Complexity of Existential Positive First-Order Logic

Manuel Bodirsky, Miki Hermann, and Florian Richoux

LIX (CNRS, UMR 7161), École Polytechnique, 91128 Palaiseau, France.
{bodirsky, hermann, richoux}@lix.polytechnique.fr

Abstract. Let Γ be a (not necessarily finite) structure with a finite relational signature. We prove that deciding whether a given existential positive sentence holds in Γ is in LOGSPACE or complete for the class $\mathrm{CSP}(\Gamma)_{\mathrm{NP}}$ under deterministic polynomial-time many-one reductions. Here, $\mathrm{CSP}(\Gamma)_{\mathrm{NP}}$ is the class of problems that can be reduced to the *constraint satisfaction problem* of Γ under *non-deterministic* polynomial-time many-one reductions.

Key words: Computational Complexity, Existential Positive First-Order Logic, Constraint Satisfaction Problems

1 Introduction

We study the computational complexity of the following class of computational problems. Let Γ be a structure with finite or infinite domain and with a finite relational signature. The model-checking problem for existential positive first-order logic, parametrized by Γ, is the following problem.

Problem: $\textsc{ExPos}(\Gamma)$
Input: An existential positive first-order sentence Φ.
Question: Does Γ satisfy Φ?

A first-order sentence is *existential positive* if it does not contain universal quantifiers and negation symbols, that is, if the only logical connectives are existential quantifiers, disjunction, conjunction, and equality. The sentence does not need to be in prenex normal form; however, every existential positive first-order sentence can be transformed in an equivalent one in this form without an exponential blowup, thanks to the absence of universal quantifiers and negation symbols.

The *constraint satisfaction problem* $\mathrm{CSP}(\Gamma)$ for Γ is defined similarly, but its input consists of a *primitive positive* sentence, that is, a sentence without universal quantifiers, negation, and disjunction. Constraint satisfaction problems frequently appear in many areas of computer science, and have attracted a lot of attention, in particular in combinatorics, artificial intelligence, and finite model theory; we refer to the recent monograph with survey articles on this subject [7]. The class of constraint satisfaction problems for infinite structures Gamma is a rich class of problems; it can be shown that for every computational problem there exists a relational structure Gamma such that CSP(Gamma) is equivalent to that problem under polynomial-time Turing reductions [1].

In this paper, we show that the complexity classification for existential positive first-order sentences over infinite structures can be reduced to the complexity classification for constraint satisfaction problems. For finite structures Γ, our result implies that $\text{EXPOS}(\Gamma)$ is in LOGSPACE or NP-complete.

For finite Γ, the polynomial-time solvable cases of $\text{EXPOS}(\Gamma)$ are precisely those relational structures Γ with an element a where all relations in Γ contain the tuple $(a, \ldots, a)$ composed only from the element a; in this case, $\text{EXPOS}(\Gamma)$ is called *a-valid*. Interestingly, this is no longer true for infinite structures Γ.

Consider the structure $\Gamma := (\mathbb{N}, \neq)$, which is clearly not a-valid. However, $\text{EXPOS}(\Gamma)$ can be reduced to the Boolean formula evaluation problem (which is known to be in LOGSPACE) as follows: atomic formulas in Φ of the form $x \neq y$ are replaced by *true*, and atomic formulas of the form $x \neq x$ are replaced by *false*. The resulting Boolean formula is equivalent to true if and only if Φ is true in Γ.

A universal-algebraic study of the model-checking problem for finite structures Γ and various other syntactic restrictions of first-order logic (for instance positive first-order logic) can be found in [6].

2 Result

We write $L \leq_m L'$ if there exists a deterministic polynomial-time many-one reduction from L to L'.

Definition 1 (from [4]). *A problem A is* non-deterministic polynomial-time many-one reducible *to a problem B ($A \leq_{\text{NP}} B$) iff there is a nondeterministic polynomial-time Turing machine M such that $x \in A$ if and only if there exists a y computed by M on input x with $y \in B$. We denote by A_{NP} the smallest class that contains A and is closed under $\leq_{\text{NP}}$.*

Observe that $\leq_{\text{NP}}$ is transitive [4]. To state the complexity classification for existential positive first-order logic, we need the following concepts. Let Φ be an existential positive τ-sentence for a finite relational signature τ. We construct a Boolean formula $F_\Gamma(\Phi)$ as follows. We first remove all existential quantifiers from Φ. Then we replace each atomic formula in Φ of the form $R(x_1, \ldots, x_k)$ where R denotes the empty k-ary relation over Γ by *false*. All other atomic formulas in Φ will be replaced by *true*. We write $F_\Gamma(\Phi)$ for the resulting Boolean formula. Note that if Φ is true in Γ then $F_\Gamma(\Phi)$ must be logically equivalent to *true*.

Definition 2. *We call a relational structure Γ* locally refutable *if every existential positive sentence Φ is true in Γ* if and only if *the Boolean formula $F(\Phi)$ (as described above) is logically equivalent to* true.

In Section 3, we will show the following result.

Theorem 3. *Let Γ be a structure with a finite relational signature τ. If Γ is locally refutable then the problem $\text{EXPOS}(\Gamma)$ to decide whether an existential positive sentence is true in Γ is in* LOGSPACE. *If Γ is not locally refutable, then $\text{EXPOS}(\Gamma)$ is complete for the class $\text{CSP}(\Gamma)_{\text{NP}}$ under polynomial-time many-one reductions.*

In particular, $\text{EXPOS}(\Gamma)$ is in P or is NP-hard (under deterministic polynomial-time many-one reductions). If Γ is finite, then $\text{EXPOS}(\Gamma)$ is in P or NP-complete, because finite domain constraint satisfaction problems are clearly in NP. The observation that $\text{EXPOS}(\Gamma)$ is in P or NP-complete has previously been made in [3] and independently in [5]. However, our proof remains the same for finite domains and is simpler than proofs in these previous works.

3 Proof

Before we prove Theorem 3, we start with the following simpler result.

Theorem 4. *Let Γ be a structure with a finite relational signature τ. If Γ is locally refutable, then the problem $\text{EXPOS}(\Gamma)$ to decide whether an existential positive sentence is true in Γ is in* LOGSPACE*. If Γ is not locally refutable, then $\text{EXPOS}(\Gamma)$ is* NP*-hard (under polynomial-time many-one reductions).*

To show Theorem 4, we first prove the following lemma.

Lemma 5. *If Γ is not locally refutable, then there are existential positive τ-formulas ψ_0 and ψ_1 with the property that*

- *ψ_0 and ψ_1 define non-empty relations over Γ;*
- *$\psi_0 \wedge \psi_1$ defines the empty relation over Γ.*

Proof. Because Γ is not locally refutable, there is an unsatisfiable instance Φ of $\text{EXPOS}(\Gamma)$ such that the Boolean formula $F(\Phi)$ described above is logically equivalent to *true*. Among all formulas with this property, let Φ be the one that is of minimal length.

If Φ is of the form $\Phi_1 \vee \Phi_2$ then both Φ_1 and Φ_2 are unsatisfiable over Γ, and one of the Boolean formulas $F_\Gamma(\Phi_1)$ or $F_\Gamma(\Phi_2)$ must be true; this contradicts the assumption that Φ is minimal.

If Φ is of the form $\Phi_1 \wedge \Phi_2$, then both $F_\Gamma(\Phi_1)$ and $F_\Gamma(\Phi_2)$ are true. If both Φ_1 and Φ_2 are satisfiable, then we are done, because we have found two satisfiable existential positive formulas such that their conjunction is unsatisfiable. If Φ_1 or Φ_2 is unsatisfiable over Γ, say Φ_i is unsatisfiable for $i \in \{1, 2\}$, then this contradicts the assumption that Φ is minimal, because Φ_i is smaller than Φ, unsatisfiable over Γ, and $F_\Gamma(\Phi_i)$ is true. If Φ is of the form $\exists x.\Phi'$ then this contradicts obviously the assumption that Φ is minimal. Note that Φ cannot be atomic, because in this case Φ is either unsatisfiable or $F_\Gamma(\Phi)$ is true (but not both). □

Proof of Theorem 4: If Γ is locally refutable, then $\text{EXPOS}(\Gamma)$ can be reduced to the positive Boolean formula evaluation problem, which is known to be LOGSPACE-complete. We only have to construct from an existential positive τ-sentence Φ a Boolean formula $F := F(\Phi)$ as described before Definition 2. Clearly, this construction can be performed with logarithmic work-space. We evaluate F, and reject if F is false, and accept otherwise.

If Γ is not locally refutable, we show NP-hardness of $\text{ExPos}(\Gamma)$ by reduction from 3-SAT. Let I be a 3-SAT instance. We construct an instance Φ of $\text{ExPos}(\Gamma)$ as follows. Let ψ_0 and ψ_1 be the formulas from Lemma 5 (suppose they are d-ary). Let $v_1, \ldots, v_n$ be the Boolean variables in I. For each v_i we introduce d new variables $\bar{x}_i = x_i^1, \ldots, x_i^d$. Let Φ be the instance of $\text{ExPos}(\Gamma)$ that contains the following conjuncts:

- For each $1 \leq i \leq n$, the formula $\psi_0(\bar{x}_i) \vee \psi_1(\bar{x}_i)$
- For each clause $l_1 \vee l_2 \vee l_3$ in I, the formula $\psi_{i_1}(\bar{x}_{j_1}) \vee \psi_{i_2}(\bar{x}_{j_2}) \vee \psi_{i_3}(\bar{x}_{j_3})$ where $i_p = 0$ if l_p equals $\neg x_{j_p}$ and $i_p = 1$ if l_p equals x_{j_p}, for all $p \in \{1, 2, 3\}$.

It is clear that Φ can be computed in deterministic polynomial time from I, and that Φ is true in Γ if and only if I is satisfiable. □

Note that, applied to finite domain constraint languages Γ, we obtain again the dichotomy from [3] and [5].

Proof of Theorem 3: If Γ is locally refutable then the statement has been shown in Theorem 4. Suppose that Γ is not locally refutable. To show that $\text{ExPos}(\Gamma)$ is contained in $\text{CSP}(\Gamma)_{\text{NP}}$, we construct a non-deterministic Turing machine T which takes as input an instance Φ of $\text{ExPos}(\Gamma)$, and which outputs an instance $T(\Phi)$ of $\text{CSP}(\Gamma)$ as follows.

On input Φ the machine T proceeds recursively as follows:

- if Φ is of the form $\exists x.\varphi$ then return $\exists x.T(\varphi)$;
- if Φ is of the form $\varphi_1 \wedge \varphi_2$ then return $T(\varphi_1) \wedge T(\varphi_2)$;
- if Φ is of the form $\varphi_1 \vee \varphi_2$ then non-deterministically return either $T(\varphi_1)$ or $T(\varphi_2)$;
- if Φ is of the form $R(x_1, \ldots, x_k)$ then return $R(x_1, \ldots, x_k)$.

The output of T can be viewed as an instance of $\text{CSP}(\Gamma)$, since it can be transformed to a primitive positive τ-sentence (by moving all existential quantifiers to the front). It is clear that T has polynomial running time, and that Φ is true in Γ if and only if there exists a computation of T on Φ that computes a sentence that is true in Γ.

We now show that $\text{ExPos}(\Gamma)$ is hard for $\text{CSP}(\Gamma)_{\text{NP}}$ under $\leq_m$-reductions. Let L be a problem with a non-deterministic polynomial-time many-one reduction to $\text{CSP}(\Gamma)$, and let M be the non-deterministic Turing machine that computes the reduction. We have to construct a deterministic Turing machine M' that computes for any input string s in polynomial time in $|s|$ an instance Φ of $\text{ExPos}(\Gamma)$ such that Φ is true in Γ if and only if there exists a computation of M on s that computes a satisfiable instance of $\text{CSP}(\Gamma)$.

Say that the running time of M on s is in $O(|s|^e)$ for a constant e. Hence, there are constants s_0 and c such that for $|s| > s_0$ the running time of M and hence also the number of constraints in the input instance of $\text{CSP}(\Gamma)$ produced by the reduction is bounded by $t := c|s|^e$. The non-deterministic computation of M can be viewed as a deterministic computation with access to non-deterministic advice bits as shown in [2]. We also know that for $|s| > s_0$, the machine M can access at most t non-deterministic bits. If w is a sufficiently long bit-string, we write M_w for the deterministic Turing machine obtained from M by using the bits in w as the non-deterministic bits, and $M_w(s)$ for the instance of $\text{CSP}(\Gamma)$ computed by M_w on input s.

If $|s| \leq s_0$, then M' returns $\exists x.(x = x)$ if there is an $w \in \{0,1\}^*$ such that $M_w(s)$ is a satisfiable instance of $\mathrm{CSP}(\Gamma)$, and M' returns $\exists \bar{x}.\psi_0(\bar{x}) \wedge \psi_1(\bar{x})$ otherwise (i.e., it returns a false instance of $\mathrm{ExPos}(\Gamma)$; ψ_0 and ψ_1 are defined in Lemma 5). Since s_0 is a fixed finite value, M' can perform these computations in constant time.

It is convenient to assume that Γ has just a single relation R (we can always find a CSP which is deterministic polynomial-time equivalent and where the template is of this form[1]). Let l be the arity of R. Then instances of $\mathrm{CSP}(\Gamma)$ with variables $x_1, \ldots, x_n$ can be encoded as sequences of numbers that are represented by binary strings of length $\lceil \log t \rceil$ as follows: The i-th number m in this sequence indicates that the $(((i-1) \bmod l) + 1)$-st variable in the $(((i-1)\ div\ l) + 1)$-st constraint is x_m.

For $|s| > s_0$, the sentence Φ computed by M' has the form

$$\exists x_1^1, \ldots, x_l^t.\ (\bigwedge_{i=1}^{t} R(x_1^i, \ldots, x_l^i) \wedge \Psi)\ . \tag{1}$$

where Ψ is an $\mathrm{ExPos}(\Gamma)$ formula defined below.

The **idea** is that *any* instance of $\mathrm{CSP}(\Gamma)$ computed by the machine M can be obtained by contracting variables in $\bigwedge_{i \leq t} R(x_1^i, \ldots, x_l^i)$. The way this is done is controlled by a Boolean formula that can be computed from the input s of M in polynomial time. The Boolean formula also contains Boolean variables for the non-deterministic advice bits of M. Each Boolean variable v in the formula is simulated by a d-tuple $\bar{x}_v$ of variables in Ψ that is forced to satisfy $\psi_0(\bar{x}_v) \vee \psi_1(\bar{x}_v)$ (ψ_0 and ψ_1 are defined in Lemma 5), similarly as in the proof of Theorem 4, such that $v = 0$ corresponds to falsity of $\psi_0(\bar{x}_v)$, and $v = 1$ corresponds to truth of $\psi_1(\bar{x}_v)$ in Γ. The formula Φ will be such that there exists a computation of M that produces a satisfiable instance I of $\mathrm{CSP}(\Gamma)$ if and only if there exists an assignment to $x_1^1, \ldots, x_l^t$ that satisfies Ψ and such that $\bigwedge_{i \leq t} R(x_1^i, \ldots, x_l^i)$ is equivalent to I.

We now provide the details of the definition of the machine M' that computes Φ. We use a construction from the proof of Cook's theorem given in [2]. In this proof, a computation of a non-deterministic Turing machine T accepting a language L is encoded by Boolean variables that represent the state and the position of the read-write head of T at time r, and the content of the tape at position j at time r. The tape content at time 0 consists of the input x, written at positions 1 through n, and the non-deterministic advice bit string w, written at positions -1 through $-|w|$. The proof in [2] specifies a deterministic polynomial-time computable transformation f_L that computes for a given string s a SAT instance $f_L(s)$ such that there is an accepting computation of T on s if and only if there is a satisfying truth assignment for $f_L(s)$.

In our case, the machine M computes a reduction and thus computes an output string. Recall our binary representation of instances of the CSP: M writes on the output tape a sequence of numbers represented by binary strings of length $\lceil \log t \rceil$. It is straightforward to modify the transformation f_L given in the proof of Theorem 2.1 in [2] to

[1] If $\Gamma = (D; R_1, \ldots, R_n)$ where R_i has arity r_i and is not empty, then $\mathrm{CSP}(\Gamma)$ is equivalent to $\mathrm{CSP}(D; R_1 \times \cdots \times R_n)$ where $R_1 \times \cdots \times R_n$ is the $\sum_{i=1}^{n} r_i$-ary relation defined as the Cartesian product of the relations $R_1, \ldots, R_n$. Similarly, $\mathrm{ExPos}(\Gamma)$ is equivalent to $\mathrm{ExPos}(D; R_1 \times \cdots \times R_n)$.

obtain for all positive integers a, a', b, b', c, c' where $a, a' \leq t$, $b, b' \leq l$, $c, c' \leq \lceil \log t \rceil$ a deterministic polynomial-time transformation $g_{a,a',b,b',c,c'}$ that computes for a given string s a SAT instance $g_{a,a',b,b',c,c'}(s)$ with distinguished variables $z_1, \ldots, z_t$ (for the non-deterministic bits in the computation of M) such that the following are equivalent:

- $g_{a,a',b,b',c,c'}(s)$ has a satisfying assignment where z_i is set to $w_i \in \{0,1\}$ for $1 \leq i \leq t$;
- the c-th bit in the b-th variable of the a-th constraint in $M_w(s)$ equals the c'-th bit in the b'-th variable of the a'-th constraint in $M_w(s)$.

We use the transformations $g_{a,a',b,b',c,c'}$ to define M' as follows. The machine M' first computes the formulas $g_{a,a',b,b',c,c'}(s)$. For every Boolean variable v in these formulas we introduce a new conjunct $\varphi_0(\overline{x}_v) \vee \varphi_1(\overline{x}_v)$ where $\overline{x}_v$ is a d-tuple of fresh variables. Then, every positive literal $l = x_j$ in the original conjuncts of the formula is replaced by $\varphi_1(\overline{x}_j)$, and every negative literal $l = \neg x_j$ by $\varphi_0(\overline{x}_j)$. We then existentially quantify over all variables except for $x_{z_1}, \ldots, x_{z_t}$. Let $\psi_{a,a',b,b',c,c'}(s)$ denote the resulting existential positive formula. It is clear that the formula

$$\exists x_{z_1}, \ldots, x_{z_t}. \bigwedge_{a,a',b,b'} \left(\left(\bigwedge_{c,c'} \psi_{a,a',b,b',c,c'} \right) \rightarrow x_b^a = x_{b'}^{a'} \right)$$

can be re-written in existential positive form Ψ without blow-up (we can replace implications $\alpha \rightarrow \beta$ by $\neg\alpha \vee \beta$, and then move the negation to the atomic level, where we can remove it by exchanging the role of φ_0 and φ_1), and hence Ψ can be computed by M' in polynomial time. The formula Ψ indeed has the properties required for the formula Ψ mentioned in Equation 1. □

References

1. M. Bodirsky and M. Grohe. Non-Dichotomies in Constraint Satisfaction Complexity. *Proceedings 35th International Colloquium on Automata, Languages and Programming (ICALP 2008), Part II, Reykjavik (Iceland)*, 5126, 184–196, 2008.
2. M. R. Garey and D. S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. *W.H. Freeman and Co*, 1979.
3. M. Hermann and F. Richoux. On the Computational Complexity of Monotone Constraint Satisfaction Problems. *Proceedings 3rd Annual Workshop on Algorithms and Computation (WALCOM 2009), Kolkata (India)*, 2009.
4. R. E. Ladner, N. A. Lynch and A. L. Selman. A Comparison of Polynomial-Time Reducibilities. *Theoretical Computer Science*, 1(2), 103–124, 1975.
5. B. Martin. Dichotomies and Duality in First-order Model Checking Problems. *CoRR abs/cs/0609022*, 2006.
6. B. Martin. First-Order Model Checking Problems Parameterized by the Model. *Proceedings 4th Conference on Computability in Europe (CiE 2008), Athens (Greece)*, 417–427, 2008.
7. H. Vollmer. Complexity of Constraints (A collection of survey articles). *Springer, LNCS 5250*, 2008.