c')$ was forced implies a contradiction. Since c' is uninterpreted in any of the models in $\mathcal{M}_P$, it follows that $S(\overline{c}, c'')$ is false for all c'' in any model in $\mathcal{M}_P$. This tells us that $\mathcal{M}_P$ is empty and, consequently, that ψ had no infinite model. □

We are now in a position to argue the key lemma in this section.

Lemma 14. *Let a be the maximum arity of any relation in ψ and suppose that there are no more than b different relations in the propositional translation of ψ (including Skolem relations in both cases). Following the strategy that we have detailed for the game $\mathcal{G}(\mathcal{C}_{\psi,n}, k)$, and with p and q the cardinality of the sets P and Q, respectively, Adversary cannot lose while both $p < k^{1/ab}$ and $p + q < n$.*

Proof. Consider the game tree of $\mathcal{G}(\mathcal{C}_{\psi,n}, k)$. Note that Adversary only answers true in the case that all involved constants are then added to his set P, or, of course, were already there. Thus, at a certain node in the game tree, the number of true answers given is trivially bounded by the size of the set of all possible questions on P, which is certainly bound by p^{ab}. Hence, whilst $p^{ab} < k$, there must be fewer than k propositional variables evaluated to true. Furthermore, if $p + q < n$ at this node, then not all of the n constants can have appeared in a free choice (since constants that have appeared in a free choice are necessarily added to either P or Q). It follows from the previous lemma that Adversary has not yet lost. □

We are now in a position to settle Case 2b.

Proof of Case 2b, Theorem 10. We aim to provide a lower bound on the size of any game tree for $\mathcal{G}(\mathcal{C}_{\psi,n}, k)$. Since a lower bound on the size of a game tree induces a lower bound on the size of a parameterized boolean decision tree, the result follows.

Consider a game tree for $\mathcal{G}(\mathcal{C}_{\psi,n}, k)$. Recall that, at any node in this tree, Adversary has in mind two sets P and Q, of size p and q, respectively, and, by the previous lemma, whilst $p < k^{1/ab}$ and $p + q < n$, he has not lost. Consider, therefore, any node in this game tree and the sets P and Q that Adversary there has in mind. Let $S(p, q)$ be some monotonic decreasing function that provides a lower bound on the size of the subtree of the game tree rooted at the chosen node; whence $S(0, 0)$ is a lower bound on the size of the game tree itself. In showing that $S(p, q)$ satisfies the recurrence relation

- $S(p, q) \geq S(p + a, q) + S(p, q + a) + 1$, with
- $S(p, q) \geq 0$, when $p \geq k^{1/ab}$ or $p + q \geq n$,

we are able to derive the following statement.

Let n, k, a and b be positive integers such that (i.) $a \geq 2$; (ii.) $n > k$; (iii.) $n \geq 7a + 1$; (iv.) $k^{1/ab} \geq (16a^2)^2$; then

$$S(0, 0) \geq n^{k^\gamma} \text{ where } \gamma := 1/16a^3b.$$

The result follows immediately from this statement for sufficiently large k ($\geq (16a^2)^{2ab}$) and n ($\geq 7a+1$). By noting that all parameterized boolean decision trees of Case 2b are of size ≥ 2, we can modify the given γ to one that works for all $n, k \geq 1$. Note that the assumption that (maximum arity) $a \geq 2$ is innocuous – there are no unary FO sentences φ which have no finite models but possess an infinite one, therefore we would be in neither Case 2a nor Case 2b. □

Example 15. We consider the (negation of the) least number principle for partial orders. Let $\psi^{\mathrm{LNP}_\infty}$ be the conjunction of the FO clauses given in Example 12 without the fourth clause (totality). $\psi^{\mathrm{LNP}_\infty}$ has models without a finite dominating set. For example, if $\mathbb{Z}$ is the set of integers, then $\mathbb{N} \times \mathbb{Z}$ under the strict partial ordering

$$(n, z) \prec (n', z') \text{ if and only if } n = n' \text{ and } z < z'$$

provides such a model.

Example 16. We return to the sentence ψ^{PHP} defined in Example 5. This has models without a finite dominating set: for example the positive integers $\mathbb{N}$, with $R(x, y) \Leftrightarrow y = x + 1$, provides such a model.

4 Embedding into Ordinary Proof Systems

Given a parameterized contradiction $(\mathcal{F}, k)$ we may attempt to derive an (ordinary) contradiction $\mathcal{F}'$ by directly axiomatizing the fact that no more than k variables of $\mathcal{F}$ may be set to true. We may then use an ordinary proof system to refute $\mathcal{F}'$. Considering the parameter preserved, we obtain from this embedding a new parameterized proof system. Formally, let PCON and CON be the classes of parameterized contradictions and (ordinary) contradictions, respectively. Let e : PCON $\rightarrow$ CON be some injection such that the range of e, and e^{-1} on that range, are polynomial-time computable. let Σ_1 be some proof alphabet and let $\Gamma : \Sigma_1^* \rightarrow$ CON be a proof system for CON. It follows that $\Gamma' : \Sigma_1^* \times \mathbb{N} \rightarrow$ PCON given by

$$\Gamma'(w, k) := \begin{cases} (\mathcal{F}, k) & \text{if } \Gamma(w) \text{ in range of } e \\ & \text{and } (\mathcal{F}, k) = e^{-1}(\Gamma(w)); \\ (\mathcal{F}_\perp, k) & \text{otherwise.} \end{cases}$$

is a parameterized proof system (where $\mathcal{F}_\perp$ is some fixed contradiction, say $v \wedge \neg v$).

Naive embeddings Suppose the variables of $\mathcal{F}$ are $v_1, \ldots, v_n$; it follows that the size of $\mathcal{F}$ is at least n. We might try to incorporate the set $\mathcal{N}_k$ (respectively, $\mathcal{N}_k'$) of all clauses involving more than k (respectively, exactly $k + 1$) negated variables. Both of these fail – though the latter less spectacularly – since the function given by $(\mathcal{F}, k) \mapsto (\mathcal{F} \cup \mathcal{N}_k)$ (respectively, $(\mathcal{F}, k) \mapsto (\mathcal{F} \cup \mathcal{N}_k')$) is not fpt-bounded. This is because both $\mathcal{N}_k$ and $\mathcal{N}_k'$ are of size $\geq n^{k+1}$. Consequently all proofs in this proof system fall into the "hard" category with size at least n^{k+1}.

Embedding using auxiliary variables Another possibility involves the use of new auxiliary variables $q_{v_i,j}$ for $i \in [n]$ and $j \in [k]$. We now add pigeonhole clauses $\neg v_i \vee \bigvee_{l=1}^{k} q_{v_i,l}$ and $\neg q_{v_i,j} \vee \neg q_{v_{i'},j}$ for $i, i' \in [n]$ $(i \neq i')$ and $j \in [k]$. Denote this set of clauses by $\mathcal{N}_k''$. These clauses essentially specify a weak pigeonhole principle from n to k and it is fairly straightforward to see that they can only be satisfied if no more than k of the variables v_i is true.

This method of auxiliary variables results in a parameterized proof system whose behavior with respect to tree-like resolution is similar to that of parameterized tree-like resolution. Since the clauses $\mathcal{N}_k'$ can be derived from these axioms in a subtree of size $2^{k!}$, the "easy" case (2a) is preserved, up to a possible factor of $2^{k!}$. Also the "hard" case (2b) remains via the same proof.

We have not defined a system of parameterized resolution, but such a definition would be a straightforward generalization. It is not clear what the complexity of the pigeonhole principle would be in this system, but we can settle the complexity of the pigeonhole principle when embedded into resolution via the method of auxiliary variables. Recalling that the pigeonhole principle falls in the "hard" case (2b) for parameterized tree-like resolution (and also when embedded into tree-like resolution via the method of auxiliary variables), it is perhaps surprising that the pigeonhole principle falls into the "easy" case (2a) when embedded into resolution.

Proposition 17. *Using the method of auxiliary variables, there is a resolution refutation of the (negation of the) pigeonhole principle of size* $2^k n^2$.

Proof. Note that the case $k \geq n$ is straightforward; assume that $k < n$. We recall from Example 5 that the axioms are $\mathcal{F} := \mathcal{C}_{\psi^{\text{PHP}},n} =$

$$\neg S_2(i,j) \vee R(i,j),\ \neg S_1(j) \vee \neg R(i,j) \text{ and}$$
$$\neg R(i,j) \vee \neg R(i',j), \text{ for } i,i',j \in [n],\ i \neq i',$$

$$\bigvee_{j=1}^{n} S_2(i,j), \text{ for } i \in [n], \text{ and } \bigvee_{i=1}^{n} S_1(i).$$

Let V be the set of variables appearing in these axioms. We now add the auxiliary clauses $\mathcal{N}_k'' :=$

$$\neg\alpha \vee \bigvee_{l=1}^{k} q_{\alpha,l} \text{ and } \neg q_{\alpha,j} \vee \neg q_{\alpha',j}$$

for $\alpha, \alpha' \in V$, $\alpha \neq \alpha'$, and $j \in [k]$. It is worth noting that, since $k < n$, the clauses $\neg S_1(j) \vee \neg R(i,j)$ and $\bigvee_{i=1}^{n} S_1(i)$ are not needed for a resolution refutation.

In order to generate a resolution refutation of $\mathcal{F} \cup \mathcal{N}_k''$ we will consider the behavior of some further new variables. For $i \in [n]$ and $j \in [k]$, define:

$$r_{ij} \equiv \bigvee_{l=1}^{n} q_{R(i,l),j}$$

It is not hard to see that the variables r_{ij} themselves specify a weak pigeonhole principle from n to k and it is this property that we will exploit. Consider the set of clauses $\mathcal{F}'' :=$ $(\neg r_{ij} \vee \neg r_{i'j})$ and $\bigvee_{j=1}^{k} r_{ij}$, for $i,i' \in [n]$, $i \neq i'$, and $j \in [k]$. It is known that there exists a resolution refutation $\mathcal{F}''$ of size 2^k such that no clause (other than the axioms) contains more than one negated variable [1]. We will convert this refutation into one for $\mathcal{F} \cup \mathcal{N}_k''$ of size at most $2^k n^2$.

First we will show how to derive any axiom of $\mathcal{F}''$ from $\mathcal{F} \cup \mathcal{N}_k''$. The axioms $\neg r_{ij} \vee \neg r_{i'j}$ are already present as n^2 different axioms of $\mathcal{N}_k''$:

$$\begin{aligned} \neg r_{ij} \vee \neg r_{i'j} &\equiv \textstyle\bigwedge_{l=1}^{n} \neg q_{R(i,l),j} \vee \bigwedge_{l'=1}^{n} \neg q_{R(i',l'),j} \\ &\equiv \textstyle\bigwedge_{l=1}^{n} \bigwedge_{l'=1}^{n} (\neg q_{R(i,l),j} \vee \neg q_{R(i',l'),j}) \end{aligned}$$

The axioms $\bigvee_{j=1}^{k} r_{ij} \equiv \bigvee_{j=1}^{k} \bigvee_{l=1}^{n} q_{R(i,l),j}$ may be generated only a little more circuitously. The axiom $\bigvee_{j=1}^{n} R(i,j)$ may be derived by resolving $\bigvee_{j=1}^{n} S_2(i,j)$ with n instances of $\neg S_2(i,j) \vee R(i,j)$, i.e., $1 \leq j \leq n$. Now this can be resolved with n instances of $\neg R(i,j) \vee \bigvee_{l=1}^{k} q_{R(i,j),l}$, i.e., $1 \leq j \leq n$.

We now demonstrate how one may simulate a resolution step on the $\mathcal{F}''$ clauses in the $\mathcal{F} \cup \mathcal{N}_k''$ clauses. For this part it is crucial that the resolution on $\mathcal{F}''$ contains no clauses with more than two negated literals. We will first consider the simplest case in which one of the clauses to be resolved is strictly positive and the other contains a single negated variable, that is they are of the form:

$(r_{i_1j_1} \vee r_{i_2j_2} \vee \ldots \vee r_{i_tj_t}) \equiv$
$\bigvee_{l=1}^{n} q_{R(i_1,l),j_1} \vee \bigvee_{l=1}^{n} q_{R(i_2,l),j_2} \vee \cdots \vee \bigvee_{l=1}^{n} q_{R(i_t,l),j_t}$

and

$(\neg r_{i_1j_1} \vee r_{i_2'j_2'} \vee \ldots \vee r_{i_{t'}'j_{t'}'}) \equiv$
$\bigwedge_{l=1}^{n} \neg q_{R(i_1,l),j_1} \vee \bigvee_{l=1}^{n} q_{R(i_2',l),j_2'} \vee \cdots \vee \bigvee_{l=1}^{n} q_{R(i_{t'}',l),j_{t'}'}$

It is clear that the second of these is equivalent to (and may be simulated by) the system of n clauses

$$\begin{gathered} \neg q_{R(i_1,1),j_1} \vee \textstyle\bigvee_{l=1}^{n} q_{R(i_2',l),j_2'} \vee \cdots \vee \bigvee_{l=1}^{n} q_{R(i_{t'}',l),j_{t'}'} \\ \vdots \\ \neg q_{R(i_1,n),j_1} \vee \textstyle\bigvee_{l=1}^{n} q_{R(i_2',l),j_2'} \vee \cdots \vee \bigvee_{l=1}^{n} q_{R(i_{t'}',l),j_{t'}'} \end{gathered}$$

It should be clear that even the extreme case, of two negated literals in each clause, may be simulated by a system of n^2 clauses.

Each clause in the resolution refutation of $\mathcal{F}''$ may now be replaced by at most n^2 clauses to obtain a refutation of $\mathcal{F} \cup \mathcal{N}_k''$, and the result follows. ☐

It may be noted that we could have defined $r_{ij} := \bigvee_{l=1}^{n} q_{S_2(i,l),j}$ in the proof of the previous proposition. The reason we have used the $q_{R(i,l),j}$ variables is to show that the result stands for the more usual encoding of the pigeohole principle, which avoids Skolem relations. However, our method can be used to demonstrate that any first-order ψ, without finite models, that translates to a propositional system involving at least one non-unary Skolem relation, has a resolution refutation (using the method of auxiliary variables) of size $2^k n^2$. It is straightforward to show, if ψ has no finite models and a propositional translation without a non-unary Skolem relation, that ψ also has no infinite models. Therefore, the method of auxiliary variables has made all of our parameterized contradictions "easy" for resolution. We note that not all contradictions derive from first-order principles, and that this method of auxiliary variables may have more relevance elsewhere.

References

[1] S. R. Buss and T. Pitassi. Resolution and the weak pigeonhole principle. In *CSL '97: Selected Papers from the 11th International Workshop on Computer Science Logic*, Lecture Notes in Computer Science 1414, Springer (1998) 149–156

[2] Cesati, M.: *Compendium of parameterized problems.* http://bravo.ce.uniroma2.it/home/cesati/research/compendium.pdf (September 2006)

[3] J. Chen, I. A. Kanj, and G. Xia. Improved parameterized upper bounds for vertex cover. In *Proceedings of the International Symposium on Mathematical Foundations of Computer Science (MFCS 2006)*, Lecture Notes in Computer Science 4162, Springer (2006) 238–249

[4] S. A. Cook and R. A. Reckhow. The relative efficiency of propositional proof systems. *J. Symbolic Logic*, 44(1) (1979) 36–50

[5] S. Dantchev and B. Martin and S. Szeider, Parameterized proof complexity: a complexity gap for parameterized tree-like resolution. Electronic Colloquium on Computational Complexity (ECCC), Technical Report TR07-001 (2007)

[6] Downey, R.G., Fellows, M.R.: *Parameterized Complexity.* Monographs in Computer Science. Springer (1999)

[7] Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness. II. On completeness for $W[1]$. Theoretical Computer Science 141(1-2) (1995) 109–131

[8] Flum, J., Grohe, M.: *Parameterized Complexity Theory.* Volume XIV of Texts in Theoretical Computer Science. An EATCS Series. Springer (2006)

[9] Impagliazzo, R., Paturi, R., Zane, F.: Which problems have strongly exponential complexity? *J. of Computer and System Sciences* 63(4) (2001) 512–530

[10] Krajíček, J.: *Bounded Arithmetic, Propositional Logic, and Complexity Theory.* Cambridge University Press, New York, NY, USA (1995)

[11] Niedermeier, R.: *Invitation to Fixed-Parameter Algorithms.* Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press (2006)

[12] Pudlák, P.: Proofs as Games. *American Mathematical Monthly*, 107(6) (2000) 541–550

[13] Riis, S.: A complexity gap for tree-resolution. *Computational Complexity* 10(3) (2001) 179–209