Electronic Colloquium on Computational Complexity, Report No. 83 (2008)

A logic for PTIME and a parameterized halting problem

Yijia Chen * Jörg Flum †

June 10, 2008

Abstract

In [2] a logic L_Y has been introduced as a possible candidate for a logic capturing the PTIME properties of structures (even in the absence of an ordering of their universe). A reformulation of this problem in terms of a parameterized halting problem p-ACC for nondeterministic Turing machines has been given in [10]. We analyze the precise relationship between L_Y and p-ACC. Moreover, we show that p-ACC is not fixed-parameter tractable if "P $\neq$ NP holds for all time constructible functions." This property also implies that the natural parameterization of Gödel's proof predicate is not fixed-parameter tractable. Furthermore, the complexity of various variants of p-ACC is considered.

1. Introduction.

The existence of a logic expressing precisely the PTIME properties remains the central problem in descriptive complexity. A proof that such a logic does not exist would yield that P $\neq$ NP and hence solve the most prominent open problem in complexity theory. It is well-known that least fixed-point logic FO(LFP) captures the PTIME properties on *ordered* structures. However the property of an *arbitrary* finite structure of having a universe of even cardinality is not expressible in FO(LFP). Of course there are artificial logics expressing precisely the PTIME properties, however they do not fulfill a natural requirement to logics in this context, namely the condition:

($*$) There is an algorithm that decides whether $\mathcal{A}$ is a model of φ for all structures $\mathcal{A}$ and sentences φ of the logic and that does this for fixed φ in time polynomial in the size of $\mathcal{A}$.

In [2] the authors introduce a logic L_Y related to FO(LFP), in which precisely the PTIME properties are expressible, however it is not known whether L_Y satisfies the effectivity condition ($*$). In [10] it has been shown that the statement "L_Y satisfies ($*$)" can be equivalently formulated as a statement concerning the complexity of a halting problem for nondeterministic Turing machines. This reformulation is best expressed in the terminology of parameterized complexity. We consider the *parameterized acceptance problem* p-ACC *for nondeterministic Turing machines.*

p-ACC	
Instance:	A nondeterministic Turing machine $\mathbb{M}$ and $n \in \mathbb{N}$ in unary.
Parameter:	$\|\mathbb{M}\|$, the size of $\mathbb{M}$.
Question:	Does $\mathbb{M}$ accept the empty input tape in at most n steps?

Then

$$L_Y \text{ satisfies } (*) \quad \text{if and only if} \quad p\text{-ACC} \in \text{XP}.^{1}$$

In this paper we mainly deal with two questions:

(a) What does "p-ACC is fixed-parameter tractable" mean for the logic L_Y?

(b) What is the complexity of p-ACC?

*Shanghai Jiaotong University, China. Email: yijia.chen@cs.sjtu.edu.cn

†Albert-Ludwigs-Universität Freiburg, Germany. Email: joerg.flum@math.uni-freiburg.de

[1] Depending on the exact formulation of ($*$), we need the class XP of uniform or of nonuniform parameterized complexity; see Section 2 for the precise definitions.

ISSN 1433-8092

While we can answer question (a), we are only able to relate the statements "p-ACC $\in$ XP" and "p-ACC $\in$ FPT" with other open problems of complexity theory.

More precisely, the content of the different sections is the following. It is known that the time bound for the evaluation of a sentence φ of FO(LFP) in a structure $\mathcal{A}$ contains a factor $\|\mathcal{A}\|^{O(|\varphi|)}$; an analysis of the model-checking algorithm shows that a factor of the form $\|\mathcal{A}\|^{O(\text{width}(\varphi))}$ suffices, where width(φ), the width of φ, essentially is the maximum number of free variables in a subformula of φ. The main result of Section 3 shows that for the logic L_Y a bound of the second type is equivalent to p-ACC $\in$ FPT.

Let P[TC] $\neq$ NP[TC] mean that there is *no* time constructible and increasing function h such that NTIME($h^{O(1)}$) $\subseteq$ DTIME($h^{O(1)}$) (and hence, NTIME($h^{O(1)}$) $=$ DTIME($h^{O(1)}$)). In Section 4 we show that the inequality P[TC] $\neq$ NP[TC] implies that p-ACC $\notin$ FPT and that a stronger hypothesis where DTIME($h^{O(1)}$) is replaced by DTIME($h^{O(\log h)}$) implies that p-ACC $\notin$ XP. In Section 2.1 we relate these hypotheses to other statements of complexity theory. In particular, we shall see that P[TC] $\neq$ NP[TC] holds if there is a P-bi-immune problem in NP.

We consider the construction problem associated with p-ACC in Section 4.2 and show that it is not fpt Turing reducible to p-ACC in case p-ACC $\notin$ XP.

Variants of p-ACC are studied in Section 5. First we deal with p-ACC$_=$, the problem obtained from p-ACC by asking for an accepting run of *exactly* n steps. We improve a result of [1] and show that p-ACC$_=$ $\in$ FPT if and only if NE $=$ E (that is, NTIME($2^{O(n)}$) $=$ DTIME($2^{O(n)}$)). Furthermore, we introduce a halting problem for deterministic Turing machines, the "deterministic version" of p-ACC, and show that it is a natural example of a problem nonuniformly fixed-parameter tractable but not contained in uniform XP.

In Section 6 we show that a further important problem is not in FPT if P[TC] $\neq$ NP[TC] holds. The undecidability of first-order logic tells us that there is no computable bound on the length of the shortest proof of a valid FO-sentence (that is, first-order sentence). By mathematicians' experience various valid FO-sentences φ only have quite long proofs, say, proofs of a length $\geq 2^{|\varphi|}$. How hard is it to recognize such "hard" valid sentences? This is a sparse problem and hence not NP-hard unless NP=P. So how do we convince ourselves that it is intractable? It turns out that the polynomial time tractability of this problem implies the fixed-parameter tractability of p-FO-PROOF, that is, of the problem asking, given a first-order sentence φ and a natural number n in unary, whether φ has a proof of length $\leq n$, where the parameter is $|\varphi|$. We show that p-FO-PROOF is not fixed-parameter tractable under P[TC] $\neq$ NP[TC].

We should mention that some of the results could be formulated in more "logical terms" by using instead of the halting problem p-ACC an appropriate parameterization of the (complement of) the problem INV introduced at the beginning of Section 3.

2. Preliminaries.

In this section we review some of the basic concepts of parameterized complexity and of logics or query languages and their complexity.

2.1. Parameterized complexity. We identify problems with subsets Q of $\{0,1\}^*$. Clearly, as done mostly, we present concrete problems in a verbal, hence uncodified form. We use both P and PTIME to denote the class of problems Q such that $x \in Q$ is solvable in polynomial time. All Turing machines have $\{0,1\}$ as alphabet.

We view *parameterized problems* as pairs (Q,κ) consisting of a classical problem $Q \subseteq \{0,1\}^*$ and a *parameterization* $\kappa : \{0,1\}^* \to \mathbb{N}$, which is required to be polynomial time computable.

We assume knowledge of the basic notions of parameterized complexity and refer to [6] for notions not defined here. We denote by FPT, XP,... the usual (strongly uniform) classes, by $\text{FPT}_{\text{uni}}, \text{XP}_{\text{uni}}, \ldots$ their *uniform* versions, and by $\text{FPT}_{\text{nu}}, \text{XP}_{\text{nu}}, \ldots$ their *nonuniform* versions. For example, $(Q,\kappa) \in$ FPT if there is an algorithm solving $x \in Q$ in time $f(\kappa(x)) \cdot |x|^{O(1)}$ for a *computable* $f : \mathbb{N} \to \mathbb{N}$; moreover, $(Q,\kappa) \in \text{FPT}_{\text{uni}}$ if there is an algorithm solving $x \in Q$ in time $f(\kappa(x)) \cdot |x|^{O(1)}$ for an *arbitrary* $f : \mathbb{N} \to \mathbb{N}$; and $(Q,\kappa) \in \text{FPT}_{\text{nu}}$ if there is a constant c, an arbitrary function $f : \mathbb{N} \to \mathbb{N}$, and for every $k \in \mathbb{N}$ an algorithm solving the problem $x \in Q$ for all x with $\kappa(x) = k$ in time $f(k) \cdot |x|^c$.

We write $(Q,\kappa) \leq^{\text{fpt}} (Q',\kappa')$ and $(Q,\kappa) \leq^{\text{fpt-T}} (Q',\kappa')$ if there are an fpt reduction and an fpt Turing reduction from (Q,κ) to (Q',κ'), respectively (these concepts refer to the strongly uniform parameterized

complexity theory).

2.2. Logic. A *vocabulary* τ is a finite set of relation symbols. Each relation symbol has an *arity*. The *arity* of τ is the maximum of the arities of the symbols in τ. A *structure* $\mathcal{A}$ of vocabulary τ, or τ*-structure* (or, simply structure), consists of a nonempty set A called the *universe*, and an interpretation $R^{\mathcal{A}} \subseteq A^r$ of each r-ary relation symbol $R \in \tau$. We say that $\mathcal{A}$ is finite, if A is a finite set. For finite $\mathcal{A}$ we denote by $\|\mathcal{A}\|$ the size of $\mathcal{A}$, that is, the length of a reasonable encoding of $\mathcal{A}$ as string in $\{0,1\}^*$ (see [6] for details). If necessary, we can assume that the universe of a finite structure is $[m] := \{1, \dots, m\}$ for some natural number $m \geq 1$, as all the properties of structures we consider are invariant under isomorphisms; in particular, it suffices that from the encoding of $\mathcal{A}$ we can recover $\mathcal{A}$ up to isomorphism. Therefore, the reader will easily convince himself that we can assume that there is a computable binary function bit such that for every vocabulary τ and $m \in \mathbb{N}$ (we just collect the properties of bit we are going to use in Section 3):

(1) $\|\mathcal{A}\| = \text{bit}(\tau, m)$ for every τ-structure $\mathcal{A}$ with universe of cardinality m;

(2) $\text{bit}(\tau, m) < \text{bit}(\tau', m')$ for all τ, τ' with $\tau \subseteq \tau'$ and m, m' with $m < m'$;

(3) $\text{bit}(\tau, m) = O(\log |\tau| \cdot |\tau| \cdot m)$ for every τ containing only unary relation symbols;

(4) $\text{bit}(\tau \cup \{R\}, m) = O(\text{bit}(\tau, m) + m^2)$ for every binary relation symbol R not in τ.

We assume familiarity with *first-order logic* FO and its extension *least fixed-point logic* FO(LFP) (e.g., see [5]). We denote by FO$[\tau]$ and FO(LFP)$[\tau]$ the set of sentences of vocabulary τ of FO and of FO(LFP), respectively.

It is known that every FO(LFP)-sentence is equivalent to an FO(LFP)-sentence in normal form, where an FO(LFP)-sentence φ is in *normal form* if it has the form

$$\varphi = \exists x_1 \dots \exists x_k [\text{LFP}_{x_1 \dots x_k, X} \psi(x_1 \dots x_k, X)]\, x_1 \dots x_k, \tag{1}$$

where ψ is a first-order formula, X is an k-ary relation variable, and $x_1, \dots, x_k$ are the (first-order) variables free in ψ. The *width* width(φ) of φ as in (1) is the width of the first-order formula ψ, that is, the maximum number of free variables of a subformula of ψ. Clearly, width$(\varphi) \leq |\varphi|$. Moreover analyzing standard proofs that convert an FO(LFP)-sentence $\neg\varphi$, where φ is in normal form, into normal form show:

Lemma 1. *There is a constant $c \in \mathbb{N}$ such that in polynomial time we obtain for every* FO(LFP)*-sentence φ in normal form a normal form for the sentence $\neg\varphi$ of width $\leq c \cdot \text{width}(\varphi)$.*

The following fact is well-known and implicit in [11]:

Proposition 2. *The model-checking problem $\mathcal{A} \models_{\text{LFP}} \varphi$ for finite structures $\mathcal{A}$ and* FO(LFP)*-sentences φ in normal form can be solved in time*

$$O(|\varphi| \cdot |A|^{2 \cdot \text{width}(\varphi)} \cdot \text{width}(\varphi) + \|\mathcal{A}\|) = O(|\varphi|^2 \cdot \|\mathcal{A}\|^{2 \cdot \text{width}(\varphi)}).$$

2.3. Logics capturing PTIME. For our purposes, a *logic* consists:

- for every vocabulary τ, of a decidable set $L[\tau]$ of strings, the set of L*-sentences of vocabulary* τ;
- of a *satisfaction relation* $\models_L$; if a pair $(\mathcal{A}, \varphi)$ is in $\models_L$, then $\mathcal{A}$ is a τ-structure and $\varphi \in L[\tau]$ for some vocabulary τ.

We take over the following terminology from [10].

Definition 3. Let L be a logic. L is a logic for P, if:

(1) (L *captures* P) for all vocabularies τ and all classes C (of encodings) of finite τ-structures closed under isomorphisms we have

$$\text{C} \in \text{P} \iff \text{for some } L[\tau]\text{-sentence } \varphi\text{:} \quad \text{C} = \{\mathcal{A} \mid \mathcal{A} \models_L \varphi\};$$

(2) (*there is a computable evaluator*) there is an algorithm $\mathbb{A}$ deciding $\models_L$, that is, for every finite structure $\mathcal{A}$ and L-sentence φ the algorithm $\mathbb{A}$ decides whether $\mathcal{A} \models_L \varphi$.

We say that L *is a* P*-bounded logic for* P if (1) holds and we can choose the algorithm $\mathbb{A}$ in (2) in such a way that for every fixed L-sentence φ it runs in polynomial time (polynomial in the size of $\mathcal{A}$).

Hence, if L is a P-bounded logic for P, then for every L-sentence φ the algorithm $\mathbb{A}$ witnesses that $\{\mathcal{A} \mid \mathcal{A} \models_L \varphi\}$ lies in P. However, we do not necessarily know ahead of time the bounding polynomial.

We say that L *is an effectively* P*-bounded logic for* P if L is a P-bounded logic for P and if in addition to the algorithm $\mathbb{A}$ there is a computable function that assigns to every L-sentence φ a polynomial $q \in \mathbb{N}[X]$ such that $\mathbb{A}$ decides whether $\mathcal{A} \models_L \varphi$ in $\leq q(\|\mathcal{A}\|)$ steps.

3. The logic L_Y for polynomial time and the parameterized halting problem p-ACC.

First we recall the definition of the syntax and semantics of the logic L_Y, introduced in a slightly different form in [2]. For a vocabulary τ let $\tau_< := \tau \cup \{<\}$, where $<$ is a binary relation symbol not in τ.

We define the relation INV between sentences of FO(LFP) and natural numbers. A pair (φ, m) is in INV if $\varphi \in$ FO(LFP)$[\tau_<]$ for some vocabulary τ and if in structures with universe of cardinality $\leq m$ the satisfaction relation for φ is *inv*ariant under a change of the ordering $<$; more precisely:

$$(\varphi, m) \in \text{INV} \iff \varphi \in \text{FO(LFP)}[\tau_<] \text{ for some vocabulary } \tau,\ m \in \mathbb{N}, \text{ and for all } \tau\text{-structures } \mathcal{A}$$

$$\text{with } |A| \leq m \text{ and all orderings } <_1 \text{ and } <_2 \text{ on } A \text{ we have}$$

$$(\mathcal{A}, <_1) \models_{\text{LFP}} \varphi \iff (\mathcal{A}, <_2) \models_{\text{LFP}} \varphi.$$

We mostly write $\text{INV}(\varphi, m)$ for $(\varphi, m) \in \text{INV}$.

Then $L_Y[\tau] :=$ FO(LFP)$[\tau_<]$ and for a τ-structure $\mathcal{A}$ and an $L_Y[\tau]$-sentence φ the satisfaction relation $\models_{L_Y}$ is defined by

$$\mathcal{A} \models_{L_Y} \varphi \quad \iff \quad \text{INV}(\varphi, |A|) \text{ and } \text{ for some ordering } < \text{ on } A\text{: } (\mathcal{A}, <) \models_{\text{LFP}} \varphi.$$

Of course, in this definition we could replace "for some ordering" by "for all orderings." In the following in effectivity considerations we will denote by $<_A$ the ordering on A given by the encoding of $\mathcal{A}$. Hence

$$\mathcal{A} \models_{L_Y} \varphi \quad \iff \quad \text{INV}(\varphi, |A|) \text{ and } (\mathcal{A}, <_A) \models_{\text{LFP}} \varphi.$$

Since

$$\text{INV}(\varphi, m) \iff \text{INV}(\neg\varphi, m),$$

we get

$$\text{INV}(\varphi, |A|) \iff (\mathcal{A} \models_{L_Y} \varphi \text{ or } \mathcal{A} \models_{L_Y} \neg\varphi). \tag{2}$$

As FO(LFP) is a "logic for P on ordered structures," one easily verifies that L_Y is a logic for P. The authors of [2] ask whether L_Y is a P-bounded logic for P. It is conjectured that it is not a P-bounded logic for P. In [10] this question (or conjecture) is reformulated as an effectivity property for a halting problem for nondeterministic Turing machines. This reformulation is best expressed in the terminology of parameterized complexity. We consider the *parameterized acceptance problem* p-ACC *for nondeterministic Turing machines*.

p-ACC	
Instance:	A nondeterministic Turing machine $\mathbb{M}$ and $n \in \mathbb{N}$ in unary.
Parameter:	$\|\mathbb{M}\|$.
Question:	Does $\mathbb{M}$ accept the empty input tape in at most n steps?

The problem p-ACC lies in FPT_{nu}. In fact, fix $k \in \mathbb{N}$; then there are only finitely many nondeterministic Turing machines $\mathbb{M}$ with $\|\mathbb{M}\| = k$, say, $\mathbb{M}_1, \ldots, \mathbb{M}_s$. For each $i \in [s]$ let ℓ_i be the smallest natural number ℓ such that there exists an accepting run of $\mathbb{M}_i$ on empty input tape of length ℓ. We set $\ell_i = \infty$ if

$\mathbb{M}_i$ does not accept the empty input tape. Hence the algorithm $\mathbb{A}_k$ that on any instance $(\mathbb{M}, n)$ of p-ACC with $\|\mathbb{M}\| = k$ determines the i with $\mathbb{M} = \mathbb{M}_i$, and then accepts if and only if $\ell_i \leq n$ has running time $O(\|\mathbb{M}\| + n)$; thus it witnesses that p-ACC $\in$ FPT$_{\text{nu}}$.

We call a parameterized problem (Q, κ) *slicewise monotone* if its instances have the form (x, n), where $x \in \{0,1\}^*$ and $n \in \mathbb{N}$ in unary, if $\kappa(x, n) = |x|$, and finally if for all $x \in \{0,1\}^*$ and $n, n' \in \mathbb{N}$ we have

$$(x, n) \in Q \text{ and } n < n' \text{ imply } (x, n') \in Q.$$

In particular, p-ACC is slicewise monotone and the preceding argument shows:

Lemma 4. *If (Q, κ) is slicewise monotone, then $(Q, \kappa) \in$ FPT$_{\text{nu}}$.*

Is p-ACC $\in$ FPT$_{\text{uni}}$ or, at least, p-ACC $\in$ XP$_{\text{uni}}$? It turns out that the conjecture "L_Y is not a P-bounded logic for P" mentioned above is equivalent to p-ACC $\notin$ XP$_{\text{uni}}$:

Proposition 5 ([10]). *(1) L_Y is an effectively* P*-bounded logic for* P *if and only if p-ACC $\in$ XP.*

(2) L_Y is a P*-bounded logic for* P *if and only if p-ACC $\in$ XP$_{\text{uni}}$.*

However it is not even clear whether p-ACC $\notin$ FPT. Do the statements p-ACC $\in$ FPT and p-ACC $\in$ FPT$_{\text{uni}}$ also correspond to natural properties of the logic L_Y? We address this problem in this section.

As every FO(LFP)-sentence is equivalent to an FO(LFP)-sentence in normal form, the same holds for L_Y. Recall that we already know that L_Y is a logic for P; in particular there is an algorithm $\mathbb{A}$ that decides whether $\mathcal{A} \models_{L_Y} \varphi$. We say that L_Y is an *(effectively) width* P*-bounded logic for* P if this algorithm $\mathbb{A}$ can be chosen in such a way that there is a (computable) function h and a constant c such that $\mathcal{A} \models_{L_Y} \varphi$ can be solved in time

$$O\big(h(|\varphi|) \cdot \|\mathcal{A}\|^{c \cdot \text{width}(\varphi)}\big)$$

for structures $\mathcal{A}$ and L_Y-sentences φ in normal form. By Proposition 2, the logic FO(LFP) "is an effectively width P-bounded logic for P on ordered structures." The main result of this section is:

Theorem 6. *(1) L_Y is an effectively width* P*-bounded logic for* P *if and only if p-ACC $\in$ FPT.*

(2) L_Y is a width P*-bounded logic for* P *if and only if p-ACC $\in$ FPT$_{\text{uni}}$.*

The following observations will lead to a proof of the direction "from right to left" in the statements of the theorem (moreover, they implicitly contain a proof of the corresponding directions of Proposition 5).

For an L_Y-sentence φ in normal form and a suitable time constructible function $t : \mathbb{N} \to \mathbb{N}$ we will need a nondeterministic Turing machine $\mathbb{M}_\varphi(t)$ that, started with empty tape, operates as follows: In a first phase it reads φ and extracts its vocabulary τ_φ, the set of relation symbols that do occur in φ; in a second phase it writes a word of the form 1^m for some $m \geq 1$ on some tape. The third phase (the main phase) consists of at most $t(m) + 1$ steps (this can be ensured as t is time constructible). If $\mathbb{M}_\varphi(t)$ does not stop during the first $t(m)$ steps of the main phase, then it stops in the next step and rejects. During this $t(m)$ steps, $\mathbb{M}_\varphi(t)$ guesses (the encoding of) a τ_φ-structure $\mathcal{A}$ with universe $[m]$ and two orderings $<_1$ and $<_2$ on $[m]$ and checks whether $((\mathcal{A}, <_1) \models_{\text{LFP}} \varphi \iff (\mathcal{A}, <_2) \models_{\text{LFP}} \varphi)$. If this is not the case, then $\mathbb{M}_\varphi(t)$ accepts; otherwise it rejects.

The first phase needs $O(|\varphi|)$, say,

$$d_1 \cdot |\varphi|$$

steps and the second one m steps. To guess a τ_φ-structure $\mathcal{A}$ with universe $[m]$ and two orderings $<_1$ and $<_2$ requires

$$\text{bit}(\tau_\varphi, m) + 2m^2$$

bits (see Subsection 2.2) and hence as many steps.

Finally, according to Proposition 2, the equivalence $((\mathcal{A}, <_1) \models_{\text{LFP}} \varphi \iff (\mathcal{A}, <_2) \models_{\text{LFP}} \varphi)$ can be checked in time $O(|\varphi|^2 \cdot \text{bit}((\tau_\varphi)_<, m)^{2 \cdot \text{width}(\varphi)})$, that is, in

$$d_2 \cdot |\varphi|^2 \cdot \text{bit}((\tau_\varphi)_<, m)^{2 \cdot \text{width}(\varphi)}$$

steps for some d_2.

We define

$$\begin{array}{lcl} t_\varphi(m) & := & \text{bit}(\tau_\varphi, m) + 2m^2 + d_2 \cdot |\varphi|^2 \cdot \text{bit}((\tau_\varphi)_<, m)^{2\cdot \text{width}(\varphi)}; \\ t'_\varphi(m) & := & d_1 \cdot |\varphi| + m + t_\varphi(m). \end{array} \tag{3}$$

Therefore, as t_φ is increasing and as we can define $\mathbb{M}_\varphi(t_\varphi)$ in such a way that, once m has been chosen, the two parts of the main phase need exactly $\text{bit}(\tau_\varphi, m) + 2m^2$ and $d_2 \cdot |\varphi|^2 \cdot \text{bit}((\tau_\varphi)_<, m)^{2\cdot \text{width}(\varphi)}$ steps, respectively, we have

$$\begin{array}{lcl} \text{INV}(\varphi, m) & \iff & \mathbb{M}_\varphi(t_\varphi) \text{ does not accept the empty string in} \leq t'_\varphi(m) \text{ steps} \\ & \iff & (\mathbb{M}_\varphi(t_\varphi), t'_\varphi(m)) \notin p\text{-ACC}. \end{array}$$

Now we can show one direction of Theorem 6, namely:

Lemma 7. *(1) If* p-ACC $\in$ FPT*, then* L_Y *is an effectively width* P*-bounded logic for* P.

(2) If p-ACC $\in \text{FPT}_{\text{uni}}$*, then* L_Y *is a width* P*-bounded logic for* P.

Proof: Assume $(\mathbb{M}, n) \in p$-ACC can be solved in time

$$f(\|\mathbb{M}\|) \cdot n^e \tag{4}$$

for some constant $e \in \mathbb{N}$ and some (computable) function $f : \mathbb{N} \to \mathbb{N}$. For every structure $\mathcal{A}$ and every L_Y-sentence φ, we have

$$\begin{array}{lcl} \mathcal{A} \models_{L_Y} \varphi & \iff & \text{INV}(\varphi, |A|) \text{ and } (\mathcal{A}, <_A) \models_{\text{LFP}} \varphi \\ & \iff & (\mathbb{M}_\varphi(t_\varphi), t'_\varphi(|A|)) \notin p\text{-ACC and } (\mathcal{A}, <_A) \models_{\text{LFP}} \varphi. \end{array}$$

By the definition (3) of the function t_φ and the properties of the bit-function mentioned in Section 2.2, we see that there is a computable function g such that $t'_\varphi(|A|) \leq g(|\varphi|) \cdot \|\mathcal{A}\|^{O(\text{width}(\varphi))}$. As $\mathbb{M}_\varphi(t_\varphi)$ depends only on φ, we see, by (4) and Proposition 2, that there is an algorithm and a (computable) function h such that for every structure $\mathcal{A}$ and every L_Y-sentence φ the problem $\mathcal{A} \models_{L_Y} \varphi$ can be solved in time $h(|\varphi|) \cdot \|\mathcal{A}\|^{c\cdot \text{width}(\varphi)}$. □

We turn to the other directions in Theorem 6. Let $\mathbb{M}$ be a nondeterministic Turing machine and let $m_0(\mathbb{M})$ be the maximum of the number of states and the number of tapes. We can assume that $[k]$ is the set of states of $\mathbb{M}$ (for some $k \leq m_0$) and that 1 is its initial state. We let $P_0, P_1, \ldots, P_k$ be unary relation symbols. We shall see that for the vocabulary $\tau := \{P_0, \ldots, P_k\}$ there is an FO(LFP)$[\tau_<]$-sentence $\varphi_\mathbb{M}$ in normal form with the following properties:

For every τ-structure $\mathcal{A}$:

(1) If $|A| < m_0$, then for all orderings $<^A$ on A we have $(\mathcal{A}, <^A) \models_{\text{LFP}} \varphi_\mathbb{M}$.

(2) If $|A| \geq m_0$ and $P_0^\mathcal{A}, \ldots, P_k^\mathcal{A}$ are not pairwise disjoint subsets of A, then for all orderings $<^A$ on A we have $(\mathcal{A}, <^A) \models_{\text{LFP}} \varphi_\mathbb{M}$.

(3) Let $|A| \geq m_0$ and assume that $P_0^\mathcal{A}, \ldots, P_k^\mathcal{A}$ are pairwise disjoint subsets of A and $<^A$ an ordering on A. Let $a_1, \ldots, a_{|A|}$ be the enumeration of the elements of A according to the ordering $<^A$ and choose i_s such that $a_s \in P_{i_s}^\mathcal{A}$ for $s \in [|A|]$.

   (a) If there is a $j \in [|A| - 1]$ such that the sequence $1, i_1, \ldots, i_j$ is the sequence of states of a complete run of $\mathbb{M}$, started with empty input tape (in particular, $i_s \neq 0$ for all $s \in [j]$), then $(\mathcal{A}, <^A) \models_{\text{LFP}} \varphi_\mathbb{M}$ if and only if this run of $\mathbb{M}$ is a rejecting one.

   (b) If for all $j \in [|A| - 1]$ the sequence $1, i_1, \ldots, i_j$ is not the sequence of states of a complete run of $\mathbb{M}$, started with empty input tape, then $(\mathcal{A}, <^A) \models_{\text{LFP}} \varphi_\mathbb{M}$.

We show that for every $m \geq m_0(\mathbb{M})$

$$(\mathbb{M}, m) \in p\text{-ACC} \iff \text{not } \text{INV}(\varphi_{\mathbb{M}}, m). \tag{5}$$

First assume that $(\mathbb{M}, m) \in p$-ACC. Then for some $j \in [m-1]$ and $i_1, \ldots, i_j \in [k]$, the sequence $1, i_1, \ldots, i_j$ is the sequence of states of an accepting run of $\mathbb{M}$. Using (3)(a), it is easy to construct a structure $\mathcal{A}$ on $[m]$ and an ordering $<^A$ such that $(\mathcal{A}, <^A) \not\models_{\text{LFP}} \varphi_{\mathbb{M}}$ and $P_0^{\mathcal{A}} = \{m\}$. We choose an ordering $<'$ on $[m]$ such that m is the first element of $<'$ and hence, $i_1 = 0$ under $<'$. By (3)(b) we see that $(\mathcal{A}, <') \models_{\text{LFP}} \varphi_{\mathbb{M}}$. Hence, not $\text{INV}(\varphi, m)$.

Contrary, if $(\mathbb{M}, m) \notin p$-ACC, it is easy to see, using (1)–(3), that $\mathcal{A} \models_{L_Y} \varphi_{\mathbb{M}}$ for every structure $\mathcal{A}$ with $|A| \leq m$; hence, $\text{INV}(\varphi, m)$.

The sentence $\varphi_{\mathbb{M}}$ is obtained by standard techniques. We sketch the construction of $\varphi_{\mathbb{M}}$. Recall that $\{0, 1\}$ is the alphabet of $\mathbb{M}$. The sentence $\varphi_{\mathbb{M}}$ will make use of the binary relation variable *State* and the ternary relation variables *Head*, *Letter*$_0$, *Letter*$_1$ with the intended meaning

State $t\, s$	iff	at time t the state is s
Head $t\, i\, j$	iff	at time t the head on tape i scans the jth cell
Letter$_0$ $t\, i\, j$	iff	at time t there is a 0 on the jth cell of tape i
Letter$_1$ $t\, i\, j$	iff	at time t there is a 1 on the jth cell of tape i.

To be able to apply the least fixed-point operator to a formula positive in the corresponding variable(s), we need also relation variables *CState*, *CHead*, *CLetter*$_0$ and *CLetter*$_1$ for the *complements* of the relations just introduced. Then using simultaneous least fixed-points we can express the intended meaning of $\varphi_{\mathbb{M}}$ as

$$\exists x[\text{S-LFP}_{t\ s\ \mathit{State},\ t\ i\ j\ \mathit{Head},\ t\ i\ j\ \mathit{Letter}_0,\ t\ i\ j\ \mathit{Letter}_1\ t\ s\ \mathit{CState},\ t\ i\ j\ \mathit{CHead},\ t\ i\ j\ \mathit{CLetter}_0,\ t\ i\ j\ \mathit{CLetter}_1} \psi_{\mathbb{M}}]x$$

(even though $\mathbb{M}$ is nondeterministic, in the formula $\psi_{\mathbb{M}}$ we always get the state to be chosen in the next step using the relations $P_1, \ldots, P_k$). By standard means this sentence can be converted in an FO(LFP)-sentence $\varphi_{\mathbb{M}}$ in normal form. Of course, the sentence $\varphi_{\mathbb{M}}$ depends on the machine $\mathbb{M}$, however, and this is important, it can be defined in such a way that its width is independent of $\mathbb{M}$; thereby it is crucial that we can address the ith element in the ordering $<$ by a formula of width 3.

Assume now that L_Y is an (effectively) width P-bounded logic for P and choose a constant c and a (computable) function $h : \mathbb{N} \to \mathbb{N}$ such that the model-checking problem $\mathcal{A} \models_{L_Y} \varphi$ for structures $\mathcal{A}$ and L_Y-sentences φ in normal form can be solved in time

$$h(|\varphi|) \cdot \|\mathcal{A}\|^{c \cdot \text{width}(\varphi)}. \tag{6}$$

We show that p-ACC $\in \text{FPT}_{\text{uni}}$ (p-ACC $\in$ FPT). Let $(\mathbb{M}, m)$ be an arbitrary instance of p-ACC. If $m < m_0(\mathbb{M}_0)$ ($\leq \|\mathbb{M}\|$), we check whether $(\mathbb{M}, m) \in p$-ACC by brute force. Otherwise, we construct from $\mathbb{M}$ the sentences $\varphi_{\mathbb{M}}$ and $\neg\varphi_{\mathbb{M}}$ (more precisely, the FO(LFP)-sentence in normal form equivalent to $\neg\varphi_{\mathbb{M}}$). Moreover we choose the τ-structure $\mathcal{A}$ with $A = [m]$ and empty relations). Then, by property (3) of the bit-function (see Section 2.2), we have

$$\|\mathcal{A}\| = O(\log |\tau| \cdot |\tau| \cdot |A|) = O(|\varphi_{\mathbb{M}}|^2 \cdot m). \tag{7}$$

As by (2)

$$\text{INV}(\varphi_{\mathbb{M}}, m) \iff (\mathcal{A} \models_{L_Y} \varphi_{\mathbb{M}} \text{ or } \mathcal{A} \models_{L_Y} \neg\varphi_{\mathbb{M}}),$$

we obtain by (5)

$$(\mathbb{M}, m) \notin p\text{-ACC} \iff (\mathcal{A} \models_{L_Y} \varphi_{\mathbb{M}} \text{ or } \mathcal{A} \models_{L_Y} \neg\varphi_{\mathbb{M}}).$$

Therefore, by (7), (6), and Lemma 1, we see that there is a (computable) function f and a constant $e \in \mathbb{M}$ (recall that there is a constant bounding the width of $\varphi_{\mathbb{M}}$ for all nondeterministic Turing machines) such that $(\mathbb{M}, m) \in p$-ACC can be solved in time $f(\|\mathbb{M}\|) \cdot m^e$. This finishes the proof of Theorem 6. □

By refining the previous proof we obtain:

Corollary 8. *For sufficiently large $w \in \mathbb{N}$ the problem*

Instance:	A structure $\mathcal{A}$ and an $L_Y[1]$-sentence φ of width $\leq w$.
Problem:	Is $\mathcal{A} \models_{L_Y} \varphi$?

is co-NP*-complete.*

Proof: Clearly the problem is in co-NP. Now let Q be any problem in co-NP. We give a polynomial reduction of Q to the problem in the statement. Let $\mathbb{M}$ be a polynomial time nondeterministic Turing machine such that for all $x \in \{0,1\}^*$ we have

$$x \in Q \iff \text{no run of } \mathbb{M} \text{ accepts } x.$$

Choose $c, d \in \mathbb{N}$ such that the running time of $\mathbb{M}$ on input x is bounded by $c \cdot |x|^d$. We fix $x \in \{0,1\}^*$. Again we assume that $[k]$ is the set of states of $\mathbb{M}$ and that 1 is its starting state and set $\tau := \{P_0, \ldots, P_k\}$ with unary relation variables $P_0, \ldots, P_k$. Along the lines of the previous proof one obtains an FO(LFP)$[\tau_<]$-sentence $\varphi_{\mathbb{M},x}$ in normal form with the properties (1)–(3) (on page 6), where now in (3)(a),(b) we consider complete runs *started with the word x in the input tape*. One can obtain $\varphi_{\mathbb{M},x}$ in such a way that its width does not depend on x; again, we use the fact that for $i \in [|x|]$ we can address the ith element in the ordering $<$ by a formula of width 3. Let $\mathcal{A}_x$ be any τ-structure with $|A| \geq \max\{m_0(\mathbb{M}),\ c \cdot |x|^d\}$. Then

$$x \in Q \iff \mathcal{A}_x \models_{L_Y} \varphi_{\mathbb{M},x},$$

hence $x \mapsto (\mathcal{A}_x, \varphi_{\mathbb{M},x})$ is the desired reduction. □

4. The parameterized complexity of p-ACC.

A simple padding argument shows that P = NP implies that

$$\text{NTIME}(h^{O(1)}) \subseteq \text{DTIME}(h^{O(1)}).$$

for every time constructible and increasing function $h : \mathbb{N} \to \mathbb{N}$. Let

$$\text{P}[\text{TC}] \neq \text{NP}[\text{TC}]$$

mean that there is *no* time constructible and increasing function h such that

$$\text{NTIME}(h^{O(1)}) \subseteq \text{DTIME}(h^{O(1)}).$$

Taking as h the identity function we see that P$[\text{TC}] \neq$ NP$[\text{TC}]$ implies P $\neq$ NP. It is not known whether the converse implication holds. In Section 4.1 we are going to relate P$[\text{TC}] \neq$ NP$[\text{TC}]$ to further statements of complexity theory. The main result of this section is:

Theorem 9. *If* P$[\text{TC}] \neq$ NP$[\text{TC}]$*, then* p-ACC $\notin$ FPT.

For the proof of this result we need the following simple lemma:

Lemma 10. *For every computable function $f : \mathbb{N} \to \mathbb{N}$ there is a time constructible and increasing function $h : \mathbb{N} \to \mathbb{N}$ such that for all $x, y \in \mathbb{N}$*

$$f((x+y)^2) \leq h(x) + h(y).$$

Proof: We define $h_0 : \mathbb{N} \to \mathbb{N}$ by

$$h_0(n) := \max\big\{f(i^2) \mid 0 \leq i \leq 2n\big\}$$

and let $h : \mathbb{N} \to \mathbb{N}$ be a time constructible and increasing function with $h_0(k) \leq h(k)$ for all $k \in \mathbb{N}$. □

Proof of Theorem 9: By contradiction assume that p-ACC $\in$ FPT. Then there is an algorithm $\mathbb{A}$ that for every nondeterministic Turing machine $\mathbb{M}$ and every natural number n decides whether $\mathbb{M}$ accepts the empty input tape in time

$$f(\|\mathbb{M}\|) \cdot n^{O(1)}$$

for a computable and increasing function $f : \mathbb{N} \to \mathbb{N}$. We choose h according to Lemma 10 and show that NTIME$(h^{O(1)}) \subseteq$ DTIME$(h^{O(1)})$.

For this purpose let $Q \subseteq \{0,1\}^*$ be in NTIME$(h^{O(1)})$. We choose a nondeterministic Turing machine $\mathbb{M}_Q$ and constants $c, d \in \mathbb{N}$ such that the machine $\mathbb{M}_Q$ decides whether $x \in Q$ in time $c \cdot h(|x|)^d$ and every run of $\mathbb{M}_Q$ on input x is $c \cdot h(|x|)^d$ time-bounded (recall that h is time constructible).

We fix $x \in \{0,1\}^*$ and let $\mathbb{M}_{Q,x}$ be the nondeterministic Turing machine that, started with empty input tape, first writes x on some tape and then simulates $\mathbb{M}_Q$ started with x.

Clearly

$$\begin{aligned} x \in Q \quad &\iff \quad \mathbb{M}_{Q,x} \text{ accepts the empty string in at most } |x| + c \cdot h(|x|)^d \text{ steps} \\ &\iff \quad \mathbb{M}_{Q,x} \text{ accepts the empty string in at most } 2c \cdot h(|x|)^d \text{ steps} \\ &\iff \quad \mathbb{A} \text{ accepts } (\mathbb{M}_{Q,x}, 2c \cdot h(|x|)^d). \end{aligned}$$

Note that the size of $\mathbb{M}_{Q,x}$ is $O(\|\mathbb{M}_Q\|) + |x| \cdot \log |x|$. The running time of $\mathbb{A}$ on input $(\mathbb{M}_{Q,x}, 2c \cdot h(|x|)^d)$ is bounded by

$$f(O(\|\mathbb{M}_Q\|) + |x| \cdot \log |x|) \cdot (2c \cdot h(x))^{O(1)} \leq \Big(h(O(\|\mathbb{M}_Q\|)) + h(|x|)\Big) \cdot h(x)^{O(1)}.$$

As $\|\mathbb{M}_Q\|$ is a constant, this shows $Q \in$ DTIME$(h(x)^{O(1)})$. □

We can refine the previous argument to get p-ACC $\notin$ XP; however we need a complexity-theoretic assumption (apparently) stronger than P[TC] $\neq$ NP[TC].

Theorem 11. *Assume that*

$$\text{NTIME}(h^{O(1)}) \not\subseteq \text{DTIME}(h^{O(\log h)})$$

*for every time constructible and increasing function h. Then p-*ACC $\notin$ XP.

Proof: Assume that p-ACC $\in$ XP. Then there is an algorithm $\mathbb{A}$ that for every nondeterministic Turing machine $\mathbb{M}$ and every natural number n decides whether $\mathbb{M}$ accepts the empty input tape in time

$$O(n^{f(\|\mathbb{M}\|)})$$

for a computable and increasing function $f : \mathbb{N} \to \mathbb{N}$. We choose h according to Lemma 10. We define $g : \mathbb{N} \to \mathbb{N}$ by $g(n) := 2^{h(n)}$; clearly g is time constructible and increasing, too. We show that NTIME$(g^{O(1)}) \subseteq$ DTIME$(g^{O(\log g)})$.

Let $Q \subseteq \{0,1\}^*$ be an arbitrary problem in NTIME$(g^{O(1)})$. Let $\mathbb{M}_Q$ and $\mathbb{M}_{Q,x}$ for $x \in \{0,1\}^*$ be defined as in the previous proof, however, now g takes over the role of h. As there we get

$$x \in Q \quad \iff \quad \mathbb{A} \text{ accepts } (\mathbb{M}_{Q,x}, 2c \cdot g(|x|)^d).$$

The running time of $\mathbb{A}$ on input $(\mathbb{M}_{Q,x}, 2c \cdot g(|x|)^d)$ is bounded by

$$O\big(g(|x|)^{d \cdot f(O(\|\mathbb{M}_Q\|)+|x|)}\big) \leq O\Big(g(|x|)^{d \cdot \left(h(O(\|\mathbb{M}_Q\|))+h(|x|)\right)}\Big) \leq O\Big(g(|x|)^{d \cdot \left(h(O(\|\mathbb{M}_Q\|))+\log g(|x|)\right)}\Big).$$

As $\|\mathbb{M}_Q\|$ is a constant, this shows $Q \in$ DTIME$(g^{O(\log g)})$.

4.1. Relating P[TC] $\neq$ NP[TC] to other statements. For the purposes of this section the following lemma will be useful.

Lemma 12. *Assume that for some time constructible and increasing $h : \mathbb{N} \to \mathbb{N}$ we have*

$$\text{NTIME}(h^{O(1)}) \subseteq \text{DTIME}(h^{O(1)}).$$

Then there is an infinite set $I \subseteq \{0,1\}^*$ such that for every problem $Q \subseteq \{0,1\}^*$ in NP *the problem $Q \cap I$ is in* P.

Proof: Let $h : \mathbb{N} \to \mathbb{N}$ be time constructible and increasing such that

$$\text{NTIME}(h^{O(1)}) \subseteq \text{DTIME}(h^{O(1)}).$$

We show that

$$I := \big\{ \underbrace{11\ldots11}_{h(m) \text{ times}} \;\big|\; m \in \mathbb{N} \big\}$$

has the desired property. In fact, let $Q \subseteq \{0,1\}^*$ be in NP. We want to show that

$$Q \cap I = \big\{ \underbrace{11\ldots11}_{h(m) \text{ times}} \;\big|\; m \in \mathbb{N} \text{ and } \underbrace{11\ldots11}_{h(m) \text{ times}} \in Q \big\}$$

is in P. However we first we consider the problem

$$Q_0 := \big\{ \underbrace{11\ldots11}_{m \text{ times}} \;\big|\; m \in \mathbb{N} \text{ and } \underbrace{11\ldots11}_{h(m) \text{ times}} \in Q \big\}.$$

Claim 1. $Q_0 \in \text{DTIME}(h^{O(1)})$.

Proof of the claim. As Q is in NP and h is time constructible, one easily verifies that $Q_0 \in \text{NTIME}(h^{O(1)})$. Hence, we get our claim by the assumption $\text{NTIME}(h^{O(1)}) \subseteq \text{DTIME}(h^{O(1)})$. $\dashv$

Clearly,

$$Q \cap I = \big\{ \underbrace{11\ldots11}_{h(m) \text{ times}} \;\big|\; m \in \mathbb{N} \text{ and } \underbrace{11\ldots11}_{m \text{ times}} \in Q_0 \big\}. \tag{8}$$

As h is time constructible, one can check for every $s \in \{0,1\}^*$ whether $s = \underbrace{11\ldots11}_{h(m) \text{ times}}$ for some $m \in \mathbb{N}$ in time polynomial in $|s|$. Therefore and by Claim 1 we have $Q \cap I \in$ P. $\square$

Let C be a classical complexity class, that is, C is a class of problems. Recall that a problem $Q \subseteq \{0,1\}^*$ is C-*bi-immune* if both Q and the complement of Q do not have an *infinite subset* that belongs to C. It has been conjectured that

$$\text{NP contains a P-bi-immune problem.} \tag{9}$$

Moreover, Mayordomo [9] has proven that (9) is implied by a further conjecture, namely by:

$$\text{NP does not have measure 0 in E.} \tag{10}$$

This statement (10) is often used as a hypothesis in the theory of resource bounded measures [7]. Of course, E is the class of problems solvable in time $2^{O(n)}$. For the corresponding notion of measure we refer to [9]. The conditions (9) and (10) imply $\text{P}[\text{TC}] \neq \text{NP}[\text{TC}]$:

Proposition 13. *Consider the following statements:*

(a) NP *does not have measure 0 in* E.

(b) NP *contains a* P*-bi-immune problem.*

(c) There is no infinite set $I \subseteq \{0,1\}^*$ such that for every $Q \subseteq \{0,1\}^*$ in NP *the problem $Q \cap I$ is in* P.

(d) $\text{P}[\text{TC}] \neq \text{NP}[\text{TC}]$.

Then (a) implies (b), (b) implies (c), and (c) implies (d).

Proof: In Lemma 12 we have seen that "not (d) implies not (c)."

(b) $\Rightarrow$ (c): By contradiction assume that there is a set I with the properties mentioned in (c). As $\{0,1\}^* \cap I = I$, the set I is in P. Let $Q \subseteq \{0,1\}^*$ be in NP. Then $Q \cap I$ is in P and hence $(\{0,1\}^* \setminus Q) \cap I = I \setminus (Q \cap I) \in$ P. As at least one of the sets $Q \cap I$ or $(\{0,1\}^* \setminus Q) \cap I$ is infinite, we see that Q is not P-bi-immune. As Q was arbitrary this contradicts (b).

We already mentioned that "(a) $\Rightarrow$ (b)" was shown in Mayordomo [9]. □

Hence, from Theorem 9, we get:

Corollary 14. *If* NP *contains a* P*-bi-immune problem, then* p-ACC $\notin$ FPT.

Remark 15. Note that the assumption (b) in Proposition 13 seems to be much stronger than the assumption (c). In fact, as shown by the proof of the previous proposition, "not (c)" means

> there is an infinite $I \in$ P such that for all $Q \in$ NP at least one of the sets $Q \cap I$ and $\big(\{0,1\}^* \setminus Q\big) \cap I$ is an infinite set in P,

while NP contains no P-bi-immune problem can be reformulated as

> for all $Q \in$ NP there is an infinite $I \in$ P such that at least one of the sets $Q \cap I$ or $\big(\{0,1\}^* \setminus Q\big) \cap I$ is an infinite set in P.

Let $\mathrm{E}_2 = \mathrm{DTIME}(2^{n^{O(1)}})$. Then one can show along the lines of the proof of Proposition 13 the following chain of implications, the last one being the assumption used in Theorem 11:

Proposition 16. *Consider the following statements:*

(a) NP *does not have measure 0 in* E_2.

(b) NP *contains an* E*-bi-immune problem.*

(c) There is no infinite set $I \subseteq \{0,1\}^*$ *such that for every* $L \subseteq \{0,1\}^*$ *in* NP *the problem* $L \cap I$ *is in* $\mathrm{DTIME}(n^{O(\log n)})$.

(d) For every time constructible and increasing function h

$$\mathrm{NTIME}(h^{O(1)}) \not\subseteq \mathrm{DTIME}(h^{O(\log h)}).$$

Then (a) implies (b), (b) implies (c), and (c) implies (d).

The statements (a) and (b) have been considered in [7, 9].

4.2. The construction problem associated with p-ACC. We consider the construction problem associated with p-ACC:

p-CONSTR-ACC	
Instance:	A nondeterministic Turing machine $\mathbb{M}$ and $n \in \mathbb{N}$ in unary.
Parameter:	$\|\mathbb{M}\|$.
Problem:	Construct an accepting run of at most n steps of $\mathbb{M}$ started with empty input tape if there is one (otherwise report that there is no such run).

Similarly as we convinced ourselves that p-ACC $\in \mathrm{FPT}_{\mathrm{uni}}$, one gets that p-CONSTR-ACC is nonuniformly fixed-parameter tractable (it should be clear what this means).

Definition 17. An *fpt*$_{\mathrm{uni}}$ *Turing reduction* (*fpt Turing reduction*) from a parameterized construction problem (Q, κ) to a parameterized (decision) problem (Q', κ') is a deterministic algorithm $\mathbb{R}$ with an oracle to (Q', κ') solving the construction problem (Q, κ) and with the property that there are a (computable) function $f : \mathbb{N} \to \mathbb{N}$, a computable $g : \mathbb{N} \to \mathbb{N}$ and a $c \in \mathbb{N}$ such that for every instance x of Q

– the run of $\mathbb{R}$ with input x has length at most $f(\kappa(x)) \cdot |x|^c$;
– for every oracle query "$x' \in Q'$?" of the run of $\mathbb{A}$ with input x we have $\kappa(x') \leq g(\kappa(x))$.

Often the construction problem has the same complexity as the corresponding decision problem, that is, the construction problem is fpt Turing reducible to the decision problem; for p-CONSTR-ACC we can show:

Theorem 18. *(1) There is an fpt$_{\text{uni}}$ Turing reduction from p-CONSTR-ACC to p-ACC.*
(2) If p-ACC $\notin$ XP, then there is no fpt Turing reduction from p-CONSTR-ACC to p-ACC.

Proof: (1) On an instance $(\mathbb{M}, n)$ the desired reduction $\mathbb{R}$ first asks the oracle query "$(\mathbb{M}, n) \in p$-ACC?". If the answer is no, then $\mathbb{R}$ rejects. Otherwise $\mathbb{R}$, by brute force, constructs an accepting run of at most n steps of $\mathbb{M}$. We analyze the running time of $\mathbb{R}$. For $m \in \mathbb{N}$ let $\mathbb{M}_1, \ldots, \mathbb{M}_\ell$ be the finitely many nondeterministic Turing machines with $\|\mathbb{M}_i\| \leq m$ and with an accepting run started with empty input tape. Let ρ_i be such a run of $\mathbb{M}_i$ of minimum length. We set

$$f(m) = \max\{|\rho_1|, \ldots, |\rho_\ell|\}.$$

Now it is not hard to see that the running time of $\mathbb{R}$ on the instance $(\mathbb{M}, n)$ can be bounded by $\|\mathbb{M}\|^{O(f(\|\mathbb{M}\|))} \cdot n$.

(2) y contradiction, assume there is an fpt Turing reduction $\mathbb{R}$ from p-CONSTR-ACC to p-ACC. We show how $\mathbb{R}$ can be turned into an algorithm witnessing p-ACC $\in$ XP.

According to the definition of fpt Turing reduction there are computable functions f, g and a $c \in \mathbb{N}$ such that for every instance $(\mathbb{M}, n)$ of p-CONSTR-ACC, the algorithm $\mathbb{R}$ will only make queries "$(\mathbb{M}', n') \in p$-ACC?" with

$$\|\mathbb{M}'\| \leq g(\|\mathbb{M}\|) \quad \text{and} \quad n' \leq f(\|\mathbb{M}\|) \cdot n^c. \tag{11}$$

There are at most $2^{g(\|\mathbb{M}\|)} \cdot g(\|\mathbb{M}\|)$ machines $\mathbb{M}'$ with $\|\mathbb{M}'\| \leq g(\|\mathbb{M}\|)$. For each such machine $\mathbb{M}'$ the answer to queries of the form "$(\mathbb{M}', n') \in p$-ACC?" with $n' \leq f(\|\mathbb{M}\|) \cdot n^c$ is determined by everyone of the following $f(\|\mathbb{M}\|) \cdot n^c + 1$ many statements: "the length of an accepting run of $\mathbb{M}'$ of minimum length is 1",..., "the length of an accepting run of $\mathbb{M}'$ of minimum length is $f(\|\mathbb{M}\|) \cdot n^c$", and "there is no accepting run of $\mathbb{M}'$ of length is $\leq f(\|\mathbb{M}\|) \cdot n^c$." Therefore the table of theoretically possible answers contains at most

$$\big(f(\|\mathbb{M}\|) \cdot n^c + 1\big)^{2^{f(\|\mathbb{M}\|) \cdot f(\|\mathbb{M}\|)}}$$

entries, that is $O(n^{h(\|\mathbb{M}\|)})$ many for some computable h. For each such possibility we simulate $\mathbb{R}$ by replacing the oracle queries accordingly. For those possibilities where $\mathbb{R}$ yields a purported accepting run of $\mathbb{M}$, we can check whether it is really an accepting run of $\mathbb{M}$. □

An analysis of the previous proof shows that we can even rule out the existence of a reduction with running time $O(|x|^{g(\kappa(x))})$ instead of $g(\kappa(x)) \cdot |x|^c$.

Furthermore, the previous theorem is a special case of a result for slicewise monotone problems: Let (Q, κ) be a slicewise monotone parameterized problem (this concept was defined just before Lemma 4) and assume that Q has a representation of the form

$$(x, n) \in Q \iff \text{there is } y \in \{0,1\}^*\text{: } \Big(|y| \leq f(|x|) \cdot n^c \text{ and } (x, n, y) \in Q_W\Big), \tag{12}$$

where $f : \mathbb{N} \to \mathbb{N}$ is computable, $c \in \mathbb{N}$ and Q_W is decidable in polynomial time. A string y with the properties on the right hand side is a *witness* for (x, n).

The construction problem p-CONSTR-(Q, κ) for every instance (x, n) of Q (with parameter $|x|$) asks for a witness of (x, n) if there is one (otherwise the nonexistence should be reported). Along the lines of the previous proofs one can show:

Proposition 19. *Let (Q, κ) be a slicewise monotone parameterized problem with a representation as in* (12)*.*
(1) p-CONSTR-(Q, κ) is nonuniformly fixed-parameter tractable.
(2) There is an fpt$_{\text{uni}}$ Turing reduction from p-CONSTR-(Q, κ) to (Q, κ).
(3) If $(Q, \kappa) \notin$ XP, then there is no fpt Turing reduction from p-CONSTR-(Q, κ) to (Q, κ).

5. Some variants of p-ACC

In Section 5.1 and Section 5.2 we analyze the complexity of the parameterized problems p-$\text{ACC}_=$ and p-DTM-EXP-ACC, respectively, where

p-$\text{ACC}_=$	
Instance:	A nondeterministic Turing machine $\mathbb{M}$ and $n \in \mathbb{N}$ in unary.
Parameter:	$\|\mathbb{M}\|$.
Question:	Does $\mathbb{M}$ accept the empty input tape in *exactly* n steps?

and

p-DTM-EXP-ACC	
Instance:	A *deterministic* Turing machine $\mathbb{M}$ and $n \in \mathbb{N}$ in unary.
Parameter:	$\|\mathbb{M}\|$.
Question:	Does $\mathbb{M}$ accept the empty input tape in at most 2^n steps?

5.1. The complexity of p-$\text{ACC}_=$. Let $\mathbb{M}$ be a nondeterministic Turing machine. By suitably adding to $\mathbb{M}$ a state, which can be accessed and left nondeterministically, one obtains a machine $\mathbb{M}^*$ such that for all $n \in \mathbb{N}$ we have:

$$\mathbb{M} \text{ accepts the empty input tape in } \leq n \text{ steps} \iff \mathbb{M}^* \text{ accepts the empty input tape in exactly } n \text{ steps.}$$

Hence p-ACC $\leq^{\text{fpt}}$ p-$\text{ACC}_=$. Recall that p-ACC $\in \text{FPT}_{\text{nu}}$. On the other hand, p-$\text{ACC}_= \notin \text{FPT}_{\text{nu}}$ if NE $\neq$ E, as shown by the following result; here NE $:=$ NTIME$(2^{O(n)})$.

Theorem 20. *The following are equivalent:*

- p-$\text{ACC}_= \notin$ FPT.
- p-$\text{ACC}_= \notin \text{XP}_{\text{nu}}$.
- NE $\neq$ E.

In [1] it is shown that p-$\text{ACC}_= \in$ XP implies NE $=$ E. We prove Theorem 20 by the following lemmas.

Lemma 21. *If* NE $=$ E, *then* p-$\text{ACC}_= \in$ FPT.

Proof: Consider the following classical problem:

Q	
Instance:	A nondeterministic Turing machine $\mathbb{M}$ and $n \in \mathbb{N}$ in *binary*.
Question:	Does $\mathbb{M}$ accept the empty input tape in exactly n many steps?

Clearly $Q \in$ NE. Hence, by the assumption NE $=$ E, we can solve Q in time

$$2^{O(\|\mathbb{M}\|+\log n)}.$$

It follows that p-$\text{ACC}_=$ is decidable in time

$$O(n) + 2^{O(\|\mathbb{M}\|+\log n)} = 2^{O(\|\mathbb{M}\|)} \cdot n^{O(1)},$$

hence it is in FPT. □

We encourage the reader to give a direct proof of the following result:

Corollary 22. *If* NE $=$ E, *then the logic* L_Y *is an effectively width* P*-bounded logic for* P.

Proof: By the previous result, if NE = E, then p-ACC$_=$ $\in$ FPT. Therefore p-ACC $\in$ FPT as p-ACC $\leq^{\text{fpt}}$ p-ACC$_=$. Now Theorem 6 implies the claim. □

Lemma 23. *If* p-ACC$_=$ $\in$ XP$_{\text{nu}}$, *then* NE = E.

Proof: Assume that p-ACC$_=$ $\in$ XP$_{\text{nu}}$ and let $Q \subseteq \{0,1\}^*$ be in NE. We have to show that $Q \in$ E. Without loss of generality we may assume that every $x \in Q$ starts with a "1." Let $n(x)$ be the natural number with binary representation x; then

$$n(x) \neq n(y) \text{ for } x, y \in Q \text{ with } x \neq y. \tag{13}$$

As $Q \in$ NE there is a nondeterministic Turing machine $\mathbb{M}$ and a $c \in \mathbb{N}$ such that for every $x \in \{0,1\}^*$ the machine $\mathbb{M}$ decides whether $x \in Q$ in time

$$2^{c \cdot |x|}$$

and every run of $\mathbb{M}$ on input x has length at most $2^{c \cdot |x|}$. Note that for x starting with a "1", we have $2^{c \cdot |x|} = n(x)^c$.

We define a nondeterministic Turing machine $\mathbb{M}^*$ that started with empty input tape runs as follows:

1. Guess a string $y \in \{0,1\}^*$
2. **if** y does not start with a "1", **then** reject
3. Simulate $\mathbb{M}$ on input y for $n(y)^c$ many steps
4. **if** $\mathbb{M}$ rejects, **then** reject
5. Make some additional dummy steps such that so far the total running time of $\mathbb{M}^*$ is $2 \cdot n(y)^c - 1$
6. Accept.

By (13) we have for every $x \in \{0,1\}^*$ starting with a "1":

$$x \in Q \iff \mathbb{M}^* \text{ accepts the empty input tape in exactly } 2 \cdot n(x)^c \text{ many steps.} \tag{14}$$

As p-ACC$_=$ $\in$ XP$_{\text{nu}}$, for some $d \in \mathbb{N}$ we can decide whether $\mathbb{M}^*$ accepts the empty string in exactly $2 \cdot n(x)^c$ many steps in time

$$(2 \cdot n(x)^c)^d.$$

Hence, $x \in Q$ can be decided in time $2^{O(|x|)}$, that is, $Q \in$ E. □

Proof of Theorem 20: Immediate by Lemma 21 and Lemma 23. □

5.2. The complexity of p-DTM-EXP-ACC. If in the problem p-ACC we replace the nondeterministic Turing machine $\mathbb{M}$ by a deterministic one simulating all computation paths of length n of $\mathbb{M}$ with empty input tape we "arrive at" p-DTM-EXP-ACC; in particular, p-ACC $\leq^{\text{fpt}}$ p-DTM-EXP-ACC. As p-DTM-EXP-ACC is a slicewise monotone parameterized problem, we know that that p-DTM-EXP-ACC $\in$ FPT$_{\text{nu}}$ by Lemma 4. Clearly, FPT$_{\text{nu}}$ $\subseteq$ XP$_{\text{nu}}$ and XP $\subseteq$ XP$_{\text{uni}}$ $\subseteq$ XP$_{\text{nu}}$. The problem p-DTM-EXP-ACC lies in FPT$_{\text{nu}}$ $\setminus$ XP$_{\text{uni}}$, as we show:

Theorem 24. p-DTM-EXP-ACC $\notin$ XP$_{\text{uni}}$.

Proof: Clearly p-DTM-EXP-ACC $\equiv^{\text{fpt}}$ p-DTM-INP-EXP-ACC, where

p-DTM-INP-EXP-ACC
Instance: A deterministic Turing machine $\mathbb{M}$, $x \in \{0,1\}^*$, and $n \in \mathbb{N}$ in unary.
Parameter: $\|\mathbb{M}\| + |x|$.
Question: Does $\mathbb{M}$ accept x in at most 2^n steps?

Thus, it suffices to show that p-DTM-INP-EXP-ACC $\notin$ XP$_{\text{uni}}$. By contradiction, let us assume that there exists an algorithm $\mathbb{A}$ that for every instance $(\mathbb{M}, x, n)$ decides whether $(\mathbb{M}, x, n) \in p$-DTM-INP-EXP-ACC in time

$$n^{f(\|\mathbb{M}\|+|x|)}$$

for some (not necessarily computable) function $f : \mathbb{N} \to \mathbb{N}$.

We denote by enc($\mathbb{M}$) the encoding of a Turing machine $\mathbb{M}$ by a string in $\{0,1\}^*$. We consider the following deterministic Turing machine $\mathbb{M}_0$:

$\mathbb{M}_0(x)$
// $x \in \{0,1\}^*$

1. **if** x is not the encoding of a deterministic Turing machine, **then** reject
2. determine the deterministic Turing machine $\mathbb{M}$ with $x = \text{enc}(\mathbb{M})$
3. $m \leftarrow$ the number of steps performed by $\mathbb{M}_0$ so far
4. Simulate $\mathbb{A}$ on $(\mathbb{M}, x, m+3)$ for at most 2^m steps
5. **if** the simulation does not halt, **then** $m \leftarrow m+1$ and goto 4
6. **if** $\mathbb{A}$ accepts $(\mathbb{M}, x, m+3)$ in at most 2^m steps **then** reject **else** accept.

We finish the proof by a diagonal argument: We set $x_0 := \text{enc}(\mathbb{M}_0)$ and start $\mathbb{M}_0$ with input x_0. For sufficiently large $m \in \mathbb{N}$ we have

$$(m+3)^{f(\|\mathbb{M}_0\|+|x_0|)} \leq 2^m.$$

Therefore eventually $\mathbb{M}_0$ with input x_0 reaches an m, we call it m_0, such that the simulation in Line 4 halts, that is,

$$\mathbb{A} \text{ halts on input } (\mathbb{M}_0, x_0, m_0+3) \text{ in at most } 2^{m_0} \text{ steps.} \tag{15}$$

At that point the number of steps (of the run of $\mathbb{M}_0$ on input x_0) is bounded by 2^{m_0+2}. Hence,

$$\mathbb{M}_0 \text{ on input } x_0 \text{ halts in } \leq 2 + 2^{m_0+2} \leq 2^{m_0+3} \text{ steps.} \tag{16}$$

Thus

$\mathbb{M}_0$ accepts x_0		
$\iff$	$\mathbb{M}_0$ accepts x_0 in $\leq 2^{m_0+3}$ steps	(by (16))
$\iff$	$\mathbb{A}$ accepts $(\mathbb{M}_0, x_0, m_0+3)$	(by definition of $\mathbb{A}$)
$\iff$	$\mathbb{A}$ accepts $(\mathbb{M}_0, x_0, m_0+3)$ in at most 2^{m_0} steps	(by (15))
$\iff$	$\mathbb{M}_0$ rejects x_0	(by Line 6 in the definition of $\mathbb{M}_0$),

the desired contradiction. □

6. On Gödel's proof predicate

The Theorem on the Undecidability of First-Order Logic tells us that the following problem is undecidable

FO-THEOREM
Instance: A first-order sentence φ.
Question: Does φ have a proof (i.e., is φ valid)?

Hence, we know that there is no computable bound on the length of shortest proofs of valid FO-sentences. By mathematicians' experience various valid FO-sentences φ only have quite long proofs, say, proofs of a length $\geq 2^{|\varphi|}$. How hard is it to recognize such "hard" valid sentences? That is, what is the complexity of the problem:

FO-EXP-PROOF	
Instance:	A first-order sentence φ and a natural number n in unary with $n \geq 2^{\lvert\varphi\rvert}$.
Question:	Does φ have a proof of length $\leq n$?

Clearly, FO-EXP-PROOF is sparse. Thus, by a result of Mahaney [8], it is not NP-hard, unless NP=P. So how do we convince ourselves that it is intractable? For this purpose, we consider the parameterized problem:

p-FO-PROOF	
Instance:	A first-order sentence φ and a natural number n in *unary*.
Parameter:	$\lvert\varphi\rvert$.
Question:	Does φ have a proof of length $\leq n$?

It is easy to see that if FO-EXP-PROOF is decidable in polynomial time, then p-FO-PROOF is fixed-parameter tractable. Hence, in this section we try to get evidence that p-FO-PROOF $\notin$ FPT. Clearly, the corresponding classical problem:

FO-PROOF	
Instance:	A first-order sentence φ and a natural number n in unary.
Question:	Does φ have a proof of length $\leq n$?

is NP-complete. This problem has been considered by Gödel and he asked whether it can be solved in time $O(n^2)$ and at the same time asked "how strongly in general the number of steps in finite combinatorial problems can be reduced with respect to simple exhaustive search" (see [3]), that is, in this context he addressed the P-NP-problem. Furthermore, he addressed the construction problem for FO-PROOF.

We turn to p-FO-PROOF. As it is a slicewise monotone problem, by Lemma 4 we have p-FO-PROOF $\in$ $\mathrm{FPT}_{\mathrm{nu}}$, a fact that is also implied by:

Proposition 25. p-FO-PROOF $\leq^{\mathrm{fpt}}$ p-ACC.

Proof: Let $\mathbb{M}_0$ be a deterministic Turing machine that on input (φ, x), where $\varphi \in$ FO and $x \in \{0,1\}^*$, checks whether x is a proof of φ. We may assume that there is a polynomial $q_0 \in \mathbb{N}[X]$ such that $\mathbb{M}_0$ on input (φ, x) halts in *exactly* $q_0(\lvert\varphi\rvert + \lvert x\rvert)$ steps.

For an FO-sentence φ we let $\mathbb{M}_\varphi$ be the Turing machine that first writes φ on a tape, then guesses a string $x \in \{0,1\}^*$, and finally simulates $\mathbb{M}_0$ on input (φ, x). One easily verifies for $\varphi \in$ FO and $n \in \mathbb{N}$ that

$$(\varphi, n) \in p\text{-FO-PROOF} \iff \Big(\mathbb{M}_\varphi, \lvert\varphi\rvert + n + q_0(\lvert\varphi\rvert + n)\Big) \in p\text{-ACC};$$

hence, $(\varphi, n) \to (\mathbb{M}_\varphi, \lvert\varphi\rvert + n + q_0(\lvert\varphi\rvert + n))$ is the desired reduction. □

Theorem 26. *If* $\mathrm{P}[\mathrm{TC}] \neq \mathrm{NP}[\mathrm{TC}]$, *then* p-FO-PROOF $\notin$ FPT.

Clearly, this result would follow from Theorem 9 if p-ACC $\leq^{\mathrm{fpt}}$ p-FO-PROOF. However we do not know whether this is true. The following lemma, which is needed for the proof of Theorem 26, has the flavour of containing such a reduction; however the sentence $\varphi_{\mathbb{M}}$ could have a short proof, even though every accepting run of $\mathbb{M}$ is long.

Lemma 27. *There exists a polynomial time algorithm* $\mathbb{A}$ *that assigns to every nondeterministic Turing machine* $\mathbb{M}$ *an* FO*-sentence* $\varphi_{\mathbb{M}}$ *such that for every* $n \in \mathbb{N}$,

$$\mathbb{M} \text{ accepts the empty input tape in} \leq n \text{ steps} \implies \varphi_{\mathbb{M}} \text{ has a proof of length} \leq n^{O(1)}. \tag{17}$$

Moreover,

$$\varphi_{\mathbb{M}} \text{ has a proof} \implies \mathbb{M} \text{ accepts the empty input tape.} \tag{18}$$

Sketch of proof: Let $\mathbb{M}$ be a nondeterministic Turing machine. As in standard proofs of the undecidability of first-order logic (e.g., compare [4, Section X.4]), one can define an FO-sentence $\varphi_{\mathbb{M}}$ of the form

$$\psi_{\mathbb{M}} \to \xi_{\mathbb{M}}$$

such that for every structure $\mathcal{A}$:

(a) $\mathcal{A} \models \psi_{\mathbb{M}}$ if and only if $\mathcal{A}$ "contains" the whole (possibly infinite) computation tree of $\mathbb{M}$ started with empty input tape.

(b) If $\mathcal{A} \models \psi_{\mathbb{M}}$, then ($\mathcal{A} \models \xi_{\mathbb{M}}$ if and only if the state of some element of the computation tree *in* $\mathcal{A}$ of $\mathbb{M}$ started with empty input tape is accepting).

To show (18) we assume $\varphi_{\mathbb{M}}$ has a proof and choose "the" structure $\mathcal{A}_0$ consisting of the whole computation tree of $\mathbb{M}$ started with empty input tape. As $\varphi_{\mathbb{M}}$ is valid, we have $\mathcal{A}_0 \models \varphi_{\mathbb{M}}$ and by (a) we know $\mathcal{A}_0 \models \psi_{\mathbb{M}}$. Hence $\mathcal{A}_0 \models \xi_{\mathbb{M}}$, which by (b) yields that $\mathbb{M}$ accepts the empty input tape.

We turn to (17) and assume that $\mathbb{M}$ accepts the empty tape in $\leq n$ steps. Using (a) and (b) one easily verifies that $\varphi_{\mathbb{M}}$ is valid. Moreover, by (a), every model of $\psi_{\mathbb{M}}$ contains an accepting path of length $\leq n$. "By following this run" we can translate it into a proof of $\psi_{\mathbb{M}} \to \xi_M$, that is of $\varphi_{\mathbb{M}}$, of length $\leq n^{O(1)}$. The details of such a translation are tedious but routine. □

Proof of Theorem 26: The proof is similar to that of Theorem 9. By contradiction assume that p-FO-PROOF $\in$ FPT. Then there is an algorithm $\mathbb{A}_0$ that for every FO-sentence φ and $n \in \mathbb{N}$ decides whether φ has a proof of length $\leq n$ in time

$$f(|\varphi|) \cdot n^{O(1)} \tag{19}$$

for some computable and increasing function $f : \mathbb{N} \to \mathbb{N}$. Furthermore, let $\mathbb{A}$ be the algorithm in Lemma 27. We choose $d \in \mathbb{N}$ such that the running time of $\mathbb{A}$ on every nondeterministic Turing machine $\mathbb{M}$ is bounded by $\|\mathbb{M}\|^d$. In particular,

$$|\varphi_{\mathbb{M}}| \leq \|\mathbb{M}\|^d. \tag{20}$$

For the function $x \mapsto f(x^d)$ we choose $h : \mathbb{N} \to \mathbb{N}$ according to Lemma 10 and show that NTIME$(h^{O(1)}) \subseteq$ DTIME$(h^{O(1)})$.

For this purpose let $Q \subseteq \{0,1\}^*$ be in NTIME$(h^{O(1)})$. We choose a nondeterministic Turing machine $\mathbb{M}_Q$ and constants $c, d \in \mathbb{N}$ such that the machine $\mathbb{M}_Q$ decides whether $x \in Q$ in time $c \cdot h(|x|)^d$ and every run of $\mathbb{M}_Q$ on input x is $c \cdot h(|x|)^d$ time-bounded (recall that h is time constructible).

We fix $x \in \{0,1\}^*$ and let $\mathbb{M}_{Q,x}$ be the nondeterministic Turing machine that, started with empty input tape, first writes x on some tape and then simulates $\mathbb{M}_Q$ started with x. We apply the algorithm $\mathbb{A}$ on $\mathbb{M}_{Q,x}$ and get an FO-sentence $\varphi := \varphi_{\mathbb{M}_{Q,x}}$. Now we have the following chain of implications (as the first statement and the last one coincide all implications are equivalences):

$x \in Q$
- $\Rightarrow$ $\mathbb{M}_Q$ accepts x in at most $c \cdot h(|x|)^d$ steps
- $\Rightarrow$ $\mathbb{M}_{Q,x}$ accepts the empty string in at most $|x| + c \cdot h(|x|)^d$ steps (by definition of $\mathbb{M}_{Q,x}$)
- $\Rightarrow$ $\mathbb{M}_{Q,x}$ accepts the empty string in at most $2c \cdot h(|x|)^d$ steps
- $\Rightarrow$ φ has a proof of length at most $(2c \cdot h(|x|))^{O(1)}$ (by (17) in Lemma 27)
- $\Rightarrow$ $\mathbb{M}_{Q,x}$ accepts the empty string (by (18) in Lemma 27)
- $\Rightarrow$ $\mathbb{M}_Q$ accepts x (by definition of $\mathbb{M}_{Q,x}$)
- $\Rightarrow$ $x \in Q$.

Hence,

$$x \in Q \iff \mathbb{A} \text{ accepts } \big(\varphi, (2c \cdot h(|x|))^{O(1)}\big).$$

As the size of $\mathbb{M}_{Q,x}$ is $O(\|\mathbb{M}_Q\|) + |x| \cdot \log |x|$, by (20) we have

$$\|\varphi\| \leq \|\mathbb{M}_{Q,x}\|^d = (O(\|\mathbb{M}_Q\|) + |x|^2)^d. \tag{21}$$

By (19) the running time of $\mathbb{A}$ on input $(\varphi, (2c \cdot h(|x|))^{O(1)})$ is bounded by

$$f(|\varphi|) \cdot (2c \cdot h(|x|))^{O(1)} \leq f((O(\|\mathbb{M}_Q\|) + |x|^2)^d) \cdot h(|x|)^{O(1)} \leq (h(O(\|\mathbb{M}_Q\|)) + h(|x|)) \cdot h(|x|)^{O(1)}.$$

This shows that $Q \in \text{DTIME}(h(|x|)^{O(1)})$. □

Remark 28. As we did in Theorem 11 for p-ACC we can refine the previous proof and show:

Assume that

$$\text{NTIME}(h^{O(1)}) \not\subseteq \text{DTIME}(h^{O(\log h)}).$$

for every time constructible and increasing function h. Then p-FO-PROOF $\notin$ XP.

We turn to the construction problem associated with p-FO-PROOF, that is, to:

p-CONSTR-FO-PROOF	
Instance:	A first-order sentence φ and a natural number n in unary.
Parameter:	$\|\varphi\|$.
Problem:	Construct a proof of φ of length $\leq n$ if there exists one.

It is well-known that the classical version of p-CONSTR-FO-PROOF, namely

CONSTR-FO-PROOF	
Instance:	An FO-sentence φ and a natural number n.
Problem:	Construct a proof of φ of length $\leq n$ if there exists one.

is NP-complete. Hence

Theorem 29. *There is a polynomial time Turing reduction from* CONSTR-FO-PROOF *to* FO-PROOF.

Assume that p-FO-PROOF $\notin$ FPT. Let φ be a first-order sentence with a proof of length n. Then part (4) of the next theorem shows that there is no efficient way to find such a proof even if we are given all the theorems of a length bounded by $g(\varphi)$ for some fixed computable g. Hence the polynomial time Turing reduction of the preceding theorem, in general will ask queries "$(\psi, m) \in$ FO-PROOF?" for sentences ψ with $|\psi|$ only bounded by some polynomial in n, the length of a proof of φ.

Theorem 30. *(1) p-CONSTR-FO-PROOF is nonuniformly fixed-parameter tractable.*

(2) There is an fpt$_{\text{uni}}$ Turing reduction from p-CONSTR-FO-PROOF to p-FO-PROOF.

(3) If p-FO-PROOF $\notin$ XP, then there is no fpt Turing reduction from p-CONSTR-FO-PROOF to p-FO-PROOF.

(4) If p-FO-PROOF $\notin$ FPT, then there is no fpt-algorithm $\mathbb{A}$ with an oracle to FO-THEOREM *(cf. page 15) that solves the problem p-CONSTR-FO-PROOF in such a way that for some computable function $g : \mathbb{N} \to \mathbb{N}$ on every instance (φ, n) of p-CONSTR-FO-PROOF the algorithm $\mathbb{A}$ only makes oracle queries "$\psi \in$ FO-THEOREM?" for ψ with $|\psi| \leq g(|\varphi|)$.*

Proof: As p-FO-PROOF is slicewise monotone, parts (1) to (3) are special instances of Proposition 19.

We turn to a proof of part (4). By contradiction assume that there is an algorithm $\mathbb{A}$ with an oracle to FO-THEOREM that solves the problem p-CONSTR-FO-PROOF in such a way that for some computable function $g : \mathbb{N} \to \mathbb{N}$ we have for every instance (φ, n) of p-CONSTR-FO-PROOF

(a) the run of $\mathbb{A}$ on input (φ, n) has length $\leq g(|\varphi|) \cdot n^c$;

(b) for every oracle query "$\psi \in$ FO-THEOREM?" of the run of $\mathbb{A}$ on input (φ, n) we have $|\psi| \leq g(|\varphi|)$.

We show how $\mathbb{A}$ can be turned into an algorithm witnessing p-FO-PROOF $\in$ FPT.

For an FO-sentence φ we compute an enumeration $\psi_1, \psi_2, \ldots, \psi_k$ of all FO-sentences of length $\leq g(|\varphi|)$. By (b) every oracle query of the run of $\mathbb{A}$ on input (φ, n) has the form "$\psi_i \in$ FO-THEOREM?" for some $i \in [k]$. For each such question there are two possibilities: "$\psi_i \in$ FO-THEOREM" and "$\psi_i \notin$ FO-THEOREM." Thus for all ψ_i together we have

$$2^k$$

possibilities, that is, $O(2^{f(|\varphi|)})$ many for some computable f (note that k only depends on φ). Now, given in addition $n \in \mathbb{N}$, for each such possibility we simulate $\mathbb{A}$ on input (φ, n) by replacing the oracle queries accordingly. For those possibilities where $\mathbb{A}$ yields a purported proof, we can check whether it is really a proof of φ of length $\leq n$. By (a) the overall time needed by this procedure is $O(2^{f(|\varphi|)} \cdot (g(|\varphi|) \cdot n^c + n^{O(1)}))$, which is an fpt-time. □

Acknowledgements. We thank Martin Grohe for drawing our attention to the problem p-ACC and for some discussions on the topic. Furthermore, we thank Christian Glaßer for pointing out to us the relationship of the property derived in Lemma 12 with bi-immune sets and to Stefan Kreutzer for some discussions on least fixed-point logic.

References

[1] Y. Aumann and Y. Dombb. Fixed structure complexity. In *Proceedings of the 3rd International Workshop on Parameterized and Exact Computation (IWPEC 2008)*, M. Grohe and R. Niedermeier (eds.), Lecture Notes in Computer Science 5018, 31–42, 2008.

[2] A. Blass, Y. Gurevich, and S. Shelah. Choiceless polynomial time. *Annals of Pure and Applied Logic*, 100:141–187, 1999.

[3] S. Buss. On Gödel's Theorem on length of proofs II: lower bounds for recognizing k symbol provability. In *Feasible Mathematics II*, P. Clote and J. Remmel (eds.), Birkhauser, 57–90, 1995.

[4] H.-D. Ebbinghaus, J. Flum, and W. Thomas, *Mathematical Logic*, 2nd edition Springer, 1994.

[5] H.-D. Ebbinghaus and J. Flum. *Finite Model Theory*, 2nd edition, Springer, 1999.

[6] J. Flum and M. Grohe. *Parameterized Complexity Theory*, Springer, 2006.

[7] J. M. Hitchcock, J. H. Lutz, and E. Mayordomo. The fractal geometry of complexity classes. In the Complexity Theory Column, L.A. Hemaspaandra (ed.), *SIGACT News* 36, 24–38, 2005.

[8] S. R. Mahaney. Sparse complete sets of NP: solution of a conjecture of Berman and Hartmanis. *Journal of Computer and System Science*, 25(2): 130–143, 1982.

[9] E. Mayordomo. Almost every set in exponential time is P-bi-immune. *Theoretical Computer Science*, 136(2): 487-506, 1994.

[10] A. Nash, J. Remmel, and V. Vianu. PTIME queries revisited. In *Proceedings of the 10th International Conference on Database Theory (ICDT 2005)*, T. Eiter and L. Libkin (eds.), Lecture Notes in Computer Science 3363, 274–288, 2005.

[11] M.Y. Vardi. On the complexity of bounded-variable queries. In *Proceedings of the 14th ACM Symposium on Principles of Database Systems*, pages 266–276, 1995.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de/