comput. complex. **20** (2011), 51 – 85
1016-3328/11/010051-35
published online April 5, 2011
DOI 10.1007/s00037-010-0001-1

computational complexity

PARAMETERIZED PROOF COMPLEXITY

Stefan Dantchev, Barnaby Martin, and Stefan Szeider

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 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[SAT] by showing that there is no fpt-bounded parameterized proof system for parameterized contradictions, 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 subcases, 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.

Keywords. Propositional proof complexity; parameterized complexity; complexity gap theorems.

Subject classification. 68Q25.

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 Cesati (2006), Downey & Fellows (1999), Flum & Grohe (2006), Niedermeier (2006)). In parameterized complexity, one considers computational problems in a two-dimensional setting: the first dimension is the usual *input size* n and 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 (Chen *et al.* 2006). 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 SAT is a further problem that is believed to be *not* fixed-parameter tractable (and which plays an important role in the sequel): given a propositional formula, 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* $\mathrm{W}[1] \subseteq \mathrm{W}[2] \subseteq \cdots \subseteq \mathrm{W}[\mathrm{SAT}]$. For example, CLIQUE is complete for W[1] and BOUNDED SAT is complete for the class W[SAT]. The restriction of BOUNDED SAT to formulas in conjunctive normal form (CNF) is complete for W[2]. 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 (Flum & Grohe 2006).

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 (Cesati 2006).

2. *k-step halting problems* for non-deterministic Turing machines are complete for the classes W[1] (single-tape) and W[2] (multi-tape) (see, e.g., Flum & Grohe 2006). A Turing machine is such an opaque and generic object that it does not appear reasonable that we should be able to decide whether 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 (Impagliazzo *et al.* 2001). ETH is closely related to the parameterized complexity class M[1], which lies between FPT and W[1] (see Flum & Grohe 2006).

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 & Reckhow (1979). 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 pigeonhole principle) where the n-th formula of the sequence states that the FO sentence has no model of size n. Riis (2001) 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 $2^{\epsilon n}$ 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 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[SAT]-complete language. Hence, FPT = co-W[SAT] = W[SAT] 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 system of resolution, in fact predominantly its weaker, tree-like version. 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 an 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.

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 $\gamma, 0 < \gamma \leq 1$, 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}.

The latter case prevails exactly when ψ has a 'sparse' model, that is without a finite dominating set. 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 (Example 3.11 and Example 3.12) 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 $2^k n^2$.

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

$$f(k)|I|^{O(1)} \tag{2.1}$$

where f denotes a computable function. FPT denotes the class of all fixed-parameter tractable decision problems; algorithms that achieve the time complexity (2.1) 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

$$|I|^{O(f(k))}. \tag{2.2}$$

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 $R(I, k) = (I', k')$, $k' \leq g(k)$.

A parameterized complexity class $\mathcal{C}$ is a class of parameterized problems closed under fpt reductions. It may easily be verified that FPT is a parameterized complexity class. Parameterized problems appear to have several degrees of intractability, as manifested by the *weft hierarchy*. The classes of this hierarchy form a chain

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

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 $\text{FPT} \neq \text{XP}$ (Downey & Fellows 1999). Each class W[t] (here and henceforth, t may be a positive integer or SAT) is defined as the class of problems fpt-reducible to a certain canonical weighted satisfiability problem for decision circuits. For W[SAT], the canonical problem is equivalent to the following satisfiability problem:

> WEIGHTED SAT
>
> *Instance:* A propositional formula $\mathcal{F}$ 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.)

Let BOUNDED SAT denote the problem obtained from WEIGHTED SAT by allowing truth assignments of weight *at most* k. Since WEIGHTED SAT remains W[SAT]-complete for antimonotone formulas (see, e.g., Flum & Grohe 2006), we may deduce the following.

LEMMA 2.3. BOUNDED SAT *is* W[SAT]*-complete under fpt reductions.*

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 2.4. *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 $\Gamma(I', k') = (I, k)$, $|I'| \leq f(k)|I|^{O(1)}$, 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[SAT]-complete problem $L =$ BOUNDED SAT. Let $S_{\mathcal{F},\tau,k}$ denote a string over some alphabet Σ_1 that encodes a 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[SAT]. We can witness that a 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 2.5. *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.

PROOF. Let $L \subseteq \Sigma^* \times \mathbb{N}$ be a co-$\mathcal{C}$-complete parameterized problem. We show the contra-positive of the statement. Assume $\mathcal{C} =$ FPT. Since FPT $=$ co-FPT, co-$\mathcal{C} =$ FPT follows. Consequently, there is a fixed-parameter algorithm that decides membership in L; let M be a Turing machine that implements this algorithm. For $(I,k) \in L$, let $M_{(I,k)}$ be a string over some alphabet Σ_1 that encodes the computation steps of M with input (I,k). By the fixed-parameter tractability of L, there is a computable function f such that $|M_{(I,k)}| \leq f(k)|I|^{O(1)}$. We may assume that (I,k) can be read off from $M_{(I,k)}$, say, by choosing an encoding where (I,k) is encoded as a prefix of $M_{(I,k)}$ where k is presented in unary. We define a mapping $\Gamma : \Sigma_1^* \times \mathbb{N} \to L$ as follows. Consider $(I',k') \in \Sigma_1^* \times \mathbb{N}$. If I' encodes a computation of M for the input (I,k), i.e. if $I' = M_{(I,k)}$, then we let $\Gamma(I',k') = (I,k)$. Otherwise, if (I',k') does not encode a computation of M for some input (I,k), we put $\Gamma(I',k') = (I_0,k_0)$ for some arbitrary fixed $(I_0,k_0) \in L$. Clearly, Γ is a proof system for L as $\Gamma(I',k')$ can be computed in linear time. Furthermore, Γ is fpt-bounded, since $|M_{(I,k)}| \leq f(k)|I|^{O(1)}$ holds for $(I,k) \in L$. $\square$

In view of this lemma, we suggest a program à la Cook-Reckhow for gaining evidence that 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.

2.3. 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.

We consider propositional formulas $\mathcal{F}$ only over the connectives $\wedge, \vee$ and $\neg$. In order to consider resolution refutations of these, it is necessary to describe a canonical translation to CNF (i.e., a set of clauses). We may consider a formula $\mathcal{F}$ to be represented as an unranked in-tree circuit with literals for leaves. We may imagine strict alternation from the root between $\wedge$- and $\vee$-gates, and for technical reasons, also that every path from the root to a leaf is of length at least 2 and culminates in a $\vee$-gate above that leaf. We describe an inductive translation of $\mathcal{F}$ to a formula in CNF. If $\mathcal{F}$ is already in CNF, we leave it unchanged. If the root gate in $\mathcal{F}$'s representation is $\wedge$, then we inductively convert each of the top conjuncts to CNF and take the union of their clauses. If the root node in $\mathcal{F}$'s representation is $\vee$, with r inputs, then we introduce new propositional variables $S_{\mathcal{F},1}, \ldots, S_{\mathcal{F},r}$ and set $\mathcal{F} := \mathcal{F}' \wedge (S_{\mathcal{F},1} \vee \ldots \vee S_{\mathcal{F},r})$, where $\mathcal{F}'$ is obtained from $\mathcal{F}$

by changing the root gate $\vee$ to $\wedge$ and adding to each disjunct at distance 2 from the root, on a branch from the root's ith input, the literal $\neg S_{\mathcal{F},i}$. Let $\mathrm{CNF}(\mathcal{F})$ be the formula generated from $\mathcal{F}$ through this translation. It is not hard to see that 1.) $\mathrm{CNF}(\mathcal{F})$ is in CNF and is computed in polynomial time, 2.) $\mathcal{F}$ is satisfiable iff $\mathrm{CNF}(\mathcal{F})$ is satisfiable, and 3.) $\mathcal{F}$ is satisfiable with weight $\leq k$ iff $\mathrm{CNF}(\mathcal{F})$ is satisfiable with weight $\leq k$ on the variables it shares in common with $\mathcal{F}$. We refer to the variables that occur in $\mathrm{CNF}(\mathcal{F})$ but not $\mathcal{F}$ as *S-variables*. Since we will principally be concerned with propositional formulas that uniformly encode FO sentences, we defer an example of the translation $\mathcal{F} \mapsto \mathrm{CNF}(\mathcal{F})$ until later.

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. 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 1995): given a truth assignment, find a clause that 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.

Definition 2.6. *A parameterized contradiction is a pair* $(\mathcal{F}, k)$ *where* $\mathcal{F}$ *is a propositional formula and* k *is a positive integer such that* $\mathcal{F}$ *has no satisfying assignment of weight at most* k*.*

Example 2.7. 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

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

is a parameterized contradiction. ◊

Let PAR CONT be the language of parameterized contradictions. Note that PAR CONT is the complement of BOUNDED SAT and, as such, is co-W[SAT]-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 2.8. *Given a parameterized contradiction* $(\mathcal{F}, k)$*, a* parameterized boolean decision tree *is a decision tree that queries values of the propositional variables of* CNF$(\mathcal{F})$ *and branches on the answers; a branch of the tree is closed as soon as (1) or (2) happens:*

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

(2) the partial assignment obtained along the branch has more than k propositional variables of $\mathcal{F}$ 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 of $\mathcal{F}$. 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.

2.4. From First Order to Propositional Logic. Next, we describe a translation of an FO sentence to a propositional formula. 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. Let $[n] = \{1, 2, \ldots, n\}$. It will be convenient to assume that the FO sentence is given as a conjunction of prenex sentences, whereupon we may just explain the translation of a prenex sentence. A sentence ψ of the form

$$\forall x_1 \exists y_1 \ldots \forall x_k \exists y_k \ \mathcal{F}(x_1, \ldots, x_k, y_1, \ldots, y_k), \tag{2.9}$$

becomes the formula $\mathcal{C}_{\psi,n}$

$$\bigwedge_{x_1 \in [n]} \bigvee_{y_1 \in [n]} \cdots \bigwedge_{x_k \in [n]} \bigvee_{y_k \in [n]} \mathcal{F}(x_1, \ldots, x_k, y_1, \ldots, y_k),$$

where each instance $R(z_{i_1}, \ldots, z_{i_p})$ of a p-ary predicate symbol in $\mathcal{F}$ becomes a propositional variable, and any instances $(z = z')$ in $\mathcal{F}$ are evaluated to either true or false. By construction, for an FO sentence $\psi, \mathcal{C}_{\psi,n}$ is satisfiable iff 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. Note that the sequence $\langle \mathrm{CNF}(\mathcal{C}_{\psi,n})\rangle_{n\in\mathbb{N}}$ is identical, up to a relabeling of the new S-variables, with the sequence of CNFs obtained from ψ by the process of *Skolemization*. A sentence of the form (2.9) may be transformed into the following purely universal sentence

$$\forall x_1, \dots x_k, y_1, \dots y_k \bigvee_{i=1}^{k} \neg S_i(x_1, y_1 \dots x_i, y_i) \vee \mathcal{F}(x_1, \dots x_k, y_1, \dots y_k),$$

together with the Skolem system

$$\forall x_1, y_1 \dots, x_i \exists y_i \; S_{i+1}(x_1, y_1, \dots, x_i, y_i), \text{ for } i \in [k],$$

where we refer to the new S-relations as *Skolem*. For the first sentence, as $\mathcal{F}$ is in CNF, we may propagate the k-disjunction of negated Skolem relations into the clauses of $\mathcal{F}$ to obtain a system of clauses for all instantiations of $x_1, \dots x_k, y_1, \dots y_k$. Each of the following sentences, for each instantiation of $x_1, y_1 \dots, x_i$, gives rise to an n-disjunct—a *Skolem clause*—asserting the existence of a witness for the variable y_i. These clauses correspond to the clauses of positive S-variables that we introduced in the translation of Section 2.3 when changing the $\vee$-gate to a $\wedge$. In the case of CNFs born of translation from an FO sentence ψ, we will always generate $\mathrm{CNF}(\mathcal{C}_{\psi,n})$ directly by Skolemization, as this is closer to the translation method of Riis (2001), and names the S-variables much more naturally. We henceforth refer to the S-variables of $\mathrm{CNF}(\mathcal{C}_{\psi,n})$ as *Skolem variables*.

REMARK 2.10. *Note that, for ψ fixed, the sizes of both $\mathcal{C}_{\psi,n}$ and* $\mathrm{CNF}(\mathcal{C}_{\psi,n})$*, with respect to some reasonable encoding, are polynomial in n.*

EXAMPLE 2.11. 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 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 system

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

For $x, y \in [n]$, we now consider $R(x,y), S_2(x,y)$ and $S_1(y)$ to be propositional variables. $\mathrm{CNF}(\mathcal{C}_{\psi^{\mathrm{PHP}},n})$ 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

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

◊

3. Complexity Gap for Parameterized Tree-like Resolution

We first recall the complexity gap theorem for tree-like resolution proved by Riis (2001).

Theorem 3.1. *Given an FO sentence ψ which fails in all finite models, consider its translation into a sequence of propositional contradictions $\langle \mathrm{CNF}(\mathcal{C}_{\psi,n}) \rangle_{n \in \mathbb{N}}$. Then, either 1 or 2 holds:*

1. $\mathrm{CNF}(\mathcal{C}_{\psi,n})$ *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 $\mathrm{CNF}(\mathcal{C}_{\psi,n})$ is of size at least $2^{\epsilon n}$.*

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

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 $\gamma, 0 < \gamma \leq 1$, 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}.*

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 Example 3.11 and Example 3.12), and bearing in mind Remark 2.10, we derive the following as a corollary.

COROLLARY 3.3. *Parameterized tree-like resolution is not fpt-bounded.*

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

Before we prove Theorem 3.2, we need to give some definitions. For a model M, let $|M|$ denote the universe of M. Given a model M of an 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. A set of vertices is *independent* if it contains no hyperedge as a subset. Given a set X of vertices, and a vertex $y \notin X$, we say that *y is independent from X* if and only if there is no hyperedge E such that $y \in E \subseteq X \cup \{y\}$. If X is not independent from y, then we say that X *dominates* y. Finally, a *dominating set* is a set X of vertices that dominates every other vertex of the hypergraph.

Remark 3.4. *Our criterion for domination may appear somewhat esoteric. Note, for example, that a vertex y with a self-loop is dominated by any set X.*

3.1. Case 2a of Theorem 3.2. We now prove Case 2a of Theorem 3.2. We shall start by reproving Case 1 of Theorem 3.1. Note that our proof is different from Riis's proof (Riis 2001) as our translation, though equivalent, is slightly different.

Proof. (of **Case 1, Theorem 3.1**) The idea is to take a (finite) resolution refutation of the FO formula ψ (such a refutation exists as the formula has no model) and to transform it into a polynomial size in n tree-like resolution refutation of $\mathcal{C}_{\psi,n}$.

As we have explained, we can consider a boolean decision tree instead of a tree-like resolution refutation. In the FO case, constructing a boolean decision tree is very similar to producing a tableau refutation. (Our method therefore differs slightly from simply inverting the classical FO resolution, as we consider only instantiations of terms as opposed to terms themselves.) The decision tree tries to build up a model of ψ, starting by witnessing some unary Skolem relation ψ with the constant 1 and deriving further constants as Skolem witnesses of already derived constants as and when necessary. (Note that we tend to discount the empty model. It is, therefore, possible to have ψ with no finite models and no outermost existential quantifier. In this case, we may instantiate a single constant at the outset to get us going.)

Note that, while we do not allow constants in our signatures, we refer to those elements that have been mentioned in decision tree questions as constants.

Let C be the set of constants thus far witnessed, and let $\overline{c}$ be some tuple over C. At each point, two kinds of queries are allowed: (I) querying the boolean value of some $R_i(\overline{c})$ and (II) querying the witness y of some $S_j(\overline{c}, y)$. In the latter case, there are two possibilities for y: it could be a constant that is already known or it could be a new one, thus extending the set of constants. For Case I, the branching factor is 2: corresponding to $R_i(\overline{c})$ being true ($\top$) or false ($\perp$). For Case II, the branching factor is $|C|+1$: we label these branches with the elements of C or a new constant c' according to the conceded witness for $S_j(\overline{c}, y)$.

The order in which the boolean decision tree performs these queries is as follows. We start with the single constant 1, witnessing a unary Skolem relation of ψ, i.e., set $C := \{1\}$, and first query all possible R_i relations on all possible tuples over C, closing any branch as soon as a contradiction is reached. We then pick up a Skolem relation $S_j(\overline{c}, y)$ and a j-tuple $\overline{c}$ of constants of C and query the witness y. There are $|C|+1$ possible outcomes—y is either one of the already known constants from C or a different constant, which we denote by c'. If $y \in C$, we pick another $S_{j'}(\overline{c}', y)$ and do the same (we assume a reasonable order over the Skolem relations S_j and tuples in C). In the case where y is a new constant which is not in C, we extend the set of constants, i.e., set $C := C \cup \{c'\}$ and repeat the same procedure, i.e., query all possible R_i relations over all possible tuples in the expanded C and so on.

It is easy to see that the decision tree constructed in this way is finite. Indeed, suppose it were infinite. Then, by König's Lemma, there must be an infinite branch which constitutes an infinite model of ψ—a contradiction. Let the depth of this tree be h and the maximum size of C along any of its branches be m. Let us now turn this finite refutation of ψ into a polynomial size in n refutation of CNF($\mathcal{C}_{\psi,n}$). We note that a node, which queries an R_i relation in the FO case, remains the same in the propositional case, and, in particular, has a branching factor 2. A node, which witnesses a Skolem relation $S_j(\overline{c}, y)$, is of constant branching factor in the FO case (bounded by m). In the propositional case, such a node can be translated into a sequence of n nodes, the l-th node querying the $S_j(\overline{c}, l)$ only if all the nodes $S_j(\overline{c}, 1), S_j(\overline{c}, 2), \ldots, S_j(\overline{c}, l-1)$ got negative answers. If the answers to all queries were negative, we arrive at a contradiction with the Skolem clause $\bigvee_{y=1}^{n} S_j(\overline{c}, y)$, while a positive answer gives us the desired witness. Thus, a node querying a Skolem relation in the FO case can be thought as a single node of branching factor n in the propositional case. As the

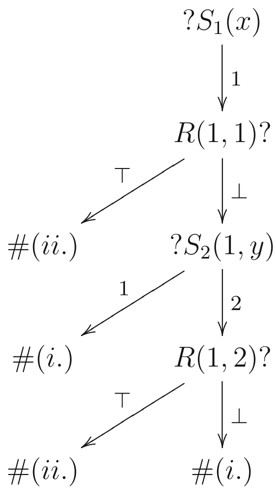

Figure 3.1: Decision tree for Example 3.5.

FO tree is of constant height h that depends on the formula ψ only, the boolean decision tree in the propositional case is of size at most $(\max\{m, n\})^h$ which is $O(n^h)$, i.e., polynomial in n as claimed. □

Example 3.5. We give an example of a decision tree constructed as in Case 1, Theorem 3.1. We consider the following sentence ψ which has no models:

$$\forall x \exists y\ R(x, y) \quad \wedge \quad \exists x \forall y\ \neg R(x, y).$$

As per our translation to propositional clauses, this is equivalent to the conjunction of the universal clauses

$$(i.) \qquad \forall x \forall y\ \neg S_2(x, y) \vee R(x, y) \text{ and}$$
$$(ii.) \qquad \forall x \forall y\ \neg S_1(x) \vee \neg R(x, y),$$

together with the Skolem system

$$\forall x \exists y\ S_2(x, y) \text{ and}$$
$$\exists x\ S_1(x).$$

Figure 3.1 shows an FO decision tree for this system of clauses. The number following each # specifies the clause that has been contradicted. For example, the bottom right # comes from the knowledge $S_2(1,2)$ and $\neg R(1,2)$—which contradicts the first universal clause. ◊

We can now modify the proof above in order to prove Case 2a of Theorem 3.2.

PROOF OF CASE 2A, THEOREM 3.2. We shall construct a boolean decision tree for the parameterized FO case in a similar manner, but with the following modification: whenever we witness a new constant and extend the set of constants by adding it, we add *another* new constant that is *independent* from all the others. That is, we actually introduce new constants to C in pairs, c' and c'', where c' is a Skolem witness for some constant in C and c'' is assumed to be independent from $C \cup \{c'\}$ (we make no assumption of the independence of c' from C). Thereafter, we may also close branches whenever we directly contradict the independence of c'' from $C \cup \{c'\}$. Now, suppose for the sake of contradiction that the decision tree constructed in this way is infinite. Again, by König's Lemma, there must be an infinite branch which constitutes an infinite model of ψ with the additional property that it has no finite dominating set. Indeed, by the construction, for every finite set of constants, we always add a new constant that is independent from the set. This gives us the desired contradiction, thus showing that the decision tree we have constructed is finite. Let the depth of this tree be h and the maximum size of C along any of its branches be m.

What remains is to estimate the branching factor of the queries in the (parameterized) propositional case. The R and S queries have branching factors 2 and n as before. The only problem is in finding a new constant that is independent from all existing constants. The parameterized boolean decision tree in the propositional case can "search" for such a constant in the following way. Denote the set of elements of the finite universe $[n]$ that have not been queried at all so far by $Z = \{z_1, z_2, \dots z_p\}$ and the set of already known constants by C. The parameterized boolean decision tree first queries all possible R relations with arguments over $C \cup \{z_1\}$ that could possibly make z_1 dominated by C. If all answers

$\exists x\ U(x)$	U-existence
$\forall x\ \neg U(x) \vee \neg R(x,x)$	U-antireflexivity
$\forall x \forall y \forall z\ \neg U(x) \vee \neg U(y) \vee \neg U(z) \vee \neg R(x,y) \vee \neg R(y,z) \vee R(x,z)$	U-transitivity
$\forall x \forall y\ \neg U(x) \vee \neg U(y) \vee R(x,y) \vee R(y,x)$	U-totality
$\forall y \exists x\ U(y) \to (U(x) \wedge R(x,y))$	U-non-minimality
$\exists x \forall y\ U(y) \vee R(x,y)$	$\neg U$-dominator

Figure 3.2: Sentence ψ of Example 3.6.

are negative, then z_1 is independent from C, so it is success—z_1 is added to C and we proceed further according to the decision tree in the FO case in the parameterized setting. Otherwise, on the first positive answer (i.e., having found out that z_1 is dominated by C), we abandon z_1 and proceed the same way with z_2 and so on. For every z_i which we query the branching factor is bounded by m^{ab} where a is the maximum arity of any R relation of ψ and b is the number of R relations of ψ. On the other hand, we do not need to test more than k elements of Z as the parameterized boolean decision tree cannot take more than k positive answers and we need to move onto a new element of Z on a positive answer only. This gives us a subtree of height k and branching factor m^{ab}, which is equivalent to a single node of branching factor m^{abk}. To conclude, let us recall that the FO tree in the parameterized case was of constant height h that depends on the formula ψ only, and thus, the parameterized boolean decision tree in the propositional case is of size at most $(\max\{m^{abk}, n\})^h$ which is not greater than $(m^{abh})^k n^h$ as claimed. □

Example 3.6. We give an example of a decision tree constructed as in Case 2a, Theorem 3.2. We consider the sentence ψ which is the conjunction depicted in Figure 3.2. The sentence ψ asserts the existence of a bipartition, in which the U-part is a non-empty strict total R-order without minimal element, and such that there is a single element with an R-edge to all the elements of the $\neg U$-part. Depending on which part this single element is in, a model of ψ will have a dominating set of size 1 or 2. As per our translation, this is equivalent to the universal clauses of Figure 3.3 together

$(i.)$	$\forall x\ \neg S_1(x) \vee U(x)$
$(ii.)$	$\forall x\ \neg U(x) \vee \neg R(x,x)$
$(iii.)$	$\forall x \forall y \forall z\ \neg U(x) \vee \neg U(y) \vee \neg U(z) \vee \neg R(x,y) \vee \neg R(y,z) \vee R(x,z)$
$(iv.)$	$\forall x \forall y\ \neg U(x) \vee \neg U(y) \vee R(x,y) \vee R(y,x)$
$(v.)$	$\forall y \forall x\ \neg S_2(x,y) \vee \neg U(y) \vee U(x)$
$(v'.)$	$\forall y \forall x\ \neg S_2(x,y) \vee \neg U(y) \vee R(x,y)$
$(vi.)$	$\forall x \forall y\ \neg S_3(x) \vee U(y) \vee R(x,y),$

Figure 3.3: Clauses for Example 3.6.

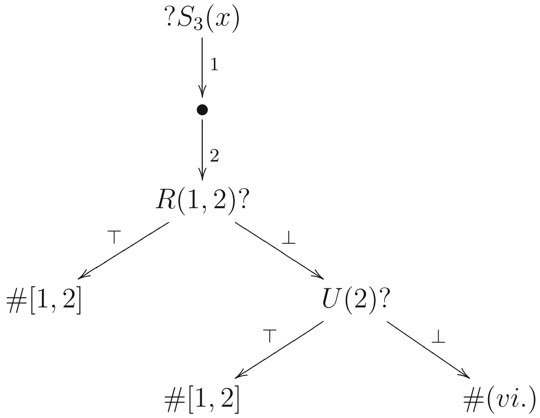

Figure 3.4: Decision tree for Example 3.6.

with the Skolem system

$$\exists x\ S_1(x)$$
$$\forall y \exists x\ S_2(x,y)$$
$$\exists x\ S_3(x).$$

Note that the Skolem relation S_1 is somewhat redundant and is included for the sake of formality (it would preserve meaning if we were to remove clause $(i.)$ and substitute $\exists x\ U(x)$ for the Skolem clause $\exists x\ S_1(x)$). Figure 3.4 shows an FO decision tree for this system in the parameterized case. (Note that we have questioned constants and relations in an intelligent, rather than natural, order. This is so that we might keep the size of the tree to a minimum; the tree would still close if we chose a natural order.) The bullet points (•) indicate where, having just witnessed a new constant, we introduce another new, independent constant. In the decision tree, we know that 2 must be independent from 1. The contradictions labeled with square brackets arise from violating the independence condition. I.e., at #[1, 2] we have just learned that 2 is dominated by 1.

The height of our tree is $h = 5$, and we never involve more than $m = 2$ constants; the maximum arity is $a = 2$, and there are $b = 2$ involved non-Skolem relations. As in the previous proof, using the bound $(m^{abh})^k n^h$, we can state that $(\mathcal{C}_{\psi,n}, k)$ has a parameterized tree-like resolution refutation of size bounded by $2^{20k} n^5$.

Owing to the rules that allow us to introduce independent constants, the character of the FO decision tree in the parameterized case is different from the ordinary FO decision tree. Notice that we have closed our tree without witnessing the Skolem relation $S_1(x)$. It would not be possible to close an ordinary FO decision tree without this since, without the U-existence clause $(i.)$, ψ has finite models. ◊

We conclude this section with a somewhat simpler example of Case 2a of Theorem 3.2, on which the previous example was based. This specimen provides a trivial instance, having, as it does, parameterized tree-like resolution refutations not just polynomial in n, but actually independent of n.

EXAMPLE 3.7. We consider the (negation of the) least number principle for total orders. Let ψ^{LNP_1} be the conjunction of the following.

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

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 $\langle(\mathcal{C}_{\psi^{\mathrm{LNP}_1},n}, k)\rangle_{n\in\mathbb{N}}$ has parameterized tree-like resolution refutations of size $2k$. ◊

3.2. Case 2b of Theorem 3.2. We now turn our attention to proving Case 2b of Theorem 3.2. Our argument will be facilitated by a game based on those described by Pudlák (2000) and Riis (2001) in which *Prover* (female) plays against *Adversary* (male). In this game, a strategy for Prover on a propositional formula $\mathcal{F}$ gives rise to a parameterized boolean decision tree on a set of clauses. Prover questions the variables of CNF($\mathcal{F}$) 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 of $\mathcal{F}$ are true ($\top$), for in either of these situations Prover is deemed the winner. Of course, assuming $\mathcal{F}$ was not satisfiable with weight $\leq k$, 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 CNF($\mathcal{C}_{\psi,n}$) 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 $\mathrm{CNF}(\mathcal{C}_{\psi,n})$ or she holds more than k variables of $\mathcal{C}_{\psi,n}$ evaluated to 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 M_ψ 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 an independent set whose members are also 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—isomorphic to some expansion M_ψ^{Sk} of M_ψ by Skolem relations—that are consistent with the information Adversary has conceded on X. Note that, while we do not consider Skolem relations in our definitions of independence or domination, we do consider them in the models of $\mathcal{M}_X$. 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 a question $R_i(\overline{c})$ true. In particular, in Case III, it will never be the case that all models in $\mathcal{M}_P$ satisfy a question $R_i(\overline{c})$. 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 lemmas.

LEMMA 3.8. *Let ψ be a sentence of FO, M be a model of ψ without a finite dominating set, and P be a finite subset of $|M|$. For any positive integer q, there exists an independent set Q of size q such that all elements of Q are independent from P.*

PROOF. Suppose for contradiction some M fails to have this property. Consider any finite P, of size p, in $|M|$. If there is a q such that all sets $Q \subseteq |M| \backslash P$ are either not independent or some element in Q is not independent from P, then there is a maximal q_0, the cardinality of a set Q_0, that is independent and whose elements are independent from P. But, $P \cup Q_0$ is now a finite dominating set of M by the maximality of q_0. □

LEMMA 3.9. *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. Our proof is broadly based on the ideas of Riis (2001). 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. If Prover asks a question $S_j(\overline{c})$, this is clearly the case. In order to see why this is the case when Prover asks some $R_i(\overline{c})$, it becomes necessary to explain why in Case II of his strategy

Adversary never loses all 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 all models M in $\mathcal{M}_P$ because Q can always be chosen to be independent, and independent from P, by Lemma 3.8. Indeed, if such an interpretation is put on Q in M, then Adversary's answer is forced to be false ($\bot$).

Suppose, in Case III, that Adversary were forced to answer true ($\top$), i.e., all models M in $\mathcal{M}_P$ satisfy the question $R_i(\overline{c})$. 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 $R_i(\overline{c})$ 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 M_ψ^{Sk}. □

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

LEMMA 3.10. *Let a be the maximum arity of any relation in ψ and suppose that there are no more than b distinct relations in ψ. 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 on variables of $\mathcal{C}_{\psi,n}$ is trivially bounded by the size of the set of all possible questions of variables of $\mathcal{C}_{\psi,n}$ on P, which is certainly bound by p^{ab}. Hence, while $p^{ab} < k$, there must be fewer than k propositional variables of $\mathcal{C}_{\psi,n}$ 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 3.2. 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 $\mathcal{G}(\mathcal{C}_{\psi,n}, k)$. Let a be the maximum arity of any relation of ψ and let b be the number of distinct relations in ψ. 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, while $p < k^{1/ab}$ 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 $T(p, q)$ 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 $T(0, 0)$ is a lower bound on the size of the game tree itself. We claim the following.

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

The second item follows from Lemma 3.10. For the first item, we consider only the free choice branching points in the game tree—that is we consider the binary tree that is a minor of the game tree in the natural way. At these points, on answering true, some constants—at most a—may be added to P. Some may have been taken from Q, but since the function T is monotonic decreasing the bound still holds. If the answer is false, then at most a constants may be added to Q and the bound holds for similar reasons.

By induction on the (complete binary tree minor of the) game tree for $\mathcal{G}(\mathcal{C}_{\psi,n}, k)$, we can prove that the given recurrence satisfies

$$T(p,q) \geq \binom{\lfloor \frac{n-p-q}{a} \rfloor}{\lfloor \frac{k^{1/ab}-p}{a} \rfloor} - 1.$$

We may now solve this to deduce that, if n, k, a and b are 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

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

The Case 2b clearly follows. □

EXAMPLE 3.11. 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 3.7 without the third 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 3.12. We return to the sentence ψ^{PHP} defined in Example 2.11. 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 CONT be the class of (ordinary) contradictions in CNF. Let e : PAR CONT $\to$ CONT 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 $\Gamma : \Sigma_1^* \to$ CONT be a proof system for CONT. It follows that $\Gamma' : \Sigma_1^* \times \mathbb{N} \to$ PAR CONT 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 of $\mathcal{F}$. Both of these fail—though the latter less spectacularly—since the function given by $(\mathcal{F},k) \mapsto \mathrm{CNF}(\mathcal{F}) \cup \mathcal{N}_k$ (respectively, $(\mathcal{F},k) \mapsto \mathrm{CNF}(\mathcal{F}) \cup \mathcal{N}_k'$) 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 $\mathcal{N}_k''$, and consider the mapping given by $(\mathcal{F},k) \mapsto \mathrm{CNF}(\mathcal{F}) \cup \mathcal{N}_k''$. The clauses $\mathcal{N}_k''$ 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 4.1. *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 2.11 that the axioms are $\mathrm{CNF}(\mathcal{C}_{\psi^{\mathrm{PHP}},n})$

$$\begin{array}{l}\neg S_2(i,j) \vee R(i,j),\ \neg S_1(j) \vee \neg R(i,j) \quad \text{and} \\ \neg R(i,j) \vee \neg R(i',j), \quad \text{for } i,i',j \in [n], i \neq i', \\ \bigvee_{j=1}^{n} S_2(i,j), \quad \text{for } i \in [n], \quad \text{and } \bigvee_{i=1}^{n} S_1(i).\end{array}$$

Let V be the set of variables in $\mathcal{C}_{\psi^{\mathrm{PHP}},n}$. We now add the auxiliary clauses $\mathcal{N}''_k :=$

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

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 $\mathrm{CNF}(\mathcal{C}_{\psi^{\mathrm{PHP}},n}) \cup \mathcal{N}''_k$, 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 of $\mathcal{F}$ of size 2^k such that no clause (other than the axioms) contains more than one negated variable (Buss & Pitassi 1998). We will convert this refutation into one for $\mathrm{CNF}(\mathcal{C}_{\psi^{\mathrm{PHP}},n}) \cup \mathcal{N}_k''$ of size at most $2^k n^2$.

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

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

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 $\mathrm{CNF}(\mathcal{C}_{\psi^{\mathrm{PHP}},n}) \cup \mathcal{N}_k''$ 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:

$$\begin{aligned}&(r_{i_1 j_1} \vee r_{i_2 j_2} \vee \ldots \vee r_{i_t j_t}) \\ &\qquad \equiv \bigvee_{l=1}^{n} q_{R(i_1,l),j_1} \vee \bigvee_{l=1}^{n} q_{R(i_2,l),j_2} \vee \ldots \vee \bigvee_{l=1}^{n} q_{R(i_t,l),j_t}\end{aligned}$$

and

$$\begin{aligned}&(\neg r_{i_1 j_1} \vee r_{i'_2 j'_2} \vee \ldots \vee r_{i'_{t'} j'_{t'}}) \\ &\qquad \equiv \bigwedge_{l=1}^{n} \neg q_{R(i_1,l),j_1} \vee \bigvee_{l=1}^{n} q_{R(i'_2,l),j'_2} \vee \ldots \vee \bigvee_{l=1}^{n} q_{R(i'_{t'},l),j'_{t'}}\end{aligned}$$

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

$$\neg q_{R(i_1,1),j_1} \vee \bigvee_{l=1}^{n} q_{R(i'_2,l),j'_2} \vee \ldots \vee \bigvee_{l=1}^{n} q_{R(i'_{t'},l),j'_{t'}}$$
$$\vdots$$
$$\neg q_{R(i_1,n),j_1} \vee \bigvee_{l=1}^{n} q_{R(i'_2,l),j'_2} \vee \ldots \vee \bigvee_{l=1}^{n} q_{R(i'_{t'},l),j'_{t'}}$$

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 $\mathrm{CNF}(\mathcal{C}_{\psi^{\mathrm{PHP}},n}) \cup \mathcal{N}''_k$, and the result follows. □

5. Final Remarks

We define parameterized tree-like resolution as a refutation system for parameterized contradictions not necessarily in CNF, yet it is customary to define tree-like resolution only for contradictions in CNF. Indeed, we give Riis's gap theorem with tree-like resolution only on CNFs. We note here that (tree-like) resolution may readily be defined over arbitrary formulas, using our canonical translation to CNF; this would clearly preserve Riis's gap theorem. However, it is actually important in the parameterized case to separate those variables that should be considered with respect to the weight parameter from those introduced purely for the translation to CNF. In particular, for our gap theorem to have a model-theoretic interpretation, the Skolem variables should not count toward the weight.

The gap theorem aside, we could have defined parameterized tree-like resolution only for CNFs and thus given a refutation system for the language PAR CNF CONT of parameterized contradictions in CNF. Being the complement of BOUNDED CNF SAT (itself a generalization of HITTING SET), PAR CNF CONT is co-W[2]-complete. We could then have had a program to gain evidence that W[2] $\neq$ FPT by proving that this flavor of parameterized tree-like resolution is not fpt-bounded. In fact, we can derive this from the "hard" case of our gap theorem, via Example 3.12. We may consider the more usual propositional encoding of the (negation of the) pigeonhole principle—without Skolem variables—to be $\neg R(x, 1), \neg R(x, y) \vee \neg R(w, y)$ and $\bigvee_{i=1}^{n} R(x, i)$, for $w, x, y \in [n], w \neq x$. The same Adversary strategy as in Section 3.2 may readily be seen to guarantee large parameterized tree-like resolutions here, in the absence of variables that do not count toward assignment weight.

It is not obvious that we can reduce our program further—e.g., to gaining evidence that W[1] $\neq$ FPT. This is because, while the language WEIGHTED 3-CNF SAT is W[1]-complete, the language BOUNDED 3-CNF SAT (and therefore PAR 3-CNF CONT) is in FPT. BOUNDED 3-CNF SAT may be solved by the following inductive algorithm. If all remaining clauses have at least one negative literal, then we may satisfy them all with the all-false assignment. Otherwise, pick a clause with three positive literals and branch on three possible evaluations to true. This process may be repeated at most k times, giving a time complexity $O(3^k \cdot n)$.

It seems odd that the method of auxiliary variables that we met in Section 4 makes the pigeonhole principle "easy" for resolution. It raises the question as to where we may look for parameterized contradictions that are "hard" when embedded in resolution. Of course, not all contradictions come from FO principles. But, in light of the previous paragraph, it is futile to search for them among random 3-CNFs. It may be observed that the parameterized contradictions that we use in this paper are somewhat unusual in that they are actually real contradictions. Perhaps the solution is to use parameterized contradictions in which the parameter is critical, i.e., those which are satisfiable but have no satisfying assignment of weight $\leq k$.

Our proposed program of parameterized proof complexity derives its legitimacy from Lemma 2.5, which essentially proves that if W[SAT] = FPT, then there is an fpt-bounded parameterized proof system for PAR CONT. We could have proved an alternative (ostensibly stronger) form of this lemma: if W[SAT] = co-W[SAT], then there is an fpt-bounded parameterized proof system for PAR CONT. The proof of this would be based on the assumed fpt reduction from PAR CONT to BOUNDED SAT: a proof for the former would be a witnessing assignment of the latter. In this form, the lemma looks more like the equivalence of Cook and Reckow, that NP = co-NP iff there is a polynomially bounded proof system for, say, CONT (not parameterized). However, we are unable to prove the converse, i.e., if there is an fpt-bounded parameterized proof system for PAR CONT, then W[SAT] = co-W[SAT].

Acknowledgements

We are grateful for robust comments from a journal referee.

A preliminary version of this paper was presented at the 48th IEEE Symposium on the Foundations of Computer Science at Providence RI, October 2007. This research was supported by the Engineering and Physical Sciences Research Council, UK: Projects EP/C526120/1 and EP/E001394/1.

References

Samuel Buss & Tonian Pitassi (1998). Resolution and the weak pigeonhole principle. In: *CSL '97, LNCS 1414*, 149–156. Springer Verlag.

Marco Cesati (2006). Compendium of Parameterized problems. http://bravo.ce.uniroma2.it/home/cesati/research/compendium/.

Jianer Chen, Iyad A. Kanj & Ge Xia (2006). Improved Parameterized Upper Bounds for Vertex Cover. In: *Proceedings of MFCS 2006, LNCS 4162*, 238–249. Springer Verlag.

Stephen Cook & Robert Reckhow (1979). The relative efficiency of propositional proof systems. *J. Symbolic Logic* **44**(1), 36–50.

Rodney G. Downey & Michael R. Fellows (1999). *Parameterized Complexity*. Monographs in Computer Science. Springer Verlag.

Jörg Flum & Martin Grohe (2006). *Parameterized Complexity Theory*, volume XIV of *Texts in Theoretical Computer Science. An EATCS Series*. Springer Verlag.

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

Jan Krajíček (1995). *Bounded arithmetic, propositional logic, and complexity theory.* Cambridge University Press, New York, NY, USA. ISBN 0-521-45205-8.

Rolf Niedermeier (2006). *Invitation to Fixed-Parameter Algorithms.* Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press.

Pavel Pudlák (2000). Proofs as Games. *American Mathematical Monthly* **107**(6), 541–550.

Soren Riis (2001). A complexity gap for tree-resolution. *Computational Complexity* **3**(10), 179–209.

Manuscript received 18 January 2008

Stefan Dantchev
Department of Computer Science,
University of Durham,
Durham DH1 3LE, UK.
s.s.dantchev@durham.ac.uk

Barnaby Martin
Department of Computer Science,
University of Durham,
Durham DH1 3LE, UK.
barnabymartin@gmail.com

Stefan Szeider
Department of Computer Science,
University of Durham,
Durham DH1 3LE, UK.
stefan.szeider@durham.ac.uk