On the Complexity of Model Expansion *

Antonina Kolokolova[1], Yongmei Liu[2], David Mitchell[3], Eugenia Ternovska[3]

[1]Memorial University of Newfoundland, Canada
[2]Sun Yat-sen University, China
[3]Simon Fraser University, Canada
kol@cs.mun.ca, ymliu@mail.sysu.edu.cn, mitchell@cs.sfu.ca, ter@cs.sfu.ca

Abstract. We study the complexity of model expansion (MX), which is the problem of expanding a given finite structure with additional relations to produce a finite model of a given formula. This is the logical task underlying many practical constraint languages and systems for representing and solving search problems, and our work is motivated by the need to provide theoretical foundations for these. We present results on both data and combined complexity of MX for several fragments and extensions of FO that are relevant for this purpose, in particular the guarded fragment GF_k of FO and extensions of FO and GF_k with inductive definitions. We present these in the context of the two closely related, but more studied, problems of model checking and finite satisfiability. To obtain results on FO(ID), the extension of FO with inductive definitions, we provide translations between FO(ID) with FO(LFP), which are of independent interest.

1 Introduction

Fagin's theorem, which states that existential second order logic ($\exists$SO) exactly captures the complexity class NP [6], initiated the area of descriptive complexity theory [14], the study of the relationship between logics and complexity classes. Descriptive complexity results can be seen as providing a way to view logics as "declarative programming languages" for problems in the corresponding classes. In [19], the third and fourth authors proposed explicitly taking this idea as the formal basis on which to build practical tools for representing and solving search problems. On this view, a problem specification is a formula ϕ in some suitable logic $\mathcal{L}$, a problem instance is a finite structure $\mathcal{A}$ for some part of the vocabulary of ϕ (the "instance vocabulary"). $\mathcal{A}$ is a model of $\exists \bar{S} \phi$, where $\bar{S}$ is the remaining ("expansion") vocabulary of ϕ, and a solution for $\mathcal{A}$ is a witness for the second order existential quantifiers. Thus, problem solving entails expanding a given structure to satisfy a given formula. It is not too hard to see that this is the formal task underlying a wide variety of actual constraint languages and systems.

The major emphasis of most practical systems is on NP search problems. Fagin's theorem tell us FO has the right expressive power for a specification language for NP, and the effectiveness of modern SAT solvers provides an easy way to build a solver via grounding to SAT. However, practical specification languages require a variety of

* Earlier versions of this work were presented at LCC 2006 and LaSh 2006. The work presented here was carried out while the first two authors were PIMS post-doctoral fellows at Simon Fraser University.

convenience features, so we are interested model expansion for logics which would provide a natural formal basis for such features. The main purpose of this paper is to summarize what is known about model expansion for the restriction of FO to guarded fragments (to support types and efficient grounding), the extension of FO with induction as in FO(ID) [4] and FO(LFP), and the combination of the two, filling in gaps as needed.

Table 1 presents the summary, placing facts about model expansion in the context of the two related, but more familiar, tasks of model checking and satisfiability in the finite. In the table, new results are presented in bold. We use K-c to denote completeness for complexity class K, and $\equiv_C$K to denote capturing of complexity class K on a class C of finite structures (see Definition 2).

Logic	**Model checking**		**Model expansion**		**Satisfiability**
	Combined	**Data**	**Combined**	**Data**	
FO	PSPACE-c [21]	$\equiv_{BIT}$AC^0 [2]	NEXP-c [25, 19]	$\equiv$NP [6]	Undecidable [24]
FO(LFP)	EXP-c [25]	$\equiv_s$P [13, 25, 18]	NEXP-c	$\equiv$NP	Undecidable
FO(ID)	**EXP-c**	$\equiv_{\mathbf{s}}$ **P**	**NEXP-c**	$\equiv$NP	Undecidable
FO^k	P-c [26, 11]	$\in$ AC^0	NP-c [26]	NP-c, $\not\equiv$NP	$k\geq 3$: Undecidable $k=2$: NEXP-c [10] $k=1$: EXP-c
GF$_k$	P-c [11, 7]	$\in$ AC^0	**NEXP-c**	$k\geq 2$: $\equiv$**NP** $k=1$: NP-c	$k\geq 2$:Undecidable $k=1$:2EXP-c [9]
RGF$_k$	na	na	**NP-c**	**NP-c**, $\not\equiv$**NP**	na
μGF	UP $\cap$ co-UP [15]	$\in$ P	NEXP-c	NP-c	2EXP-c [12]
GF$_k$(ID)	$\in$EXP	$\in$P	NEXP-c	$k\geq 2$: $\equiv$NP	$k\geq 2$:Undecidable

Table 1. Complexity of model checking, model expansion and satisfiability problems, for FO and related logics on finite structures. $\equiv_C$K denotes capturing of complexity class K on class C of finite structures; K-c denotes completeness for class K. New results are presented in bold; other results are provided with references or are easy corollaries.

2 Definitions

For each logic $\mathcal{L}$, we consider three tasks: satisfiability, model checking, and model expansion, all for finite structures. For model checking and model expansion, we consider two notions of complexity, so-called data complexity and combined complexity. In this section we present definitions for these. We denote by $vocab(\phi)$ the vocabulary of formula ϕ.

Definition 1. *For a given logic $\mathcal{L}$, we consider complexity of three problems.*

1. Model Checking*: given $(\mathcal{A}, \phi)$, where ϕ is a sentence of $\mathcal{L}$ and $\mathcal{A}$ is a finite structure for $vocab(\phi)$, does $\mathcal{A} \models \phi$?*

2. Model Expansion *(MX): given $(\mathcal{A}, \phi)$, where ϕ is a sentence in L, $\mathcal{A}$ is a finite σ-structure where $\sigma \subset vocab(\phi)$, is there an expansion $\mathcal{A}'$ of $\mathcal{A}$ to $vocab(\phi)$ such that $\mathcal{A}' \vDash \phi$?*
3. Finite Satisfiability*: given a sentence ϕ in L, is there a finite $\mathcal{A}$ for $vocab(\phi)$ such that $\mathcal{A} \vDash \phi$?*

The first and the last of these problems have been studied for a long time. Our focus is on model expansion.

Example 1. Let $\mathcal{A}$ be a graph $G = (V; E)$, and let ϕ be

$$\forall x \forall y \, [(Clique(x) \wedge Clique(y)) \supset (x = y \vee E(x, y))].$$

If $\mathcal{B}$ is an expansion of $\mathcal{A}$ to $vocab(\phi)$, then $\mathcal{B} \models \phi$ iff $Clique^{\mathcal{B}}$ is a set of vertices that forms a clique in $\mathcal{B}$.

For model checking and model expansion we consider two notions of complexity. These were introduced by [25]; here we follow the presentation of [16]. Let $enc()$ denote some standard encoding of structures and formulas as binary strings.

Definition 2. *Let K be a complexity class and $\mathcal{L}$ a logic. Let P be the problem of model checking or model expansion.*

1. *The* data complexity *of P for $\mathcal{L}$ is K if for every sentence ϕ of $\mathcal{L}$ the language $\{enc(\mathcal{A}) | (\mathcal{A}, \phi) \in P\}$ belongs to K. The* data complexity *of P for $\mathcal{L}$ is K-hard if for some sentence ϕ of $\mathcal{L}$ the language $\{enc(\mathcal{A}) | (\mathcal{A}, \phi) \in P\}$ is K-hard. The* combined complexity *of $\mathcal{L}$ is K (resp. K-hard) if the language $\{(enc(\mathcal{A}), enc(\phi)) \mid (\mathcal{A}, \phi) \in P\}$ belongs to K (resp. is K-hard).*
2. *Let C be a class of finite structures. P for $\mathcal{L}$* captures *K on C if the data complexity of P for $\mathcal{L}$ is K and for every property Q of structures from C that is in K there is a sentence ϕ_Q of $\mathcal{L}$ such that $\mathcal{A} \models \phi_Q$ iff $\mathcal{A}$ has property Q, for every $\mathcal{A} \in C$.*

Clearly, the complexity of MX lies between complexities of model checking and satisfiability, since in that case, a part of the input structure is given. E.g. in the case of FO, we avoid undecidability by fixing the universe.

3 Complexity of MX for first-order logic

Complexities of model checking and satisfiability for first-order logic were determined decades ago. The combined complexity of model checking for FO is PSPACE-complete by reduction to QBF [21]. The data complexity of FO AC^0; moreover, FO captures AC^0 over structures with the BIT predicate (or arithmetic structures) [2].

MX for a logic $\mathcal{L}$ is equivalent to model checking for $\exists SO(\mathcal{L})$. That is, there exists an expansion of a structure $\mathcal{A}$ that satisfies a formula ϕ iff $\mathcal{A}$ satisfies ϕ preceded by existential quantifiers for all expansion predicates. Many complexity properties for MX follow from this observation and well-known results on model checking.

Theorem 1. *The combined complexity of MX for FO is NEXP-complete; MX for FO captures NP.*

Proof. The first part is implicit in the proof of expression complexity of ∃SO in [25] (a different proof is presented in [19].) The second part is just Fagin's theorem [6].

It follows that MX for Π_i formulas captures level Σ_{i+1} of the polynomial hierarchy.

Remark 1. In some cases, the only information about the model that is given as an instance for the model expansion is the size of the model (i.e., the instance vocabulary σ is empty). In that case, it is reasonable to give the size of a model as a number in binary notation. This leads to an exponential increase in complexity (since the structure itself is exponential in the size of the input).

Although data complexity of model expansion for full first-order logic is NP-complete, there are fragments of FO for which model expansion is tractable. In particular, the results of [8] translate into the following.

Definition 3. *A* universal Horn formula *is a first-order formula consisting of a conjunction of Horn clauses, preceded by universal first-order quantifiers. Here, a clause is Horn if it contains at most one positive occurrence of an expansion predicate. A* universal Krom formula *is defined similarly, except that the restriction is at most two occurrences of expansion predicates per clause.*

Theorem 2. *The data complexity of the MX problem for universal Horn and Krom formulae is, respectively, P-complete and NL-complete. Moreover, MX for universal Horn and Krom captures P and NL, respectively, over successor structures.*

4 Complexity of MX for guarded fragments of FO

The guarded fragment GF of FO was introduced by Andréka *et al.* [1], and has recently received considerable attention. Here any existentially quantified subformula ϕ must be conjoined with a guard, i.e., an atomic formula over all free variables of ϕ. Gottlob *et al.* [7] extended GF to the k-guarded fragment GF_k, where the conjunction of up to k atoms may act as a guard.

The combined complexity of model checking for GF_k is P-complete [11, 7]. In particular, model checking for GF_k can be done in time $O(ln^k)$, where l is the size of the formula, and n is the size of the structure [17]. The finite satisfiability problem for GF is 2EXP-complete [9].

In this section, we discuss complexity of MX for GF_k: we show that the combined complexity of MX for GF_k, $k \geq 1$, is the same as that for FO, and MX for GF_k, $k \geq 2$, captures NP just as MX for FO does. We also identify a fragment of GF_k, which we denote by RGF_k, such that the combined complexity of MX for RGF_k is NP-complete. Although the data complexity of MX for RGF_k is NP-complete, we show that it does not capture NP. As a corollary of our main results, we show that finite satisfiability for GF_k, $k \geq 2$ is undecidable.

Definition 4. *The* k-guarded fragment *GF_k of FO is the smallest set of formulas that*
1. Contains all atomic formulas;
2. Is closed under Boolean operations;
3. Contains $\exists \bar{x}(G_1 \wedge \ldots \wedge G_m \wedge \phi)$, if the G_i are atomic, $m \leq k$, $\phi \in GF_k$, and every free variable of ϕ appears in some G_i. Here $G_1 \wedge \ldots \wedge G_m$ is called the guard *of ϕ.*

A fragment of GF_k that is of particular interest in application of model expansion is RGF_k, which we use to denote sentences from GF_k in which all guards are given by the instance structure (i.e., no expansion predicates appear in guards). Let FO^k denote FO formulas that use at most k distinct variables. Then it is easy to see that any FO^k formula can be rewritten in linear time into an equivalent one in RGF_k, by using atoms of the form $x = x$ as parts of the guards when necessary. For example, the formula $\exists x \exists y[R(x) \wedge E(x, y)]$ can be rewritten to $\exists x \exists y[R(x) \wedge y = y \wedge E(x, y)]$, where R is an instance predicate, and E is an expansion predicate.

Lemma 1. *There is a polynomial-time algorithm that, given an arbitrary ∃SO sentence, constructs an equivalent ∃SO sentence whose first-order part is in GF_2.*

Proof. Suppose ϕ is a ∃SO sentence $\exists X_1 \ldots \exists X_m \, \varphi$, where φ is FO. Let l be the size of ϕ, and let k be the width of φ, that is, the maximum number of free variables in any subformula of φ. We introduce k new predicates $U_1, \ldots, U_k$ such that the arity of U_i is i, $1 \leq i \leq k$. Let φ' be the formula obtained from φ by replacing any subformula $\exists \bar{x} \, \psi(\bar{x})$ with $\exists \bar{x}(U_i(\bar{x}) \wedge \psi(\bar{x}))$ and any subformula $\forall \bar{x} \, \psi(\bar{x})$ with $\forall \bar{x}(U_i(\bar{x}) \supset \psi(\bar{x}))$, where i is the length of $\bar{x}$. Let η be the formula

$$\bigwedge_{i=0}^{k-1} \forall x_1 \ldots \forall x_{i+1}(x_1 = x_1 \wedge U_i(x_2 \ldots x_{i+1}) \supset U_{i+1}(x_1 \ldots x_{i+1})).$$

It is easy to see that any model of η interprets U_i as the i-ary universal relation, $1 \leq i \leq k$. Now let ϕ' be the ∃SO sentence $\exists X_1 \ldots \exists X_m \exists U_1 \ldots \exists U_k \; (\varphi' \wedge \eta)$. Clearly, $\varphi' \wedge \eta \in \text{GF}_2$, and ϕ' is equivalent to ϕ. Also, both φ' and η are of size $O(l^2)$, and hence ϕ' is of size $O(l^2)$.

Lemma 2. *There exists a polynomial-time algorithm that, given a structure $\mathcal{M}$ and an ∃SO sentence ϕ, constructs a structure $\mathcal{M}'$and an ∃SO sentence $\phi_{\mathcal{M}}$ such that the first-order part of $\phi_{\mathcal{M}}$ is in GF_1, and $\mathcal{M} \models \phi$ iff $\mathcal{M}' \models \phi_{\mathcal{M}}$.*

Proof. Suppose $\mathcal{M}$ is a structure, and ϕ is an ∃SO sentence. Let n be the size of $\mathcal{M}$, and let l be the size of ϕ. For each domain element a of $\mathcal{M}$, we introduce a new constant symbol c_a. Let $\mathcal{M}'$be the structure that expands $\mathcal{M}$ by interpreting c_a as a. Let ϕ' be the ∃SO sentence constructed from ϕ as in the proof of Lemma 1. Now let $\phi_{\mathcal{M}}$ be the sentence obtained from ϕ' by replacing each subformula $\forall x_1 \ldots \forall x_{i+1}(x_1 = x_1 \wedge U_i(x_2 \ldots x_{i+1}) \supset U_{i+1}(x_1 \ldots x_{i+1}))$ with

$$\bigwedge_{a \in dom(M)} \forall x_2 \ldots \forall x_{i+1}(U_i(x_2 \ldots x_{i+1}) \supset U_{i+1}(c_a x_2 \ldots x_{i+1})).$$

Clearly, the first-order part of $\phi_{\mathcal{M}}$ is in GF_1, $\mathcal{M} \models \phi$ iff $\mathcal{M}' \models \phi_{\mathcal{M}}$, and the size of $\phi_{\mathcal{M}}$ is $O(l^2 n)$.

Theorem 3. *1. The combined complexity of MX for GF_k, $k \geq 1$ is NEXP-complete;*
2. MX for GF_k, $k \geq 2$, captures NP;
3. The finite satisfiability problem for GF_k, $k \geq 2$, is undecidable.

Proof. 1. follows from Lemma 2 and the fact that the combined complexity of MX for FO is in NEXP;
2. follows from Lemma 1 and the fact that MX for FO captures NP;
3. By the proof of Lemma 1, finite satisfiability for FO can be reduced to that for GF_2.

Lemma 3 ([19]). *3-SAT can be reduced to MX for a formula $\phi \in RGF_1$.*

Proof. Suppose $\Gamma = \{C_1, \ldots, C_m\}$ is a set of 3-clauses. Let $\mathcal{A}$ be the structure with universe $\{a, \neg a \mid a \in atoms(\Gamma)\}$ such that $\mathcal{A}$ interprets $Clause$ as the set of clauses in Γ and interprets $Complements$ as the set of complementary literals. Let ϕ be

$$\forall x \forall y \forall z(Clause(x, y, z) \supset True(x) \vee True(y) \vee True(z))$$
$$\wedge \forall x \forall y(Complements(x, y) \supset (True(x) \equiv \neg True(y))).$$

Clearly, $\phi \in \text{RGF}_1$, and Γ is satisfiable iff $\mathcal{A}$ can be expanded to a model of ϕ.

We quote the following result concerning polynomial-time grounding of RGF_k sentences:

Lemma 4 ([20]). *There is an algorithm that, given a structure $\mathcal{A}$ and a RGF_k sentence ϕ, constructs in $O(l^2 n^k)$ time a propositional formula ψ such that $\mathcal{A}$ can be expanded to a model of ϕ iff ψ is satisfiable, where l is the size of ϕ, and n the size of $\mathcal{A}$.*

Theorem 4. *(1) The combined complexity of MX for RGF_k is NP-complete. (2) The data complexity of MX for GF_1 and RGF_k is NP-complete. (3) MX for RGF_k and hence also FO^k does not capture NP.*

Proof. (1) follows from Lemmas 4 and 3. (2) follows from Lemma 3 and that the data complexity of MX for FO is in NP. (3): Since SAT can be decided in nondeterministic $O(n^2)$ time, by Lemma 4, MX for RGF_k can be decided in nondeterministic $O(n^{2k})$ time. By Cook's NTIME hierarchy theorem [3], for any $i > 2k$, there is a problem that can be solved in nondeterministic $O(n^i)$ time but not nondeterministic $O(n^{i-1})$ time. Thus there are infinitely many problems in NP that cannot be expressed by MX for RGF_k.

5 Complexity of ID-logic

One limitation of first-order logic as a practical language is its lack of mechanism to describe recursion or induction. Therefore, a natural way to extend first-order logic is by adding inductive definitions. One such approach, called ID-logic, is presented in [4]. ID-logic is an extention of classical logic in which (non-monotone) definitions can appear as atomic formulae. FO(ID) is the extension of FO with such definitions.

Definition 5. *An* inductive definition *Δ is a set of rules of the form $\forall \bar{x}(X(\bar{t}) \leftarrow \phi)$ where X is a predicate symbol of arity r, $\bar{x}$ is a tuple of object variables, $\bar{t}$ a tuple of object variables of length r, ϕ is an arbitrary first-order formula.*

The semantics of the logic is defined by the standard satisfaction recursion of classical logic, augmented with one additional rule saying that a valuation I satifies a definition D if it is the 2-valued well-founded model of this definition. While the well-founded semantics can be defined in several ways, we use a definition where one constructs a sequence $(I^\xi, J^\xi)_{\xi \geq o}$ of approximations (under- and over-estimates) of the intended model of the definition Δ extending I_o, which is a structure providing interpretations to open (i.e., those not appearing in the heads of the rules) symbols of Δ (see [4] for details). Each new element I^ξ is obtained by what we call a "double step" in the proof below, i.e., the square of the so-called stable operator ST_Δ. This operator is in tern is equivalent to a least fixpoint, as will be seen in the proof. The sequence $(I^\xi, J^\xi)_{\xi \geq o}$ has a limit (I, J), where I and J are the least (respectively, the greatest) fixpoints of ST^2_Δ. A good (total) definition is such that $I = J$, otherwise the entire theory does not have a model. Here, we introduce a formula $CONS_\Delta$ which expresses this consistency (totality) criterion, but using least fixpoints only.

Example 2. Consider formula $\Delta_{even} \wedge \forall x(E(x) \vee E(s(x)))$, where

$$\Delta_{even} \equiv \left\{ \begin{array}{c} E(x) \leftarrow x = 0 \\ E(s(s(x))) \leftarrow E(x) \wedge \neg E(s(x)) \end{array} \right\}.$$

This formula states that every number is either even or odd. Definition Δ_{even} is one of possible definitions of even numbers, which is total on natural numbers, but not on integers.

5.1 Equivalence between FO(ID) and FO(LFP)

In this section we show that first-order logic with inductive definitions, FO(ID), is equivalent to first-order logic with least fixed point operator, FO(LFP), just as first-order logic with monotone inductive definitions. This allows us to transfer known complexity results for FO(LFP) to FO(ID).

Lemma 5. *FO(ID) $\subseteq$ FO(LFP).*

Proof. We start by showing how to evaluate a single definition Δ (which can have multiple defined predicates). If a definition is not total on I_0, we need to ensure that there is no model for the whole theory. Then we can use evaluated definitions to construct a $FO(LFP)$ formula corresponding to the original FO(ID) formula.

A definition Δ for a given initialization of open predicates from I_0 is evaluated as follows.

Replace in Δ all occurrences of X_i by X'_i for new variables X'_i. For example, a rule $\forall \bar{x}(X_i(\bar{t}(\bar{x})) \leftarrow \neg X_j(\bar{t}'(\bar{x})))$ becomes replaced with $\forall \bar{x}(X_i(\bar{t}(\bar{x})) \leftarrow \neg X'_j(\bar{t}'(\bar{x})))$ Let ϕ be a formula encoding Δ after this substitution.

Computing one (double) step of the evaluation (a step corresponding to evaluating ϕ with I and then J giving the values for negated literals) becomes

$$\psi \equiv LFP_{\bar{x},\bar{X}}\phi([LFP_{\bar{x},\bar{X}}\phi]^j / X'_j), \qquad (*)$$

by semantics of FO(ID). Here fixpoints are simultaneous on all X_i and the notation $[LFP_{\bar{x},\bar{X}}\phi]^j/X'_j$ means replacing the occurrences of X'_j in ϕ with the fixpoint of X_j in the simultaneous least fixed point of ϕ over all $\bar{X}$.

To simplify the presentation assume, using the fact that simultaneous LFP is equivalent to LFP, that a variable X encodes all variables X_i. Then, the simultaneous LFPs from ψ become just LFPs.

Let Y be a variable encoding the fixpoint of X after the double step $(*)$. This variable is used to initialize X'_i before the next double step. Since after each step ψ the variable Y contains the partial truth assignments on structure I after the i^{th} (double) step of the evaluation procedure, Y is monotone. Therefore, there exists a fixpoint of Y defined by ψ, and it is the least fixed point. Therefore, the formula

$$\Psi_\Delta(\bar{u}) \equiv [LFP_{\bar{y},Y}\psi(Y)]\bar{u}$$

computes the values of the defined predicates in ϕ whenever the fixpoint exists. This is also true when Y is treated as a list of predicates $X_1 \dots X_k$ being defined in Δ, in which case LFP in Ψ_Δ is a simultaneous fixed point.

It is possible, though, that the value computed using the upper bound estimation (the innermost LFP of the double step $(*)$) is different from the outer LFP in the double step. If this is the case, then the following formula is false:

$$CONS_\Delta \equiv \forall\bar{z}([LFP_{\bar{y},Y}\psi(Y)]\bar{z} \leftrightarrow LFP_{\bar{x},\bar{X}}\psi(\bar{x}, LFP_{\bar{y},Y}\psi(Y)/X'_i))[\bar{z}]$$

Suppose now that the FO(ID) theory is defined by a formula with multiple definitions. Let ϕ' be a first-order formula with occurrences of definitions $\Delta_1 \dots \Delta_m$ for some m. To simplify the presentation, view each definition as defining one predicate P_i. If the fixpoint of Δ_i exists, then $\forall\bar{x}P_i(\bar{x}) \leftrightarrow \Psi_{\Delta_i}(\bar{x})$, so occurrences of P_i in ϕ' can be treated as occurrences of Ψ_{Δ_i}. From the point of view of evaluation, it is more efficient to compute $P_i \equiv \Psi_{\Delta_i}$ and then refer just to P_i.

Finally, ϕ' is converted to a formula

$$\Phi \equiv \bigwedge_{i=1}^{m} CONS_{\Delta_i} \wedge \phi'((\forall\bar{x}(P_i(\bar{x}) \leftrightarrow \Psi_{\Delta_i}(\bar{x})))/\Delta_i)$$

That is, Φ is a conjunction of two parts: the conjunction of consistency formulas ensures that all definitions were total, and ϕ' remains the same except all definitions are replaced by the FO(LFP) formulas computing them.

The resulting formula is in FO(LFP).

Example 3. Recall the formula from Example 2 stating that every number is either even or odd. The following describes a construction of an equivalent FO(LFP) formula.

A formula corresponding to Δ_{even} becomes, after replacing $\neg E$ with $\neg E'$,

$$(\phi_E(x, E, E') \equiv (\exists y(x = y \wedge y = 0)) \vee (\exists y(x = s(s(y)) \wedge E(y) \wedge \neg E'(s(y))))).$$

Define $\psi_E(z, E') \equiv [LFP_{x,E}\phi_E(x, E, LFP_{E,x}\phi_E(x, E, E')))]z$. This computes one iteration of the stable operator ST^2_Δ.

Now, $\Psi_\Delta \equiv LFP_{z,E'}\psi_E(z,E')$. Consistency is checked by $\forall u \Psi_\Delta(u) \leftrightarrow [LFP_{x,E}\psi_E(x,E,\Psi_\Delta)]u$. Now, the final formula becomes

$$(\forall u \Psi_\Delta(u) \leftrightarrow [LFP_{x,E}\psi_E(x,E,\Psi_\Delta)]u) \wedge (\forall x(P(x) \leftrightarrow \Psi_\Delta(x)) \wedge (P(x) \vee O(s(X))).$$

Here, the first conjunct checks that the definition "makes sense", otherwise the formula does not have a model, the second part defines a particular variable $P(x)$ to represent the defined E, and the last part uses P outside of the definition Δ_E.

Lemma 6. *FO(LFP) $\subseteq$ FO(ID)*

Proof. By Theorem 9.4.2 of [5], every FO(LFP) formula is equivalent to one of the form $\forall u[LFP_{\bar{z},Z}\psi]\tilde{u}$, where $\psi \in \Delta_2$, which can be written as an FO(ID) formula $\{Z(\bar{z}) \leftarrow \psi\}$.

Therefore, the following holds:

Theorem 5. *The complexity of model checking for ID-logic and FO(LFP) coincide over finite structures.*

Corollary 1. *The combined complexity of the model checking for FO(ID) is complete for EXP; the expression complexity for FO(ID) is complete for P; $FO(ID)$ captures P over successor structures.*

5.2 Complexity of MX for FO(ID)

Intuitively, adding polynomial-time computable predicates to an NP predicate should not add expressive power, suggesting that both combined and data complexities of MX for FO(ID) (or, equivalently, FO(LFP)) should coincide with the those for FO without inductive definitions or least fixed points.

Theorem 6. *The combined complexity of MX for FO(ID) is NEXP-complete; The data complexity for MX of FO(ID) is NP-complete, and NP is captured by existential second-order with inductive definitions $\exists$SO(ID).*

Proof. We know from Theorem 1 that data complexity for MX problem is hard for NP and combined complexity hard for NEXP. Therefore, it is sufficient to show membership of MX in these classes.

The evaluation algorithm proceeds as follows. Use non-determinism to guess the expansion predicates. Now the problem is reduced to evaluating the FO(ID) formula on an expanded structure. This can be done in polynomial time of the size of the structure when formula is fixed (by [13, 25, 18]) and in exponential time when the formula is a part of the input by [25]. In the second case, the size of the expansion predicates can be exponential in the size of the structure (since their arity is not constant), but in NEXP we can guess exponential-size certificates.

Fragments of FO(ID) with polytime MX Recall that MX for universal Horn formulae was P-complete. We would like to add inductive definitions to such formulae so that the complexity of the resulting logic is still in P. The following example shows that allowing unrestricted use of expansion predicates in the inductive definitions makes it possible to encode NP-complete problems

Example 4. The classical example of 3-colourability is representable as a formula with three expansion predicates R, B, G, encoding colours:

$$\forall v, w(R(v) \vee B(v) \vee G(v)) \wedge \bigwedge_{Q \in R,G,B} (\neg Q(v) \vee \neg Q(w) \vee \neg E(v, w)).$$

The only part of this formula which is not Horn is the first disjunction. It can be replaced by the inductive definition with a rule $X(i) \leftarrow Q(i)$ for every colour Q. Now, the first disjunction is equivalent to $\forall v X(v)$. Note that the definition of FO(ID) requires that such X be minimal, therefore, this does not introduce spurious positives.

However, if we disallow any occurrences of the expansion predicates in inductive definitions, P-completeness is preserved.

Lemma 7. *Adding inductive definitions to universal Horn formulae defined on page 4 preserves data complexity of MX to be P-complete, when expansion predicates do not occur in inductive definitions.*

Proof. By theorem 2, data complexity of MX for universal Horn formulae is P-complete. Therefore, a polytime algorithm for MX of universal Horn formulae can first evaluate all inductive definitions, and then run Grädel's algorithm for evaluating existential second-order Horn formulae replacing all defined predicates by their computed values.

We can also add expansion predicates in a restricted fashion. First, all expansion predicates occurring in definitions have to be defined (i.e, occur in a head of a rule of some definition). Second, such predicates cannot be defined in terms of each other unless they are in the same definition. Third, the definitions can only occur as conjunction to the rest of the formula. Intuitively, in this case, if expansion predicates in the body of a definition are either given values already, or are being defined in that definition, then the definition can be evaluated. The intuition here is similar to the intuition of RGF_k.

Definition 6. *Let $\{\bar{X}_1, \ldots, \bar{X}_k\}$ be all expansion predicates occurring in a first-order formula ϕ. Then ϕ is in* RFO(ID) *if (1) for each $\bar{X}_i$ there is a definition Δ_i defining all predicates in $\bar{X}_i$, and Δ_i is conjuncted with the rest of the formula. (2) The only expansion predicates allowed in the body of Δ_i are among $\bar{X}_1, \ldots, \bar{X}_{i-1}$; the body of Δ_1 contains no expansion predicates.*
More generally, ϕ is in RuHorn(ID) if there are also expansion predicates $\bar{P}$ which do not occur in the definitions and with all definitions removed, ϕ is universal Horn with respect to $\bar{P}$

Theorem 7. *MX for RFO(ID) is P-complete.*

Corollary 2. *MX for RuHorn(ID) is P-complete.*

6 Conclusion

We presented complexity results for model expansion for a number of logics closely related to FO. For providing a foundation for practical constraint languages, future work needs to consider similar problems in the presence of arithmetic and other interpreted function symbols. For some steps in this direction, see [23, 22]

We know that GF(ID) (guarded logic with inductive definitions) coincides with μGF on total structures; however, the question is still open whether they coincide everywhere, as FO(ID) and FO(LFP) do. The problem lies in a different treatment of inductive definitions that are not total.

Acknowledgments

The authors are grateful to the Natural Sciences and Engineering Council of Canada and to the Pacific Institute for Mathematical Sciences for financial support.

References

1. Andréka, H., van Benthem, J., Németi, I.: Modal languages and bounded fragments of predicate logic. Journal of Philosophical Logic 49(3), 217–274 (1998)
2. Barrington, D.M., Immerman, N., Straubing, H.: On uniformity within NC^1. Journal of Computer and System Sciences 41(3), 274 – 306 (1990)
3. Cook, S.A.: A hierarchy for nondeterministic time complexity. Journal of Computer and System Sciences 7(4), 343–353 (1973)
4. Denecker, M., Ternovska, E.: A logic of non-monotone inductive definitions. ACM transactions on computational logic (TOCL) 9(2), 1–52 (2008)
5. Ebbinghaus, H.D., Flum, J.: Finite model theory. Springer Verlag (1995)
6. Fagin, R.: Generalized first-order spectra and polynomial-time recognizable sets. In: Complexity of computation, SIAM-AMC. vol. 7, pp. 43–73 (1974)
7. Gottlob, G., Leone, N., Scarcello, F.: Robbers, marshals, and guards: game theoretic and logical characterizations of hypertree width. In: Twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS '01). pp. 195–206 (2001)
8. Grädel, E.: Capturing Complexity Classes by Fragments of Second Order Logic. Theoretical Computer Science 101, 35–57 (1992)
9. Grädel, E.: On the restraining power of guards. Journal of Symbolic Logic 64, 1719–1742 (1999)
10. Grädel, E., Kolaitis, P.G., Vardi, M.Y.: On the decision problem for two-variable first-order logic. Bulletin of Symbolic Logic 3, 53–69 (1997)
11. Grädel, E., Otto, M.: On logics with two variables. Theoretical Computer Science 224, 73–113 (1999)
12. Grädel, E., Walukiewicz, I.: Guarded fixed point logic. In: Fourteenth Annual IEEE Symposium on Logic in Computer Science (LICS '99). pp. 45–55 (1999)
13. Immerman, N.: Relational queries computable in polytime. In: Fourteenth Annual ACM Symposium on Theory of Computing (STOC'82). pp. 147 –152 (1982)
14. Immerman, N.: Descriptive complexity. Springer Verlag, New York (1999)
15. Jurdzinski, M.: Deciding the winner in parity games is in UP $\cap$ co-UP. Information Processing Letters 69, 119–124 (1998)
16. Libkin, L.: Elements of Finite Model Theory. Springer Verlag (2004)

17. Liu, Y., Levesque, H.J.: A tractability result for reasoning with incomplete first-order knowledge bases. In: 18th Int. Joint Conf. on Artif. Intell. (IJCAI '03). pp. 83–88 (2003)
18. Livchak, A.: Languages for polynomial-time queries. In: Computer-based modeling and optimization of heat-power and electrochemical objects. p. 41 (1982)
19. Mitchell, D., Ternovska, E.: A framework for representing and solving NP search problems. In: 20th National Conf. on Artif. Intell. (AAAI). pp. 430–435 (2005)
20. Patterson, M., Liu, Y., Ternovska, E., Gupta, A.: Grounding for model expansion in k-guarded formulas with inductive definitions. In: 22nd International Joint Conference on Artificial Intelligence (IJCAI'07) (2007)
21. Stockmeyer, L.: The Complexity of Decision Problems in Automata Theory. Ph.D. thesis, MIT (1974)
22. Tasharrofi, S., Ternovska, E.: Capturing NP for search problems with built-in arithmetic. In: this volume (2010)
23. Ternovska, E., Mitchell, D.: Declarative programming of search problems with built-in arithmetic. In: 21st International Joint Conference on Artificial Intelligence (IJCAI'09) (2009)
24. Trahtenbrot, B.: The impossibility of an algorithm for the decision problem for finite domains (In Russian). Doklady Academii Nauk SSSR 70, 569–572 (1950)
25. Vardi, M.Y.: The complexity of relational query languages. In: Fourteenth Annual ACM Symposium on Theory of Computing (STOC'82). pp. 137–146 (1982)
26. Vardi, M.Y.: On the complexity of bounded-variable queries. In: Fourteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS '95). pp. 266–276 (1995)