CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Background

One branch of the study of *descriptive complexity* aims at characterizing complexity classes according to the logical resources needed to describe problems in each complexity class. It is a very remarkable fact that many natural complexity classes have nice logical characterizations.

The study of descriptive complexity was initiated by Fagin (1974), who proved that NP consists of exactly those problems that are definable by existential second-order sentences. Later, similar characterizations have been proved for many basic complexity classes, including PTIME and PSPACE. The results are:

- On the class of finite structures with a built-in linear order, FO + IFP captures PTIME. This was proved independently by Immerman (1986) and Vardi (1982).
- On the class of finite structures with a built-in linear order, FO + PFP captures PSPACE. Vardi (1982) first proved that the database language *while* captures PSPACE in the presence of order. Later, Abiteboul and Vianu (1991b) proved the equivalence of *while* and the logic FO + PFP.

One of the most intriguing open problems in descriptive complexity is whether there exists a logic (in a sense will be formalized later) which captures PTIME on the class of all finite structures (both ordered and unordered).

Gurevich (1988) formalized the notion of a logic capturing PTIME and conjectured a negative answer to the above question. We saw above that in the presence of a linear order, FO + IFP captures PTIME. It can also be seen that this built-in order is necessary. On arbitrary finite structures, FO + IFP cannot express such queries as evenness.

Immerman suggested a new logic FO + IFP + C extending FO + IFP with a counting construct (Immerman (1986)). But even this logic fails to capture PTIME, as has been proved by Cai et al. (1992).

Abiteboul and Vianu defined another extension of FO + IFP, called FO + IFP + W, which has a nondeterministic choice operator W, called the *witness* operator (Abiteboul and Vianu (1991a)). This operator chooses an arbitrary element from a set given as argument. This logic is a nondeterministic logic in that each formula defines several different relations on a given structure. It was shown that the *deterministic fragment* of FO + IFP + W, i.e., those formulas of FO + IFP + W which define only deterministic queries, captures PTIME. But this is an undecidable fragment and so this still does not give us a logic which captures PTIME.

A restricted form of nondeterminism called *semi-determinism* was considered in the context of query database languages by Gyssens et al. (1994). This is based on a *semi-deterministic choice operator* which chooses an arbitrary element from a set given as argument, provided the set is an automorphism class.

Recently, Gire and Hoang (1998) explored several semi-deterministic choice extensions of FO + IFP and their relation with the class PTIME and the logic FO + IFP + C. In this thesis, we look at some of the results from that paper and some related results.

1.2 Preliminaries

We start with defining some basic notions of logic.

A *first order vocabulary* is a finite tuple $\sigma = \langle R_1, \ldots, R_n \rangle$ of *relation symbols* R_i each with an associated *arity* a_i.

A *structure* over σ is a tuple $\mathfrak{A} = \langle \mathrm{A}, R_1^{\mathrm{A}}, \ldots, R_n^{\mathrm{A}} \rangle$ where A is a set called the *universe* or *domain* of $\mathfrak{A}$ (sometimes denoted $|\mathfrak{A}|$) and for each i, R_i^{A} is an a_i-ary relation on A, i.e., $R_i^{\mathrm{A}} \subseteq \mathrm{A}^{a_i}$.

A structure $\mathfrak{A}$ is called *finite* if A is finite. In this thesis, we will be dealing exclusively with finite structures. We denote the class of all finite structures over σ by $\mathrm{fin}(\sigma)$.

Let $\mathfrak{A}$ and $\mathfrak{B}$ be two structures over σ. By an *isomorphism* between $\mathfrak{A}$ and $\mathfrak{B}$, we mean a bijection $\pi : \mathrm{A} \longrightarrow \mathrm{B}$ such that for all i and for all a_i-tuples $\bar{x} \in \mathrm{A}^{a_i}$, $R_i^{\mathrm{A}}(\bar{x})$ iff $R_i^{\mathrm{B}}(\pi(\bar{x}))$. We denote this fact by $\pi : \mathfrak{A} \cong \mathfrak{B}$. We say that $\mathfrak{A}$ is *isomorphic* to $\mathfrak{B}$ if there is an isomorphism between $\mathfrak{A}$ and $\mathfrak{B}$.

An *m-ary query* over σ is a partial recursive map $q : \mathrm{fin}(\sigma) \longrightarrow \mathrm{fin}(\tau)$ where $\tau = \langle R \rangle$ is a vocabulary with a single m-ary relation symbol R, which satisfies the following conditions:

- for each $\mathfrak{A} \in \mathrm{fin}(\sigma)$, $R^{q(\mathfrak{A})} \subseteq \mathrm{A}^m$.
- whenever $\pi : \mathfrak{A} \cong \mathfrak{B}$, we also have $\pi : q(\mathfrak{A}) \cong q(\mathfrak{B})$.

A 0-ary query over σ, also called a *Boolean query*, is a partial recursive map $q : \mathrm{fin}(\sigma) \longrightarrow \{0, 1\}$ such that $\mathfrak{A} \cong \mathfrak{B}$ implies $q(\mathfrak{A}) = q(\mathfrak{B})$. Such a query can be identified with a subclass of $\mathrm{fin}(\sigma)$, consisting of those structures $\mathfrak{A}$ for which $q(\mathfrak{A}) = 1$.

Any first order formula with m free variables defines in a natural way an m-ary query and a first order sentence defines a Boolean query.

In this thesis we also consider nondeterministic queries.

A *nondeterministic m-ary query* over σ is a recursively enumerable subset q of $\mathrm{fin}(\sigma) \times \mathrm{fin}(\tau)$ where $\tau = \langle R \rangle$ where R is an m-ary relation symbol, which satisfies the following conditions:

- for each $\mathfrak{A} \in \mathrm{fin}(\sigma)$, if $(\mathfrak{A}, \mathfrak{B}) \in q$, then $R^{\mathfrak{B}} \subseteq \mathrm{A}^m$.
- whenever $(\mathfrak{A}, \mathfrak{B}) \in q$, then for all isomorphisms π from $\mathfrak{A}$ to other structures we have $(\pi(\mathfrak{A}), \pi(\mathfrak{B})) \in q$.

A *Boolean nondeterministic query* is a recursively enumerable subset of $\mathrm{fin}(\sigma) \times \{0, 1\}$ which satisfies the following condition:

- For any structure $\mathfrak{A} \in \mathrm{fin}(\sigma)$, let $q(\mathfrak{A}) = \{a \in \{0, 1\} \mid (\mathfrak{A}, a) \in q\}$. Then whenever $\mathfrak{A} \cong \mathfrak{B}$, $q(\mathfrak{A}) = q(\mathfrak{B})$.

We assume familiarity with the syntax and semantics of first order logic. FO denotes first order logic. In general a logic L (such as FO or FO + IFP) denotes both a set of formulas as well as the class of queries definable in that logic.

For investigations in descriptive complexity theory, FO is considered a very weak logic. The reason is its low expressive power. Two typical examples of queries not expressible in FO are

the evenness query determine whether the input structure has an even number of elements.

the connectedness query determine whether the input graph is connected.

These examples suggest that FO lacks two very important things, the ability to count, and a recursion mechanism. Naturally we look at extensions of FO. FO + IFP is an interesting extension of FO which has the power of recursion. We define it next.

Definition 1.1 The logic FO + IFP.

Syntax

The set of formulas of the logic FO + IFP is defined by simultaneous induction over all vocabularies σ. The set of formulas of the logic FO + IFP over a vocabulary σ is the least set $\mathit{Form}(\sigma)$ satisfying:

- if φ is an atomic formula over σ then $\varphi \in Form(\sigma)$
- if $\varphi \in Form(\sigma)$ then $\neg\varphi \in Form(\sigma)$
- if $\varphi, \psi \in Form(\sigma)$ then $\varphi \vee \psi \in Form(\sigma)$
- if $\varphi \in Form(\sigma)$ then $\forall x\varphi \in Form(\sigma)$
- Let $\varphi(\bar{x}, S) \in Form(\sigma \cup \{S\})$, where S is a relation symbol not in σ such that the length of $\bar{x}$ equals the arity of S, and let $\bar{t}$ be a tuple of terms whose length is equal to the length of $\bar{x}$. Then

$$(\mathbf{ifp}_{S,\bar{x}}\ \varphi)(\bar{t}) \in Form(\sigma)$$

Semantics

For each σ-structure $\mathfrak{A}$, and $\sigma \cup \{S\}$ formula φ as given above, we can define a sequence of relations as follows:

$$S^0 = \emptyset,$$

$$\textit{and for each } i \geq 0,\ S^{i+1} = S^i \cup \{\bar{a} \mid \mathfrak{A} \models \varphi(\bar{a}, S^i)\}.$$

Define $S^\infty = \bigcup_{i=0}^{\infty} S^i$. We say $\mathfrak{A} \models (\mathbf{ifp}_{S,\bar{x}}\ \varphi)(\bar{t})$ if and only if the tuple $\bar{t}^{\mathrm{A}} \in S^\infty$. □

One nice property of the **ifp** operator is that it is an inductive operator which keeps on adding to the relation S at each stage, so the fixed point is reached in a polynomial number of iterations. Thus any FO + IFP formula can be evaluated in PTIME. We give below an example of the use of the **ifp** operator.

Example 1.2 The following FO + IFP-sentence says that the input graph is connected:

$$\varphi\ \equiv \forall s \forall t(\mathbf{ifp}_{x,X}\ x = s\ \vee\ \exists y(Xy\ \wedge\ Eyx))(t).$$ □

FO + IFP derives its importance from the following celebrated result by Immerman (1986) and Vardi (1982) which says that *on the class of ordered structures*, FO + IFP = PTIME.

On the other hand, on the class of all finite structures (both ordered and unordered), FO + IFP is far weaker than PTIME. For instance, it is known that the evenness query which we encountered earlier is not expressible in FO + IFP.

We now present the logic FO + IFP + C, which was introduced by Immerman (1986).

Definition 1.3 The logic FO + IFP + C

We associate with each structure $\mathfrak{A}$ a two-sorted structure $\mathfrak{A}^+ = \mathfrak{A} \uplus (\{0, \ldots, |\mathrm{A}|\}, \leq)$. That is, the second sort is to be considered as a set of numbers from 0 to the cardinality of the universe equipped with the usual ordering $\leq$ on natural numbers. Members of the first sort are called *points*, those of the second sort *numbers*.

Syntax

There are two sorts of variables, point variables and number variables. We usually use x, y, z for point variables and μ, ν for number variables. There are also two sorts of quantifiers. Further we also allow mixed relations, i.e., relations in which some arguments are numbers and some others are points.

The formula-formation rules are similar to the rules for FO + IFP, except for the following addition:

if $\varphi(\bar{x}, \bar{y}, \bar{\nu})$ is an FO + IFP + C formula, $\bar{x}$ and $\bar{y}$ are tuples of point variables, and $\bar{\mu}$ and $\bar{\nu}$ are tuples of number variables such that $|\bar{x}| = |\bar{\mu}|$ ($\bar{y}$ and $\bar{\nu}$ are to be treated as parameters), then

$$\mathbf{Count}(\bar{x}, \varphi(\bar{x}, \bar{y}, \bar{\nu})) = \bar{\mu}$$

is an FO + IFP + C formula.

Semantics

The formulas of FO + IFP + C are interpreted over the two-sorted counterpart $\mathfrak{A}^+$ of $\mathfrak{A}$. Tuples of number variables $\bar{\mu}$ are evaluated as follows: Let $\bar{m} = m_k m_{k-1} \ldots m_0$ be an interpretation for $\bar{\mu}$, where for $0 \leq i \leq k-1$, $m_i \in \{0, \ldots, |\mathrm{A}| - 1\}$ and $m_k \in \{0, \ldots, |\mathrm{A}|\}$. Then $val(\bar{m}) = \sum_{i=0}^{k} m_i . |\mathrm{A}|^i$.

We now specify the semantics of the **Count** construct. Let $\bar{m}$ and $\bar{n}$ be interpretations of the number variables $\bar{\mu}$ and $\bar{\nu}$ respectively and let $\bar{b}$ be an interpretation of the point variables $\bar{y}$. Then

$$\mathfrak{A}^+ \models \mathbf{Count}(\bar{x}, \varphi(\bar{x}, \bar{y}, \bar{\nu})) = \bar{\mu}[\bar{b}, \bar{n}, \bar{m}] \ \textit{iff}$$
$$|\{\bar{a} \in A \mid \mathfrak{A}^+ \models \varphi(\bar{a}, \bar{b}, \bar{n})\}| = val(\bar{m}).$$

For formulas φ which do not have free number-variables, we define $\mathfrak{A} \models \varphi$ if and only if $\mathfrak{A}^+ \models \varphi$. □

We will only consider those FO + IFP + C formulas whose free variables range only over points. For convenience we will directly use formulas of the form $\bar{\mu} \leq \bar{\nu}$. If $\bar{m}$ and $\bar{n}$ interpret $\bar{\mu}$ and $\bar{\nu}$ then the above formula is interpreted as $val(\bar{m}) \leq val(\bar{n})$. Of course these formulas are easily FO + IFP-definable using the atomic formulas.

Example 1.4 1. The following FO + IFP + C-sentence asserts that the input structure has an even cardinality.

$$\varphi \equiv \exists\mu((\mathbf{ifp}_{\mu,X}\ \mu = 0 \ \vee \ \exists\nu \in X(\mu = \nu + 2))(\mu) \ \wedge \ \forall\nu(\nu \leq \mu)).$$

2.The following FO + IFP + C formula says that a vertex x of a graph has even degree.

$$\varphi(x) \equiv \exists\mu((\mathbf{ifp}_{\mu,X}\ \mu = 0 \ \vee \ \exists\nu \in X(\mu = \nu + 2))(\mu) \ \wedge \ \mathbf{Count}(y, Exy) = \mu).$$

Note that we crucially use both the **Count** construct and the order(and hence arithmetic) available on the number sort to achieve more expressive power. □

FO + IFP + C is a strictly more expressive logic than FO + IFP, but it still does not give us all of PTIME, as has been shown by Cai et al. (1992). In their paper, they construct a class of graphs and give a PTIME query on this class not expressible in FO + IFP + C. We will not go into their proof in this thesis, but we will use their construction in one of our results.

Now is a good time to make precise the notion of a logic capturing PTIME, as given by Gurevich (1988).

Definition 1.5 A *logic* L consists of a mapping that assigns a recursive set $L(\sigma)$ of sentences for each vocabulary σ, and a *satisfaction relation* $\models_L$ between sentences and structures such that for all $\varphi \in L(\sigma)$ the class $\mathrm{Mod}(\varphi) = \{\mathfrak{A} \in \mathrm{fin}(\sigma) \mid \mathfrak{A} \models_L \varphi\}$ is an isomorphism closed class of σ-structures.

A class $\mathcal{C}$ of σ-structures is *definable* in L if there is a $L(\sigma)$-sentence φ such that $\mathcal{C} = \mathrm{Mod}(\varphi)$. □

Definition 1.6 A logic L *captures* PTIME iff the following two conditions hold:

(i) Each PTIME-computable class is definable in L.

(ii) For each vocabulary σ, there is a Turing machine M which, given any $L(\sigma)$ formula φ as input, outputs another Turing machine M_φ and a polynomial P such that M_φ computes the query $q(\mathfrak{A}) = \varphi^{\mathrm{A}}$ on σ-structures in time bounded by $P(|A|)$. □

Gurevich also conjectured that there exists no logic which captures PTIME in this sense.

In the next chapter we will see some nondeterministic extensions of FO + IFP which seek to capture PTIME.

CHAPTER 2

SEMI-DETERMINISTIC CHOICE LOGICS

2.1 A nondeterministic logic

In this chapter we look at several choice-based extensions of FO + IFP studied by Gire and Hoang (1998). First, we introduce the logic FO + c-IFP. This is an extension of FO + IFP with a construct called the *inflationary choice fixpoint* operator.

Definition 2.7 The logic FO + c-IFP

Syntax

The set of formulas of the logic FO + c-IFP over a vocabulary σ is the least set $Form(\sigma)$ satisfying:

- if φ is an atomic formula over σ then $\varphi \in Form(\sigma)$
- if $\varphi \in Form(\sigma)$ then $\neg\varphi \in Form(\sigma)$
- if $\varphi, \psi \in Form(\sigma)$ then $\varphi \vee \psi \in Form(\sigma)$
- if $\varphi \in Form(\sigma)$ then $\forall x\varphi \in Form(\sigma)$
- Let $\psi(\bar{x}, S, T), \varphi(\bar{y}, S, T) \in Form(\sigma \cup \{S, T\})$, where S, T are two relation symbols not in σ such that the length of $\bar{x}$ equals the arity of S and the length of $\bar{y}$ equals the arity of T, and let $\bar{t}$ be a tuple of terms whose length is equal to the length of $\bar{x}$. Then

$$(\textbf{c-ifp}_{S\bar{x},T\bar{y}}\ \psi,\ \varphi)(\bar{t}) \in Form(\sigma)$$

Semantics

We will use the operation **choice** which has the following semantics:
for any set X given as argument, each invocation of **choice**(X) returns an arbitrary element of X. For each σ-structure $\mathfrak{A}$, and two formulas ψ and φ as given above, we can define a sequence of relations as follows:

$$S^0 = \emptyset; \quad T^0 = \emptyset;$$

and for each $i \geq 0$, the *choice step* defines

$$T^{i+1} = \{\bar{b}'\} \textit{ where } \textbf{choice}(\{\bar{b} \mid \mathfrak{A} \models \varphi(\bar{b}, S^i, T^i)\}) \textit{ returns } \bar{b}'$$

and the *induction step* defines

$$S^{i+1} = S^i \cup \{\bar{a} \mid \mathfrak{A} \models \psi(\bar{a}, S^i, T^{i+1})\}.$$

Define $S^\infty = \bigcup_{i=0}^{\infty} S^i$. We say $\mathfrak{A} \models (\textbf{c-ifp}_{S\bar{x},T\bar{y}}\ \psi,\ \varphi)(\bar{t})$ if and only if the tuple $\bar{t}^{\mathrm{A}} \in S^\infty$. □

Remark 2.8 1. Note that the sequence $\{S^i\}$ is monotone but the sequence $\{T^i\}$ is not monotone, containing just a single element at each stage. But the semantics ensures that still the fixed point is reached in a polynomial number of iterations.

2. Note that $\mathrm{FO} + \mathrm{IFP} \subseteq \mathrm{FO} + \text{c-IFP}$. That is, for every formula $\psi \in \mathrm{FO} + \mathrm{IFP}$, there is an equivalent formula $\theta \in \mathrm{FO} + \text{c-IFP}$. Let $\psi(\bar{x})$ be an $\mathrm{FO} + \mathrm{IFP}$ formula with free variables $\bar{x}$. Let S be a new relation symbol whose arity is the same as $|\bar{x}|$, and let T be a new unary relation symbol. Then $\theta \equiv (\textbf{c-ifp}_{S\bar{x},Ty}\ \psi(\bar{x}), y \neq y)(\bar{x})$ is an $\mathrm{FO} + \text{c-IFP}$ formula equivalent to ψ. T is always $\emptyset$. In the very first iteration, S reaches the fixed point. □

We give an example of the use of the **c-ifp** operator below.

Example 2.9 In this example we assume that we have a constant *min* available in the vocabulary. This is just to simplify our formulas. We can easily

find equivalent formulas which do not use constants. We show how to define a linear order on any σ-structure using the **c-ifp** operator. The idea is this:

In the first iteration, pick the element denoted by *min* and make it the minimum element of the order. In subsequent iterations, arbitrarily pick an element from the currently unordered elements and make it the successor to the currently largest element in the order. This strategy is formalized by the formula θ given below:

$$\theta(x,y) \equiv (\textbf{c-ifp}_{Sxy,Ty}\psi, \varphi)(x,y)$$
where
$$\psi(x,y) \equiv [(x = min) \wedge Ty \wedge \neg\exists v(S\ min\ v)] \vee [Ty \wedge \exists u(Sux) \wedge \neg\exists v(Sxv)]$$
and
$$\varphi(y) \equiv \neg\exists u(Suy) \wedge (y \neq min).$$
where $S, T \notin \sigma$.

We can avoid the use of constants by picking the first and second elements of the ordering in the first iteration and picking each subsequent element in the remaining stages. □

From Example 2.9 and Remark 2.8, it follows that any FO + IFP formula using a built-in order can be simulated by an FO + c-IFP formula which defines an order and uses it in the simulation of the FO + IFP formula. Thus FO + c-IFP expresses all PTIME queries.

A new notion has been introduced in this logic: nondeterminism. The **choice** operation returns an arbitrary element and different invocations of **choice** on the same set are independent of each other, so in general a formula using the **c-ifp** operator defines more than one relation on each input structure. One can look at the situation as different "runs" of the formula defining different relations. For instance, the formula in Example 2.9 generates all possible linear orderings of the input structure. (There are exponentially many of them.)

The logic FO + IFP + W, which has a nondeterministic *witness* operator was earlier introduced by Abiteboul and Vianu (1991a). It can be

shown that for each FO + IFP + W formula we can *effectively* construct an equivalent FO + c-IFP formula and vice versa. Therefore results on FO + IFP + W (see Abiteboul and Vianu (1991a)) carry over to FO + c-IFP. In particular, the deterministic fragment of FO + c-IFP captures PTIME but this is not a decidable fragment. So this does not give us a logic capturing PTIME.

2.2 Symmetric Choice fixpoint logic

A natural restriction of nondeterministic queries which has been studied is the class of *semi-deterministic* queries. These are queries in which the different outputs on the same input structure are isomorphic to each other. That is, a query q is semi-deterministic iff for all structures $\mathfrak{A}$, $(\mathfrak{A}, \mathfrak{B}) \in q$ and $(\mathfrak{A}, \mathfrak{C}) \in q \Rightarrow \mathfrak{B} \cong \mathfrak{C}$. Another term used for semi-determinism is *sound nondeterminism.*

We next define the logic FO + sc-IFP whose formulas define only semi-deterministic queries. It uses the so called *symmetric choice fixpoint* operator. Before that we need to introduce the notion of *automorphism class.*

Definition 2.10 Let $\mathfrak{A}$ be a structure. For all k and k-tuples $(x_1 \ldots x_k)$ and $(y_1 \ldots y_k)$ of A, we say $(x_1 \ldots x_k) \cong (y_1 \ldots y_k)$ iff there is an automorphism f of $\mathfrak{A}$ such that $f(x_i) = y_i$ for $1 \leq i \leq k$. The equivalence classes of the relation $\cong$ are called the *automorphism classes* of $\mathfrak{A}$. □

The elements of an automorphism class are called *symmetric elements.* This explains the reason why all the logics which follow are called *symmetric choice logics.*

Definition 2.11 The logic FO + sc-IFP

Syntax

The set of formulas of the logic FO + sc-IFP over a vocabulary σ is the least set $Form(\sigma)$ satisfying:

- Let $\psi(\bar{x}, S, T), \varphi(\bar{y}, S, T) \in \mathrm{FO} + \mathrm{IFP}(\sigma \cup \{S, T\})$, where S, T are two relation symbols not in σ such that the length of $\bar{x}$ equals the arity of S and the length of $\bar{y}$ equals the arity of T, and let $\bar{t}$ be a tuple of terms whose length is equal to the length of $\bar{x}$. Then

$$(\textbf{sc-ifp}_{S\bar{x},T\bar{y}}\ \psi,\ \varphi)(\bar{t}) \in Form(\sigma).$$

Semantics

The semantics of the **sc-ifp** operator is the same as for the **c-ifp** operator except for the following change:

in each choice step i, if the choice set is an automorphism class of $(\mathfrak{A}, S^i, T^i)$ then $T^{i+1} = \{\bar{b}\}$ where $\bar{b}$ is the element returned by the **choice** operation. If the choice set is not an automorphism class, T^{i+1} is defined to be $\emptyset$. □

Remark 2.12 1. It can be shown that any FO + sc-IFP formula $(\textbf{sc-ifp}_{S\bar{x},T\bar{y}}\ \psi,\ \varphi)(\bar{t})$ is equivalent to a formula of the form $(\textbf{sc-ifp}_{S'\bar{x},T'\bar{y}}\ \psi',\ \varphi')(\bar{t}')$ where ψ' and φ' are first order formulas. We will briefly mention the main ideas in the proof.

By the normal form theorem for FO + IFP (Immerman (1986), Gurevich and Shelah (1986)) we can assume that $\psi \equiv (\textbf{ifp}_{S_0\bar{x}}\psi_0)$ and $\varphi \equiv (\textbf{ifp}_{T_0\bar{x}}\varphi_0)$ where ψ_0 and φ_0 are first order. The basic idea now is to compute both S_0 and T_0 in S'. This is a standard trick which can be found in many places in the literature, Section 7.2 of Ebbinghaus and Flum (1995) for instance. In the present case, we have to circumvent one problem. By definition, T is not a monotone relation. The choice set also is not increasing from stage to stage, rather the choice sets of two different stages do not have any relation with each other. But since S' grows monotonically, the elements of all the choice sets till the present stage will be in S'. We propose the following solution. Assume that all the elements of the choice sets have a "tag"

either "0" or "1". We distinguish between the elements of the current choice set and the elements of the previous choice sets as follows. Each element of the previous choice sets appear twice in S', once with tag "0" and once with tag "1". The elements of the current choice set appear only once in S', with tag "0". Now each time we evaluate φ_0 we insert all its elements in S' with tag "0". Once φ_0 reaches the fixed point, apply the **choice** operation to the current choice set. After making the choice, again insert the elements of the current choice set in S', with tag "1". The other details are standard.

2. Exactly the same argument as given in Remark 2.8 will show that FO + IFP $\subseteq$ FO + sc-IFP. $\square$

We give an example of the expressiveness of this logic.

Example 2.13 In this example we show that on structures with just one unary relation R defined on them, we can define a linear order by an FO + sc-IFP formula. The idea is simple: Let all elements satisfying $\neg R$ precede all elements satisfying R. Within these two sets define a linear order following exactly the same strategy used in Example 2.9 using the fact that at each stage the set of unordered elements satisfying $R(\neg R)$ is an automorphism class. This is the essential idea. We give formulas which combine both the ordering of the individual sets and the ordering between $\neg R$ and R. That makes the formulas a little bit complicated. Again we use constants min_1, min_2 for simplifying the formulas but they can be eliminated.

$$\theta(x,y) \equiv (\textbf{sc-ifp}_{Sxy,Ty}\psi, \varphi)(x,y)$$

where

$$\begin{aligned}\psi(x,y) \equiv\ & \{\exists v(\neg Rv \wedge \neg\exists u(Suv)) \Rightarrow \\ & \quad [(x = min_1) \wedge Ty \wedge \neg\exists v(S\ min_1\ v)] \vee \\ & \quad [Ty \wedge \exists u(Sux) \wedge \neg\exists v(Sxv)]\} \\ & \wedge \{\forall v(\neg Rv \Rightarrow \exists u(Suv)) \Rightarrow \\ & \quad [(x = min_2) \wedge Ty \wedge \neg\exists v(S\ min_2\ v)] \vee \\ & \quad [Ty \wedge \exists u(Sux) \wedge \neg\exists v(Sxv)]\}\end{aligned}$$

and

$$\varphi(y) \equiv [\exists v(\neg Rv \wedge \neg\exists u(Suv)) \Rightarrow (\neg Ry \wedge \neg\exists u(Suy) \wedge (y \neq min_1)] \wedge$$

$$[\forall v(\neg Rv \Rightarrow \exists u(Suv)) \Rightarrow (Ry \wedge \neg\exists u(Suy) \wedge (y \neq min_2)]$$

where $S, T \notin \sigma$.

Since FO + IFP captures PTIME on ordered structures and since this example shows how to define an order on sets with just one unary relation, we conclude that on the class of such structures FO + sc-IFP captures PTIME. □

The logic FO + sc-IFP is also nondeterministic, but we will now show that all formulas of this logic define only semi-deterministic queries.

Proposition 2.14 Let $\theta \equiv (\textbf{sc-ifp}_{S\bar{x},T\bar{y}}\psi, \varphi)(\bar{t})$ be an FO + sc-IFP(σ) formula. Let $\{S_1^0\ T_1^0, S_1^1\ T_1^1, \ldots\}$ and $\{S_2^0\ T_2^0, S_2^1\ T_2^1, \ldots\}$ be two sequences of computations of θ on an input structure $\mathfrak{A} \in \text{fin}(\sigma)$. Then for each $i \geq 0, (\mathfrak{A}, S_1^i, T_1^i) \cong (\mathfrak{A}, S_2^i, T_2^i)$.

Proof We proceed by induction on i.

Basis. $(i = 0)$. In this case, $S_1^0 = S_2^0 = \emptyset, T_1^0 = T_2^0 = \emptyset$. Thus, $(\mathfrak{A}, S_1^0, T_1^0) \cong (\mathfrak{A}, S_2^0, T_2^0)$.

Induction. We assume that $(\mathfrak{A}, S_1^i, T_1^i) \cong (\mathfrak{A}, S_2^i, T_2^i)$ and prove that $(\mathfrak{A}, S_1^{i+1}, T_1^{i+1}) \cong (\mathfrak{A}, S_2^{i+1}, T_2^{i+1})$.

We first note that it suffices to prove that $(\mathfrak{A}, S_1^i, T_1^{i+1}) \cong (\mathfrak{A}, S_2^i, T_2^{i+1})$. This is because S^{i+1} is defined by ψ, a deterministic formula, and the induction step involves no nondeterminism. Therefore isomorphism is preserved in the definition of S^{i+1}.

Next, we prove that $(\mathfrak{A}, S_1^i, T_1^{i+1}) \cong (\mathfrak{A}, S_2^i, T_2^{i+1})$.

Let $T_1^{i+1} = \textbf{choice}(X_1)$ and $T_2^{i+1} = \textbf{choice}(X_2)$ where $X_1 = \{\bar{b} \mid (\mathfrak{A}, S_1^i, T_1^i) \models \varphi(\bar{b})\}$ and $X_2 = \{\bar{b} \mid (\mathfrak{A}, S_2^i, T_2^i) \models \varphi(\bar{b})\}$. From the induction hypothesis it follows that $(\mathfrak{A}, S_1^i, T_1^i, X_1) \cong (\mathfrak{A}, S_2^i, T_2^i, X_2)$. Let $\pi : (\mathfrak{A}, S_1^i, T_1^i, X_1) \cong (\mathfrak{A}, S_2^i, T_2^i, X_2)$.

Now X_1 is an automorphism class of $(\mathfrak{A}, S_1^i, T_1^i)$ if and only if X_2 is an automorphism class of $(\mathfrak{A}, S_2^i, T_2^i)$. So $T_1^{i+1} \neq \emptyset$ if and only if $T_2^{i+1} \neq \emptyset$. If $T_1^{i+1} = T_2^{i+1} = \emptyset$, clearly $\pi : (\mathfrak{A}, S_1^i, T_1^{i+1}) \cong (\mathfrak{A}, S_2^i, T_2^{i+1})$. Otherwise, let $T_1^{i+1} = \{\bar{b}_1\}$ and $T_2^{i+1} = \{\bar{b}_2\}$. Clearly $\pi(\bar{b}_1) \in X_2$. So X_2 is an automorphism class of $(\mathfrak{A}, S_2^i, T_2^i)$. This means that for any $\bar{b}, \bar{b}' \in X_2$, there is an automorphism ρ of $(\mathfrak{A}, S_2^i, T_2^i)$ such that $\rho(\bar{b}) = \bar{b}'$. Therefore there is an automorphism ρ of $(\mathfrak{A}, S_2^i, T_2^i)$ such that $\rho(\pi(\bar{b}_1)) = \bar{b}_2$. Thus $\rho \circ \pi : (\mathfrak{A}, S_1^i, T_1^{i+1}) \cong (\mathfrak{A}, S_2^i, T_2^{i+1})$.

This proves the proposition. □

Curiously, we do not allow formulas of the form $\varphi \wedge \psi$ or $\varphi \vee \psi$ where φ and ψ are FO + sc-IFP formulas. This restriction is crucial if we want our logic to remain semi-deterministic. The following example shows an instance where a conjunction of two FO + sc-IFP formulas causes a problem.

Example 2.15 Consider the vocabulary $\sigma = \langle R_1, R_2, E \rangle$ where R_1, R_2 are unary and E is binary. Consider the σ-structure $\mathfrak{A} = \langle \{a, b, c, d\}, R_1^{\mathrm{A}}, R_2^{\mathrm{A}}, E^{\mathrm{A}} \rangle$ where $R_1^{\mathrm{A}} = \{a, b\}, R_2^{\mathrm{A}} = \{c, d\}, E^{\mathrm{A}} = \{(a, c), (b, d)\}$.

Consider the different execution sequences of the formula

$$\varphi(x, y) \equiv (\textbf{sc-ifp}_{Sz, Tx}\ Tz, R_1 x)(x) \ \wedge \ (\textbf{sc-ifp}_{Sz, Ty}\ Tz, R_2 y)(y) \ \wedge \ Exy.$$

Note that R_1 and R_2 are automorphism classes of $\mathfrak{A}$. Both the first and second conjunct collect all the elements returned by the **choice** operation at each stage in S. A look at the structure $\mathfrak{A}$ will convince us that both the conjuncts reach the fixed point just after the first iteration. Thus the first conjunct has the effect of choosing one element from R_1 and the second conjunct has the effect of choosing one element from R_2. If no particular order is specified for the evaluation of the two conjuncts, then the different interpretations of φ are not isomorphic. For instance, $\varphi^{\mathrm{A}} = \{(a, c)\}$ if a and c are chosen by the first and second conjuncts of φ, whereas $\varphi^{\mathrm{A}} = \emptyset$ if a and d are chosen by the first and second conjuncts of φ. □

We will introduce a different logic later which uses choice but is

deterministic and also allows arbitrary nesting of the **sc-ifp** operator with other operators.

Remark 2.16 Note that *boolean* semi-deterministic queries are in fact deterministic queries. Therefore, all sentences of the form $Q_1 x_1 \ldots Q_m x_m \, \varphi(x_1, \ldots, x_m)$, where φ is an FO + sc-IFP formula and each Q_i is a quantifier $\exists$ or $\forall$, express deterministic queries. We will abuse terminology and call such formulas, "sentences" of FO + sc-IFP. □

We have seen that FO + sc-IFP is a more well-behaved logic than FO + c-IFP. Does it have the same expressive power? We next show that it does not. More precisely, we prove that the evenness query is not expressible in FO + sc-IFP.

Definition 2.17 A structure $\mathfrak{A}$ is said to be *rigid* if it does not have any non-trivial automorphisms, i.e., the only automorphism on $\mathfrak{A}$ is the identity mapping. □

Lemma 2.18 On the class of finite rigid structures, FO + sc-IFP has the same power as FO + IFP.

Proof

It is enough to show that each formula of the form $(\textbf{sc-ifp}_{S\bar{x},T\bar{y}}\psi, \varphi)(\bar{t})$ is equivalent to an FO + IFP formula. So let $\mathfrak{A}$ be a rigid structure and θ be such an FO + sc-IFP formula. Since $\mathfrak{A}$ is a rigid structure, any set of tuples from $\mathfrak{A}$ is an automorphism class iff it has exactly one element. Consider the relation T^{i+1} defined at the ith stage. Let us say the **choice** operation actually returns an element. This means that the choice set is an automorphism class which in turn means that it has exactly one element.

So it follows that θ is equivalent to the following formula θ' which defines S and T by a simultaneous induction.

$$\theta' \equiv (\textbf{ifp}_{S\bar{x},T\bar{y}} \; \psi, \;\; \varphi')(\bar{t})$$

where φ' is the formula $[\exists!\bar{z}(\varphi(\bar{z})) \wedge \varphi(\bar{y}]$. It is well known that simultaneous fixed points can be simulated by ordinary FO + IFP formulas. Hence the result. □

We need to introduce some more concepts before we can prove the next proposition.

Definition 2.19 For each class C of σ-structures and $n \geq 1$ we define

$$\mu_n(C) = \frac{\mid \{\mathfrak{A} \in C \mid |\mathrm{A}| = n\} \mid}{\mid \{\mathfrak{A} \in \mathit{fin}(\sigma) \mid |\mathrm{A}| = n\} \mid}.$$

We let $\mu(C) = lim_{n\to\infty}\mu_n(C)$ if it exists.

A property P of σ-structures is said to satisfy the *0-1 law* if $\mu(C)$ is either 0 or 1, where C is the class of σ-structures satisfying P. In other words, if the property P holds for almost all σ-structures or the property does not hold for almost all σ-structures. A logic is said to satisfy the 0-1 law if any property definable in the logic satisfies the 0-1 law. □

An example of a property which satisfies the 0-1 law is rigidity. In fact, almost all structures are rigid (see Compton (1989)). On the other hand, one can easily see that the property of evenness does not satisfy the 0-1 law.

Proposition 2.20 The evenness query cannot be expressed in the logic FO + sc-IFP.

Proof

It has been shown by Blass et al. (1985) that FO + IFP satisfies the 0-1 law. Since almost all structures are rigid, on almost all structures FO + sc-IFP = FO + IFP and hence FO + sc-IFP too satisfies the 0-1 law. This shows that the evenness query cannot be expressed in the logic FO + sc-IFP. □

Proposition 2.20 immediately implies that FO + sc-IFP does not capture PTIME. What about the related question of whether FO + sc-IFP

is contained in PTIME? This obviously does not hold because FO + sc-IFP defines nondeterministic queries in general. But we know that "sentences" of FO + sc-IFP define only deterministic queries, so we can modify the above question and ask whether the deterministic fragment of FO + sc-IFP lies within PTIME.

The answer is that we do not know! The evaluation of any FO + sc-IFP formula does an isomorphism test when it checks whether the choice set at each stage is an automorphism class. It is not known whether isomorphism testing can be done in PTIME. This motivates the definition of the next logic.

2.3 Specified Symmetric Choice fixpoint logic

We are going to introduce a restricted version of FO + sc-IFP in this section. This logic is called FO + ssc-IFP and uses the so called *specified symmetric choice fixpoint* operator, in symbols **ssc-ifp**. Any execution sequence ("run") of an FO + ssc-IFP formula can be simulated in PTIME.

The difference between this logic and the earlier one lies in the **ssc-ifp** operator. Apart from demanding that the choice set should be symmetric, we now also specify a set of mappings $\chi_{\bar{a}\bar{b}}$, one for each pair of elements $\bar{a}, \bar{b}$ from the choice set. "Automorphism check" now means to test whether for each pair of distinct elements $\bar{a}, \bar{b}$ in the choice set, $\chi_{\bar{a}\bar{b}}$ is an automorphism mapping $\bar{a}$ to $\bar{b}$. Further, the mappings are specified by an FO + IFP formula. This ensures that the evaluation of the **choice** operation at each stage can be done in PTIME.

Definition 2.21 The logic FO + ssc-IFP

Syntax

The set of formulas of the logic FO + ssc-IFP over a vocabulary σ is the least set $Form(\sigma)$ satisfying:

- Let $\psi(\bar{x}, S, T), \varphi(\bar{y}, S, T)$ and $\chi(\bar{z}, S, T) \in \text{FO} + \text{IFP}(\sigma \cup \{S, T\})$, where S, T are two relation symbols not in σ such that the length of $\bar{x}$ equals the arity of S, the length of $\bar{y}$ equals the arity of T and the length of $\bar{z}$ equals 2.arity$(T) + 2$, and let $\bar{t}$ be a tuple of terms whose length is equal to the length of $\bar{x}$. Then

$$(\textbf{ssc-ifp}_{S\bar{x},T\bar{y}}\ \psi,\ \varphi,\ \chi)(\bar{t}) \in \mathit{Form}(\sigma).$$

Semantics

The semantics of the **ssc-ifp** operator is the same as for the **sc-ifp** operator except for the following change in the meaning of the **choice** operator:

for each i, let X^i be the choice set $\{\bar{b} \mid (\mathfrak{A}, S^i, T^i) \models \varphi(\bar{b})\}$. For each $\bar{u}, \bar{v} \in X^i$, let $\chi_{\bar{u}\bar{v}} = \{(x, y) \mid (\mathfrak{A}, S^i, T^i) \models \chi(\bar{u}, \bar{v}, x, y)\}$. So each $\chi_{\bar{u}\bar{v}}$ is a binary relation. If for any two distinct tuples $\bar{a}, \bar{b} \in X^i$, $\chi_{\bar{a}\bar{b}}$ is the graph of a function which is an automorphism mapping $\bar{a}$ to $\bar{b}$, then T^{i+1} is defined to be $\{\bar{b}'\}$ where $\bar{b}'$ is the element returned by **choice**$(X^i, \{\chi_{\bar{u}\bar{v}} \mid \bar{u}, \bar{v} \in X^i\})$. Otherwise T^{i+1} is defined to be $\emptyset$. □

Remark 2.22 1. It can be shown that each FO + ssc-IFP formula as defined above is equivalent to a formula of the form $(\textbf{ssc-ifp}_{S\bar{x},T\bar{y}}\ \psi,\ \varphi,\ \chi)(\bar{t})$ where ψ, φ, and χ are first order formulas. The proof of this is much the same as the proof of a similar claim about FO + sc-IFP in Remark 2.12.

2. Using similar arguments to that given in Remark 2.8 we can conclude that FO + IFP $\subseteq$ FO + ssc-IFP. □

We give an example of the the use of the **ssc-ifp** operator.

Example 2.23 We saw in Example 2.13 that we can define an order on structures which have only one unary relation defined on them. Here we show that we can define the order with an FO + ssc-IFP formula. Essentially the same formulas in Example 2.13 work, but we also need to specify the

automorphisms now. That is easy to do. For each pair of elements a, b of the choice set, a mapping which transposes a and b and fixes the other elements is an automorphism of the desired type. We show that this can easily be formalized by a first order formula.

$$
\begin{aligned}
\theta(x,y) \;\equiv\; & (\textbf{ssc-ifp}_{Sxy,Ty}\psi, \varphi, \chi)(x,y) \\
\text{where} & \\
\psi(x,y) \;\equiv\; & \{\exists v(\neg Rv \;\wedge\; \neg\exists u(Suv)) \;\Rightarrow \\
& \quad [(x = min_1) \;\wedge\; Ty \;\wedge\; \neg\exists v(S\ min_1\ v)] \vee \\
& \quad [Ty \;\wedge\; \exists u(Sux) \;\wedge\; \neg\exists v(Sxv)]\} \\
& \wedge \{\forall v(\neg Rv \;\Rightarrow\; \exists u(Suv)) \;\Rightarrow \\
& \quad [(x = min_2) \;\wedge\; Ty \;\wedge\; \neg\exists v(S\ min_2\ v)] \vee \\
& \quad [Ty \;\wedge\; \exists u(Sux) \;\wedge\; \neg\exists v(Sxv)]\}, \\
\varphi(y) \;\equiv\; & [\exists v(\neg Rv \;\wedge\; \neg\exists u(Suv)) \;\Rightarrow\; (\neg Ry \;\wedge\; \neg\exists u(Suy) \;\wedge\; (y \neq min_1)] \;\wedge \\
& [\forall v(\neg Rv \;\Rightarrow\; \exists u(Suv)) \;\Rightarrow\; (Ry \;\wedge\; \neg\exists u(Suy) \;\wedge\; (y \neq min_2)] \\
\text{and} & \\
\chi(u,v,x,y) \equiv\; & (x = u \;\wedge\; y = v) \;\vee\; (x = v \;\wedge\; y = u) \\
& \vee\; (x = y \;\wedge\; x \neq u \;\wedge\; x \neq v)
\end{aligned}
$$

where $S, T \notin \sigma$.

We know that FO + IFP captures PTIME on ordered structures and this example shows how to define an order on sets with just one unary relation. Hence on the class of such structures FO + ssc-IFP captures PTIME. □

As we already noted, FO + ssc-IFP is a sublogic of FO + sc-IFP and so does not express evenness. This shows that it does not have the full power of FO + IFP + C. Conversely, are all the deterministic FO + ssc-IFP-definable queries definable in FO + IFP + C? Gire and Hoang (1998) show that the answer is "No!".

Theorem 2.24 (Gire and Hoang) There is a PTIME property of structures, which is not definable in FO + IFP + C but can be expressed by an FO + ssc-IFP formula.

We will look at the proof of this theorem and other related results in the next chapter.

2.4 An Extension with the Logical Reduction Operator

On rigid structures, the logics FO + sc-IFP and FO + ssc-IFP have the same expressive power as FO + IFP. The reason for this is that since there are no nontrivial automorphisms, there is no symmetry for the **choice** operation to exploit. In this section we add a new feature to our logics: the *logical reduction* operator. This gives us a new logic which we call FO + ssc-IFP + I. The most important result about this logic is that it subsumes FO + IFP + C and therefore has strictly more expressive power than FO + ssc-IFP (even on rigid structures).

Logical reductions between problems mean reductions which can be expressed in a logical language. This notion is derived from the idea of interpretation between theories and was used in Immerman (1987) and Dawar (1993). Here we use this notion in the form of a constructor integrated in the language itself.

The reduction operator is denoted **I**. The idea behind this operator is that sometimes it is easier to evaluate a query on an abstract view defined on the input structure rather than evaluating the query directly on the input structure. Before defining the operator **I** formally we present a motivating example.

Example 2.25 We saw in Example 2.23 that FO + ssc-IFP expresses all PTIME queries on the class of structures with just one unary relation. In particular, there is an FO + ssc-IFP formula *Even* such that on any input structure $\mathfrak{A}$ with just one unary relation R, R has even cardinality iff $\mathfrak{A} \models$ *Even*.

Now suppose we want to find whether a graph $G = (V, E)$ has an even number of edges. It is not evident how to write an FO + sc-IFP formula for it. There might not be enough symmetry in G to help us. We now show a different approach to the problem.

Define a new structure $\pi(G) = (B, R)$ over the scheme $\{R\}$ where

R is unary. B is defined to be V^2 and R is defined to be $\{(x,y) \in B \mid G \models E(x,y)\}$. It is clear that G has even number of edges iff R has an even number of elements. Further $\pi(G)$ is a unary structure with a single unary relation R so R has even cardinality iff $\pi(G) \models \mathit{Even}$. So we can say that G has an even number of edges iff $\pi(G) \models \mathit{Even}$.

The reduction operator allows us to define queries in this indirect manner. □

Definition 2.26 The I operator. Boolean queries.

Syntax

Suppose that we have:

- a schema σ, called the *source schema*,
- a schema $\tau = \langle R_1, \ldots, R_r \rangle$ called the *target schema*, with $\text{arity}(R_i) = n_i$, $1 \leq i \leq r$,
- a tuple of σ formulas $\pi = \langle \varphi_1, \ldots, \varphi_r \rangle$ where for each i, $\text{arity}(\varphi_i) = k_i n_i$ for some k_i, and such that $\bar{x}$ is the union of all the free variables occurring in the φ_i's, and
- a *sentence* ψ over τ.

Then

$$\varphi \equiv \mathbf{I}_{\bar{x}}(\varphi_1, \ldots, \varphi_r; \psi)$$

is a *sentence* over σ.

Semantics

For each σ-structure $\mathfrak{A}$, we define $\pi(\mathfrak{A})$ to be the τ-structure whose domain is $\mathrm{A}^{k_1} \cup \ldots \cup \mathrm{A}^{k_r}$ (i.e., for each distinct k_i we have a different sort in the target domain), and for each i,

$$R_i^{\pi(\mathfrak{A})} = \{(\bar{a}_1, \ldots, \bar{a}_{n_i}) \mid \bar{a}_1, \ldots, \bar{a}_{n_i} \in \mathrm{A}^{k_i} \text{ and } \mathfrak{A} \models \varphi_i(\bar{a}_1, \ldots, \bar{a}_{n_i})\}.$$

We now define $\mathfrak{A} \models \varphi$ iff $\pi(\mathfrak{A}) \models \psi$. □

Example 2.27 The query of the previous example can be defined by the FO + ssc-IFP + I formula $Even2 \ = \ \mathbf{I}_{xy}(E(x,y) \ ; \ Even)$. □

Note that the **I** operator as given in Definition 2.26 defines only boolean queries. We now present the version which allows non-boolean queries.

Definition 2.28 The I operator. Non-boolean queries.

Syntax

Let σ, τ, ψ be as in the previous definition. Let

$$\pi \ = \ \langle \varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r) \rangle$$

be a tuple of σ formulas with, for each i, $\text{arity}(\bar{x}_i) = k_i n_i$ for some k_i. Let $\bar{x}$ be the union of all the variables in all the $\bar{x}_i$'s and $\bar{y}$ be the union of all the variables in all the $\bar{y}_i$'s. Then

$$\varphi(\bar{y}) \ = \ \mathbf{I}_{\bar{x}}(\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r) \ ; \ \psi)$$

is a *formula* whose set of free variables is exactly $\bar{y}$.

Semantics

For each σ-structure $\mathfrak{A}$ and each $\bar{y}$, define $\pi_{\bar{y}}(\mathfrak{A})$ to be the τ-structure whose domain is $\mathrm{A}^{k_1} \cup \cdots \cup \mathrm{A}^{k_r}$ (again, for each distinct k_i we have a different sort in the target domain), and for each i,

$$R_i^{\pi_{\bar{y}}(\mathfrak{A})} \ = \ \{(\bar{a}_1, \ldots, \bar{a}_{n_i}) \ \mid \ \bar{a}_1, \ldots, \bar{a}_{n_i} \in \mathrm{A}^{k_i} \text{ and } \mathfrak{A} \models \varphi_i(\bar{a}_1, \ldots, \bar{a}_{n_i}, \ \bar{y}_i)\}.$$

We now define $\mathfrak{A} \models \varphi(\bar{y})$ iff $\pi_{\bar{y}}(\mathfrak{A}) \models \psi$. □

Example 2.29 The set of vertices of the graph $G = (V, E)$ which have even in-degree is expressed by $Even{-}in{-}Degree(y) \equiv \mathbf{I}_x(E(x,y); \ Even)$. □

Definition 2.28 is the standard way of defining non-boolean formulas in the context of reductions. We now mention one obvious alternative approach which does not work. The idea is to allow non-boolean formulas to occur in place of ψ. This does not work because the elements of the target structure do not in general correspond to elements of the source structure. The previous example illustrates this. Each element of B corresponds to a pair of elements in V and not to an element of V. Therefore a non-boolean formula on the target structure returns a relation which does not have any meaning on the source structure.

The logical reduction operator is particularly interesting in the context of symmetry-based choice languages. Using the reduction operator **I** we can express some queries on a structure $\mathfrak{A}$ using symmetric properties of the structure $\pi(\mathfrak{A})$. There are cases when $\pi(\mathfrak{A})$ has more FO-definable symmetries than $\mathfrak{A}$. We saw this happen in Example 2.27 and Example 2.29. Later we will use the power of the **I** operator combined with symmetric choice to simulate all counting queries. This will show that the **I** operator strictly increases the power of FO + ssc-IFP. This is not the case for some other logics like FO + IFP because any query on a structure $\mathfrak{A}$ which is defined by an FO + IFP formula on $\pi(\mathfrak{A})$ can also be expressed by an FO + IFP formula on the input $\mathfrak{A}$ itself.

A very interesting feature of the logic FO + ssc-IFP + I that we are going to introduce is that it is strictly deterministic. Thus we have a logic which crucially uses the nondeterministic **choice** operator to gain expressive power but allows only deterministic queries. All these considerations make the definition of the logic a little complicated. We present the definition of this logic in stages. First we present the definition given in (Gire and Hoang (1998)).

Definition 2.30 The logic FO + ssc-IFP + I. Version 1.

Syntax

The set of formulas of the logic FO + ssc-IFP + I over a vocabulary σ is the least set $Form(\sigma)$ satisfying:

- if φ is an atomic formula over σ then $\varphi \in \mathit{Form}(\sigma)$
- if $\varphi \in \mathit{Form}(\sigma)$ then $\neg\varphi \in \mathit{Form}(\sigma)$
- if $\varphi, \psi \in \mathit{Form}(\sigma)$ then $\varphi \vee \psi \in \mathit{Form}(\sigma)$
- if $\varphi \in \mathit{Form}(\sigma)$ then $\forall x\varphi \in \mathit{Form}(\sigma)$
- Let $\varphi(\bar{x}, S) \in \mathit{Form}(\sigma \cup \{S\})$, where S is a relation symbol not in σ such that the length of $\bar{x}$ equals the arity of S, and let $\bar{t}$ be a tuple of terms whose length is equal to the length of $\bar{x}$. Then

  $$(\mathbf{ifp}_{S,\bar{x}}\ \varphi)(\bar{t}) \in \mathit{Form}(\sigma)$$

- Let $\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r)$ be formulas of $\mathit{Form}(\sigma)$ where $\bar{x}$ is the union of all the $\bar{x}_i$'s, $\bar{y}$ is the union of all the $\bar{y}_i$'s, where we treat the $\bar{y}_i$'s as parameters. Let ψ be an $\mathrm{FO} + \text{ssc-IFP}(\tau)$-*sentence* or an $\mathit{Form}(\tau)$-*sentence* where τ is the vocabulary of the target structure defined by the φ_i's.
  Then the formula

  $$\varphi(\bar{y}) \ \equiv\ \mathbf{I}_{\bar{x}}(\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r)\ ;\ \psi)$$

  is in $\mathit{Form}(\sigma)$.

Semantics

We have already defined the semantics of the **I** operator. □

Lemma 2.31 The logic $\mathrm{FO} + \text{ssc-IFP} + \mathrm{I}$ as defined in Definition 2.30 is deterministic.

Proof

We have to prove that all formulas of $\mathrm{FO} + \text{ssc-IFP} + \mathrm{I}$ are deterministic. The proof is by induction on the complexity of the formula. The only nontrivial case is when

$$\varphi(\bar{y}) \ \equiv\ \mathbf{I}_{\bar{x}}(\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r)\ ;\ \psi).$$

By induction hypothesis $\varphi_1, \ldots, \varphi_r$ are deterministic. If $\psi \in Form(\tau)$ then by induction hypothesis ψ is deterministic. If $\psi \in \text{FO} + \text{ssc-IFP}(\tau)$ then since ψ is a sentence and since boolean semi-deterministic queries are deterministic, ψ is deterministic.

Given any source structure $\mathfrak{A}$ and any $\bar{y}, \varphi_1, \ldots, \varphi_r$ define a unique target structure $\pi_{\bar{y}}(\mathfrak{A})$. The deterministic sentence ψ evaluates to a unique value on $\pi_{\bar{y}}(\mathfrak{A})$. Thus $\varphi(\bar{y})$ is a deterministic query. □

There is one feature of FO + ssc-IFP + I as given in Definition 2.30 that we feel is too restrictive. We require all the formulas which define the target structure (the $\varphi_i's$) to belong to $Form(\sigma)$. We know that all these are deterministic formulas. So the φ_i's cannot define an order on the input structure. If a naked set is given as input, there is no way to define a target structure which is the set with an order on it. We will later see that when we simulate counting we need precisely such a capability. With all these points in mind, we propose two different versions of the logic.

Definition 2.32 The logic FO + ssc-IFP + I. Version 2.

Syntax

The set of formulas of the logic FO + ssc-IFP + I over a vocabulary σ is the least set $Form(\sigma)$ satisfying:

- if φ is an atomic formula over σ then $\varphi \in Form(\sigma)$
- if $\varphi \in Form(\sigma)$ then $\neg\varphi \in Form(\sigma)$
- if $\varphi, \psi \in Form(\sigma)$ then $\varphi \vee \psi \in Form(\sigma)$
- if $\varphi \in Form(\sigma)$ then $\forall x\varphi \in Form(\sigma)$
- Let $\varphi(\bar{x}, S) \in Form(\sigma \cup \{S\})$, where S is a relation symbol not in σ such that the length of $\bar{x}$ equals the arity of S, and let $\bar{t}$ be a tuple of terms whose length is equal to the length of $\bar{x}$. Then

$$(\mathbf{ifp}_{S,\bar{x}}\ \varphi)(\bar{t}) \in Form(\sigma)$$

- Let $\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r)$ be formulas of $Form(\sigma)$ or $\mathrm{FO} + \text{ssc-IFP}(\sigma)$ where $\bar{x}$ is the union of all the $\bar{x}_i$'s, $\bar{y}$ is the union of all the $\bar{y}_i$'s, where we treat the $\bar{y}_i$'s as parameters. *The important restriction on the φ_i's is that if $\varphi_i \in FO + ssc\text{-}IFP(\sigma)$, then it has no parameters, i.e. $\bar{y}_i$ is empty.*

  Let ψ be an $\mathrm{FO} + \text{ssc-IFP}(\tau)$-sentence or an $Form(\tau)$-sentence where τ is the vocabulary of the target structure defined by the φ_i's.
  Then the formula

$$\varphi(\bar{y}) \;=\; \mathbf{I}_{\bar{x}}(\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r) \;;\; \psi)$$

  is in $Form(\sigma)$.

Semantics

There is a change in the definition of the target structure $\pi_{\bar{y}}(\mathfrak{A})$. It is defined as follows:

Let $\varphi_1, \ldots, \varphi_m$ be the formulas in $\mathrm{FO} + \text{ssc-IFP} + \mathrm{I}$ and let $\varphi_{m+1}, \ldots, \varphi_r$ be the formulas in $\mathrm{FO} + \text{ssc-IFP}$. For each $i, m+1 \leq i \leq r$, define $Sort_i$ to be a copy of A^{k_i}. Then the domain of $\pi_{\bar{y}}(\mathfrak{A})$ is $\mathrm{A}^{k_1} \cup \cdots \cup \mathrm{A}^{k_m} \uplus Sort_{m+1} \uplus \cdots \uplus Sort_r$, i.e., we form a new "sort" for each of the nondeterministic queries among the φ_i's.

For each $i, 1 \leq i \leq m$, define
$R_i^{\pi_{\bar{y}}(\mathfrak{A})} \;=\; \{(\bar{a}_1, \ldots, \bar{a}_{n_i}) \;\mid\; \bar{a}_1, \ldots, \bar{a}_{n_i} \in \mathrm{A}^{k_i} \text{ and } \mathfrak{A} \models \varphi_i(\bar{a}_1, \ldots, \bar{a}_{n_i},\ \bar{y}_i)\}$.
For each $i, m+1 \leq i \leq r$, define
$R_i^{\pi_{\bar{y}}(\mathfrak{A})} \;=\; \{(\bar{a}_1, \ldots, \bar{a}_{n_i}) \;\mid\; \bar{a}_1, \ldots, \bar{a}_{n_i} \in Sort_i \text{ and } \mathfrak{A} \models \varphi_i(\bar{a}_1, \ldots, \bar{a}_{n_i},\ \bar{y}_i)\}$.

As usual we define $\mathfrak{A} \models \varphi(\bar{y})$ iff $\pi_{\bar{y}}(\mathfrak{A}) \models \psi$. □

Lemma 2.33 The logic $\mathrm{FO} + \text{ssc-IFP} + \mathrm{I}$ as defined in Definition 2.32 is deterministic.

Proof

Let us consider the formula

$$\varphi(\bar{y}) \equiv \mathbf{I}_{\bar{x}}(\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r) \ ; \ \psi)$$

and the role played by $\varphi_1, \ldots, \varphi_r$ in it. Their only role is to define the target structure. We create a new sort in the target structure whenever the defining formula is semi-deterministic. This ensures that in the target structure, each sort has either has exactly one semi-deterministic relation defined on it or all the relations defined on the sort are deterministic. Now these considerations and Proposition 2.14 imply that any two target structures defined by the same tuple $\langle \varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r) \rangle$ are isomorphic to each other. Also note that we disallow parameters for defining formulas which are FO + ssc-IFP formulas. Further the target formulas are just sentences. So they will evaluate the same on isomorphic structures. This proves that $\varphi(\bar{y})$ is deterministic. □

We also note that the restrictions we have imposed are necessary to obtain determinism:

1. Semi-deterministic formulas cannot be allowed to carry parameters. That will make $\varphi(\bar{y})$ nondeterministic.

2. We can have at most one nondeterministic relation defined on each sort. Otherwise the different target structures do not remain isomorphic any longer. We give an example of this below.
Consider a set $\mathrm{A} = \{a, b, c\}$. Let us say we define two binary relations R and S on A by means of semi-deterministic formulas φ and ψ. Let $R_1 = \{(a, b), (b, c)\}$ and $R_2 = \{(c, b), (b, a)\}$ be the result of two "runs" of φ on A. Let $S = \{(a, b), (b, c)\}$ be the result of one "run" of ψ on A. Clearly (A, R_1, S) is not isomorphic to (A, R_2, S) even though $(\mathrm{A}, R_1) \cong (\mathrm{A}, R_2)$.

We now define a third and final version of the logic FO + ssc-IFP + I. This is the official version we use henceforth. We first motivate the need for this new definition. We will later see that on some classes of structures we can define a linear order using an FO + ssc-IFP formula. From this it follows that all *Boolean* PTIME queries on those classes are definable in the logic FO + ssc-IFP. We cannot express arbitrary PTIME queries because only boolean FO + ssc-IFP formulas define deterministic queries. Now if we can

create a target structure which is the input structure augmented with this FO + ssc-IFP-definable order, then we can express all PTIME queries on the target structure in the logic FO + IFP itself. Our next definition makes this possible.

Definition 2.34 The logic FO + ssc-IFP + I. Version 3.

Syntax

The set of formulas of the logic FO + ssc-IFP + I over a vocabulary σ is the least set $Form(\sigma)$ satisfying:

- if φ is an atomic formula over σ then $\varphi \in Form(\sigma)$
- if $\varphi \in Form(\sigma)$ then $\neg\varphi \in Form(\sigma)$
- if $\varphi, \psi \in Form(\sigma)$ then $\varphi \vee \psi \in Form(\sigma)$
- if $\varphi \in Form(\sigma)$ then $\forall x\varphi \in Form(\sigma)$
- Let $\varphi(\bar{x}, S) \in Form(\sigma \cup \{S\})$, where S is a relation symbol not in σ such that the length of $\bar{x}$ equals the arity of S, and let $\bar{t}$ be a tuple of terms whose length is equal to the length of $\bar{x}$. Then

$$(\mathbf{ifp}_{S,\bar{x}}\ \varphi)(\bar{t}) \in Form(\sigma)$$

- Let $\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r)$ be formulas of $Form(\sigma)$ or FO + ssc-IFP(σ) where $\bar{x}$ is the union of all the $\bar{x}_i$'s, $\bar{y}$ is the union of all the $\bar{y}_i$'s, where we treat the $\bar{y}_i$'s as parameters.
  There are two important restriction on the φ_i's:
  1. if $\varphi_i \in FO + ssc\text{-}IFP(\sigma)$, then it has no parameters, i.e. $\bar{y}_i$ is empty.
  2. for each distinct sort there is at most one semi-deterministic relation, i.e., at most one $FO + ssc\text{-}IFP(\sigma)$ formula associated with that sort. In more mathematical language, if φ_j and φ_k are $FO + ssc\text{-}IFP(\sigma)$ formulas, then $n_j \neq n_k$.

  Let ψ be an FO + ssc-IFP(τ)-sentence or an $Form(\tau)$-sentence where τ is the vocabulary of the target structure defined by the φ_i's.

Then the formula

$$\varphi(\bar{y}) \;=\; \mathbf{I}_{\bar{x}}(\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r) \;;\; \psi)$$

is in $Form(\sigma)$.

Semantics

The semantics is exactly as given in Definition 2.30. We have placed all the necessary restrictions on the syntax. □

Lemma 2.35 The logic FO + ssc-IFP + I as defined in Definition 2.34 is deterministic.

Proof

Let us consider the formula

$$\varphi(\bar{y}) \;\equiv\; \mathbf{I}_{\bar{x}}(\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r) \;;\; \psi)$$

and the role played by $\varphi_1, \ldots, \varphi_r$ in it. Their only role is to define the target structure. The definition ensures that in the target structure, each sort has at most one semi-deterministic relation and any number of deterministic relations defined on it. Suppose that $\mathfrak{B}$ and $\mathfrak{C}$ are two target structures defined by the same tuple $\langle \varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_r(\bar{x}_r, \bar{y}_r) \rangle$. We will prove that $\mathfrak{B} \cong \mathfrak{C}$. Without loss of generality consider $\mathfrak{B}$ and $\mathfrak{C}$ to have just one sort. Let φ_1 be the only FO + ssc-IFP formula among the φ_i's. Let us suppose one "run" of it defines the relation R which is part of $\mathfrak{B}$ and another "run" defines R' which is part of $\mathfrak{C}$. Now we know that $(\mathrm{A}, R) \cong (\mathrm{A}, R')$. The rest of the relations of $\mathfrak{B}$ and $\mathfrak{C}$ can be considered to be relations defined on (A, R) and (A, R') by *deterministic formulas.* This implies that $\mathfrak{B} \cong \mathfrak{C}$. Also note that we disallow parameters for defining formulas which are FO + ssc-IFP formulas. Further the target formulas are just sentences. So they will evaluate the same on isomorphic structures. This proves that $\varphi(\bar{y})$ is deterministic. □

Remark 2.36 It is easy to see that FO + ssc-IFP + I $\subseteq$ PTIME.
We only need to argue for the case of a formula using the **I** operator. But the

I operator just defines a target structure using a finite number of formulas and evaluates some formula on the target structure, where all these formulas are known to be in PTIME. This proves the claim. □

2.5 Simulation of Counting

In this section we will give a proof of the fact that counting can be simulated by using the capabilities of the symmetric choice fixpoint operator and the logical reduction operator.

Theorem 2.37 $\mathrm{FO} + \mathrm{IFP} + \mathrm{C} \subseteq \mathrm{FO} + \text{ssc-IFP} + \mathrm{I}$.

Proof

Let us fix a vocabulary $\sigma = \langle R_1, \ldots, R_r \rangle$. Remember we are interested only in those $\mathrm{FO} + \mathrm{IFP} + \mathrm{C}(\sigma)$ formulas whose only free variables are point variables. Given any such formula $\theta(u_1, \ldots, u_m)$ we will show that there is an equivalent formula $\zeta(u_1, \ldots, u_m) \in \mathrm{FO} + \text{ssc-IFP} + \mathrm{I}(\sigma)$.

There are two main ideas in the simulation:

1. The formula θ has access to a number sort with a natural ordering. This enables θ to compare the results of **Count** terms and do other arithmetic on numbers. We simulate this feature by using the **I** operator to create a new sort which we will call *List*, which consists of a "large enough" sorted list. "Large enough" is explained now. Let $k = max\{|\bar{x}| \mid \mathbf{Count}(\bar{x}, \varphi(\bar{x})) = \bar{\mu} \text{ is a subformula of } \theta\}$. This means that the values of the **Count** terms range from 0 to $|\mathrm{A}|^k$. We choose *List* to consist of $|\mathrm{A}|^{k+2}$ elements. The role of the number variables occurring in θ is taken over by new variables in ζ referring to elements from *List*.

2. Each subformula of θ which is of the form $\mathbf{Count}(\bar{x}, \varphi(\bar{x}, \bar{y}, \bar{\nu})) = \bar{\mu}$ is replaced by an $\mathrm{FO} + \text{ssc-IFP} + \mathrm{I}$ formula which have free variables $\bar{y}$

and "new" free variable $\bar{z}$ and z' referring to elements of *List* replacing the free number variables $\bar{\nu}$ and $\bar{\mu}$ respectively. This simulation says that each element of the set defined by $\varphi(\bar{x}, \bar{y}, \bar{\nu})$ can be paired off with each element of the initial segment of *List* up to z.

We now present the formal details. We will build the formula ζ in stages.

1. Given any σ-structure $\mathfrak{A}$ we will first define a new target structure $\mathfrak{A}'_{\bar{u}}$ over the signature $\sigma_U = \sigma \cup \{U, E_1, \ldots, E_m\}$ whose restriction to σ is $\mathfrak{A}$, U is a unary relation symbol interpreted by a new sort consisting of a set of cardinality $|\mathrm{A}|^{k+2}$, and for $1 \leq i \leq m$, E_i is a unary relation symbol which will be interpreted by a relation containing a single element. There is different target structure for each assignment to the variables $\bar{u}$. Let

$$\varphi_{List}(x_1, \ldots, x_{(k+2).1}) \equiv (x_1 = x_1).$$

φ_{List} defines a new set *List* of cardinality $|\mathrm{A}|^{k+2}$. We use the subscript $(k+2).1$ to indicate the fact that *List* is a unary relation on A^{k+2} rather than a $(k+2)$-ary relation on A. The desired $\mathrm{FO} + \text{ssc-IFP} + \mathrm{I}(\sigma)$ formula is:

$$\zeta(\bar{u}) \;\equiv\; \mathbf{I}_{\bar{x},\bar{z}}(\varphi_{List}, R_1, \ldots, R_r, u_1 = z_1, \ldots, u_m = z_m;\ \zeta')$$

where ζ' is defined below and where $\bar{x}$ is the union of all the free variables in the formulas $R_1, \ldots, R_r$. It follows from the semantics of the $\mathbf{I}$ operator that

$$\mathfrak{A} \models \zeta(\bar{u})[\bar{a}] \;\textit{iff}\; \mathfrak{A}'_{\bar{a}} \models \zeta'. \tag{2.1}$$

2. ζ' is a σ_U *sentence*. In our case U is interpreted by *List* and $E_1, \ldots, E_m$ by unary relations containing a single element. Now ζ' defines a new target structure $\mathfrak{A}''_{\bar{u}}$ over the signature $\sigma_{Succ} = \sigma \cup \{Succ, E_1, \ldots, E_m\}$ whose restriction to σ is $\mathfrak{A}$, *Succ* is a binary relation symbol which is interpreted as a linear order on *List*, and $E_1, \ldots, E_m$ are interpreted as in $\mathfrak{A}'_{\bar{u}}$. *List* is just a set and thus an automorphism class. Hence we can write a formula φ_{Succ} which defines a linear order *Succ* on *List* using ideas similar to those in Example 2.23. The formulas in this case are much simpler though. They are given below:

$\varphi_{Succ}(x,y) \equiv (\textbf{ssc-ifp}_{Sxy,Tx}\psi, \varphi, \chi)(x,y)$
where
$\psi(x,y) \equiv [(x = min) \wedge Ty \wedge \forall z(\neg S\ min\ z)] \vee$
$[Ty \wedge \exists z(Szx) \wedge \forall z(\neg Sxz)]$
$\varphi(x) \equiv Ux \wedge \forall z(\neg Szx) \wedge (x \neq min)$
$\chi(u,v,x,y) \equiv (x = u \wedge y = v) \vee (x = v \wedge y = u)$
$\vee (x = y \wedge x \neq u \wedge x \neq v)$
where $S, T \notin \sigma_U$.

We now define ζ' to be the *sentence*

$$\zeta' \equiv \mathbf{I}_{xy,\bar{x}}(\varphi_{Succ}(x,y), R_1, \ldots, R_r, E_1, \ldots, E_m;\ \theta')$$

where θ' is defined below and where $\bar{x}$ is the union of all the free variables in the formulas $R_1, \ldots, R_r$. It follows from the semantics of the **I** operator that

$$\mathfrak{A}'_{\bar{a}} \models \zeta' \textit{ iff } \mathfrak{A}''_{\bar{a}} \models \theta'. \tag{2.2}$$

3. We will define θ' in such a way that

$$\mathfrak{A} \models \theta(\bar{u})[\bar{a}] \textit{ iff } \mathfrak{A}''_{\bar{a}} \models \theta'. \tag{2.3}$$

Equations 2.1, 2.2 and 2.3 imply that θ and ζ are equivalent. Our next step is to define the σ_{Succ} *sentence* θ'. In defining it, we will replace all number variables occurring in θ by new variables. Further, we will also replace the tuple of number variables $\bar{\mu}$ by a *single* new variable whenever $\bar{\mu}$ is the result of **Count** term. To make the simulation more uniform, we assume the following about the FO + IFP + C formula θ. θ will have several subformulas of the form $\textbf{Count}(\bar{x}, \varphi(\bar{x}, \bar{y}, \bar{\nu})) = \bar{\mu}$. Let $\bar{\mu} = \mu_k \ldots \mu_0$. We will assume that θ treats any such tuple $\bar{\mu}$ as an indivisible block of variables. That is, none of the variables μ_i is used as a separate individual variable in atomic formulas. This allows us to replace each such tuple $\bar{\mu}$ by a "new" number variable μ'. Now technically, the variable μ' ranges from 0 to A^k so this is not an FO + IFP + C formula, but we can modify the definition suitably to allow this change. It can also be shown that this is not a restriction on the logic FO + IFP + C. We reiterate that such a modification is done only on tuples $\bar{\mu}$ which are results of count terms. Once we have made

this change we can replace each number variable by a "new" variable referring to *List*.

$$\theta' \equiv \exists! u_1(\ldots(\exists! u_m(E_1u_1 \wedge \ldots \wedge E_mu_m \wedge \theta^*))\ldots).$$

We now define for each $\theta \in \mathrm{FO} + \mathrm{IFP} + \mathrm{C}(\sigma)$, $\theta^* \in \mathrm{FO} + \text{ssc-IFP} + \mathrm{I}(\sigma_{Succ})$. The definition is by induction on the complexity of θ.

- For each term t occurring in θ, we first define the corresponding term t^* that will appear in θ^*:
  - if x is a point variable, then $x^* \equiv x$.
  - if μ is a number variable, then $\mu^* \equiv z$, where z is a "new" variable.
- For each formula θ, we now define θ^*:
  - $(R_i x_1 \ldots x_l \mu_1 \ldots \mu_m)^* \equiv R_i x_1^* \ldots x_l^* \mu_1^* \ldots \mu_m^*$.
  - $(\neg\theta)^* \equiv \neg(\theta)^*$.
  - $(\theta_1 \vee \theta_2)^* \equiv \theta_1^* \vee \theta_2^*$.
  - $(\forall x(\theta))^* \equiv \forall x^*(\theta^*)$.
  - $(\forall \mu(\theta))^* \equiv \forall \mu^*(\exists z(Succ\ z\mu^* \vee Succ\mu^* z) \Rightarrow \theta^*)$.
    We are quantifying μ^* to belong to *List* in a roundabout way.
  - $((\mathbf{ifp}_{S,x_0\ldots x_k}\ \theta)(t_0\ldots t_k))^* \equiv (\mathbf{ifp}_{S,x_0^*\ldots x_k^*}\ \theta^*)(t_0^*\ldots t_k^*)$.
  - The only nontrivial case is when θ is of the form $\mathbf{Count}(\bar{x}, \varphi(\bar{x}, \bar{y}, \bar{\nu})) = \bar{\mu}$. We will explain below how to deal with this case.

4. We now detail the definition of θ^* when θ is of the form $\mathbf{Count}(\bar{x}, \varphi(\bar{x}, \bar{y}, \bar{\nu})) = \bar{\mu}$. θ^* is an σ_{Succ} formula which first creates a new target structure $\mathfrak{B}$ over the signature $\tau = \langle Succ, R, E\rangle$ where R and E are unary and $Succ$ is binary and then evaluates a τ-sentence ξ on this $\mathfrak{B}$. The interpretation of $Succ$ on $\mathfrak{B}$ is the successor relation on *List*, the intepretation of R is a set which is disjoint from *List* and E is interpreted as a set with a single element which is is also an element of *List*.

$$\theta^*(\bar{y}^*, \bar{\nu}^*, z) \equiv \mathbf{I}_{xy,\bar{x}^*,w}(Succ(x, y), \varphi^*(\bar{x}^*, \bar{y}^*, \bar{\nu}^*), w = z;\ \xi).$$

Note that the variable z would have been introduced in the formula θ^* where it would have been quantified to be an element of *List*. So we have that E is interpreted by a set containing a single element of *List*. The τ sentence ξ is given below.

5. Now we define the τ-sentence ξ on the structure $\mathfrak{B}$. Note that $z, \bar{y}^*$ and $\bar{\nu}^*$ have now taken the place of $\bar{\mu}, \bar{y}$ and $\bar{\nu}$ repectively, *List* has taken the place of the number sort and *Succ* has taken the place of the order $\leq$. Therefore an equivalent way of saying that $\varphi(\bar{x}, \bar{y}, \bar{\nu})$ has $\bar{\mu}$ elements is "the set $\varphi(\bar{x}^*, \bar{y}^*, \bar{\nu}^*)$ has the same cardinality as the initial segment of *List* from 1 (*not 0!*) to the element represented by z (which is the unique element of E)". That is exactly what we do.

$\xi \equiv \exists! z(Ez \wedge \exists xy(\xi'(x,y)))$
where
$\xi'(x,y) \equiv (\textbf{ssc-ifp}_{xy,y,uvxy}\ \xi_1, \xi_2, \xi_3)(x,y)$
and
$\xi_1(x,y) \equiv [\forall t(\neg Tt) \Rightarrow (\exists t(Szt) \wedge \forall s(\exists t(Sst) \Rightarrow Succ^* sz))]$
$\wedge [\exists t(Tt) \Rightarrow [(\forall st(\neg Sst) \wedge Ty \wedge \exists x_0(\forall s'(Succ^* x_0 s') \wedge Succ\ x_0 x)) \vee$
$(\exists s_1(\exists t_1(Ss_1t_1) \wedge \forall s'(\exists t'(Ss't') \Rightarrow Succ^* s's_1)$
$\wedge Succ\ s_1 x) \wedge Ty)]]$
where $Succ^*(x,y) \equiv (\textbf{ifp}_{S,xy}\ x = y \vee \exists z(Sxz \wedge Succ\ zy))(x,y)$,
$\xi_2(y) \equiv Ry \wedge \forall x(\neg Sxy)$ and
$\xi_3(u,v,x,y) \equiv (x = u \wedge y = v) \vee (x = v \wedge y = u)$
$\vee (x = y \wedge x \neq u \wedge x \neq v)$.

This completes the proof. □

CHAPTER 3

EXPRESSIVE POWER

In this chapter, we will prove three main results. We will show two classes of graphs on which the logic FO + ssc-IFP + I captures PTIME, and one class on which FO + sc-IFP + I captures PTIME. The basic idea is the same in all three cases. In the first two cases, we show that we can define an order by an FO + ssc-IFP + I formula and in the third an order is defined by an FO + sc-IFP + I formula. We start off this chapter by proving Theorem 2.24. The proof vitally uses the construction by Cai et al. (1992). We first present their construction.

3.1 FO + IFP + C is strictly contained in PTIME

In this section we outline the construction of a class of graphs and a query on this class which is in PTIME but not expressible by any FO + IFP + C formula. The result by Cai et al. actually show that the above mentioned query cannot be expressed by any $\mathcal{C}^{\omega}_{\infty\omega}$ formula. We assume the reader is familiar with this logic as also the logics $\mathcal{C}^{k}_{\infty\omega}$. We will freely use well known facts about these logics. Also we will state several lemmas about the graphs we construct, proofs of which can be found in Cai et al. (1992).

Fact. FO + IFP + C $\subseteq \mathcal{C}^{\omega}_{\infty\omega}$. Thus any query not expressible in $\mathcal{C}^{\omega}_{\infty\omega}$ is also not expressible in FO + IFP + C.

For each k, we write $\mathfrak{A}, \bar{a} \equiv^{\mathcal{C},k} \mathfrak{B}, \bar{b}$ to denote the fact that for all formulas $\varphi(\bar{x}) \in \mathcal{C}^{k}_{\infty\omega}$,

$$\mathfrak{A}, \bar{a} \models \varphi(\bar{x}) \;\Leftrightarrow\; \mathfrak{B}, \bar{b} \models \varphi(\bar{x}).$$

We give a similar meaning to $\mathfrak{A}, \bar{a} \equiv^{\mathcal{C},\omega} \mathfrak{B}, \bar{b}$.

Fact. If q is a query such that for every $k \geq 0$, there exist structures $\mathfrak{A}_k$ and $\mathfrak{B}_k$ satisfying:

$$\mathfrak{A}_k \in q,\ \mathfrak{B}_k \notin q,\ \mathfrak{A}_k \equiv^{\mathcal{C},k} \mathfrak{B}_k$$

then q is not expressible in $\mathcal{C}^{\omega}_{\infty\omega}$.

Now we actually come to the construction. We use the following gadget graphs $X = (V, E, \preceq)$ in our constructions. We describe them below:

$V = A \cup B \cup M$ where $A = \{a_1, a_2, a_3\},\ B = \{b_1, b_2, b_3\}$ and
$M = \{m_S \mid S \subseteq \{1, 2, 3\}$ of even cardinality $\} = \{m_0, m_{12}, m_{13}, m_{23}\}$.

$E = \{(m_S, a_i) \mid i \in S\} \cup \{(m_S, b_i) \mid i \notin S\}$.

Further $\preceq$ is a partial order on the vertices of the graph such that $\{a_1, b_1\} \prec \{a_2, b_2\} \prec \{a_3, b_3\} \prec \{m_0, m_{12}, m_{13}, m_{23}\}$. We call such a partial order an order of *width* 4.

Thus X consists of four middle vertices each of which is connected to one vertex from each of the pairs $\{a_1, b_1\}, \{a_2, b_2\}, \{a_3, b_3\}$. Furthermore each of the middle vertices is connected to an even number of a_i's. Also each vertex from among the a_i's and b_i's is connected to exactly two of the middle vertices.

Note that because of the partial order $\preceq$ any automorphism of X will leave the sets $\{a_i, b_i\}$ fixed as also the set $\{m_0, m_{12}, m_{13}, m_{23}\}$. We now state a crucial fact about the automorphisms of X.

Lemma 3.38 There are exactly four automorphisms of X. Each is determined by interchanging a_i and b_i for each i in some $S \subseteq \{1, 2, 3\}$ of even cardinality. Also any such automorphism which is not the identity mapping interchanges the middle vertices connected to the same vertex of the pair $\{a_j, b_j\}$ where $j \notin S$.

Let G be a *finite, connected, undirected, 3-regular*(i.e., each vertex of G has degree 3) graph. Further let $\leq$ be a linear order on the vertices of G. For each vertex v of G, we replace v by a copy of the graph

X defined earlier, call it $X(v)$. To each edge (v,w) incident on v we associate one of the pairs $\{a_i, b_i\}$ from $X(v)$, call this pair $\{a(v,w), b(v,w)\}$. Finally we connect the a vertices and b vertices at each end of each edge, that is we draw the edges $(a(v,w), a(w,v))$ and $(b(v,w), b(w,v))$. We call this new graph $X(G)$. We also extend the order $\leq$ on G to a partial order $\preceq$ on $X(G)$ in the natural manner. If for some edge (v,w) of G, we replace the edges $(a(v,w), a(w,v))$ and $(b(v,w), b(w,v))$ of $X(G)$ with the edges $(a(v,w), b(w,v))$ and $b(v,w), a(w,v))$ we are said to have introduced a "twist" in $X(G)$. Any graph with an arbitrary number of twists introduced in $X(G)$ is denoted $\hat{X}(G)$. We define $\tilde{X}(G)(X$ "twist" of $G)$ to be the graph in which exactly one twist is introduced in $X(G)$ at one of the edges incident with the least(with respect to $\leq$) vertex of G.

Now we state the main theorem of this section.

Theorem 3.39 There exists a sequence of graphs $\{T_k\}, k \in \mathbb{N}$, where each T_k is a finite, connected, undirected, 3-regular graph with a linear order $\leqslant$ defined on it, and a LOGSPACE computable query q such that for each k:

$$X(T_k) \equiv^{\mathcal{C},k} \tilde{X}(T_k),\ X(T_k) \in q,\ \tilde{X}(T_k) \notin q.$$

Therefore q is a PTIME query not expressible in $\mathcal{C}^{\omega}_{\infty\omega}$ and hence not expressible in FO + IFP + C.

3.2 Defining an order on $\hat{X}(G)$

We already saw in Section 2.5 that FO + IFP + C $\subseteq$ FO + ssc-IFP + I. In this section, we show that there are queries even definable in FO + ssc-IFP but which are not definable in FO + IFP + C. In particular, we show that the query q of Theorem 3.39 can be expressed by even an FO + ssc-IFP formula. The result we show is more general than that. We show that FO + ssc-IFP expresses all Boolean PTIME queries on the class of all graphs of the form $\hat{X}(G)$. For showing this, it suffices to show that there is an FO + ssc-IFP formula which defines a linear order uniformly on all graphs of the form $\hat{X}(G)$. We present a proof of this result next. It is essentially the same proof given in Gire and Hoang (1998) with some details filled in.

Theorem 3.40 Let $\mathcal{C} = \{\hat{X}(G) \mid G$ is a finite, connected, undirected, 3-regular graph with a linear order $\leq$ defined on it$\}$. There is an FO + ssc-IFP formula which defines a linear order on each graph in $\mathcal{C}$.

Proof

We fix a particular G satisfying the required conditions. The graph $\hat{X}(G) = (V, E)$ has a partial order $\preceq$ of width 4 defined on it. We only need to extend this to a total order $\leqq$.

First we introduce the following predicates which are easily FO definable using E and $\preceq$:

- $pair(x, x')$ iff x and x' are vertices of the same pair.
- $middle(m)$ iff m is a middle vertex.
- $V_3(s, t)$ iff s and t are vertices of the same copy of X, say $X(v)$ where v is a vertex of G.

Definition 3.41 A set of subgraphs of $\hat{X}(G)$ is called a *cycle* iff it consists of subgraphs $X(v_0), \ldots, X(v_n)$ such that for each i between 0 and $n-1$, v_i is adjacent to v_{i+1} and v_n is adjacent to v_0. □

Such a cycle has the following property: there is an automorphism of $\hat{X}(G)$ that exchanges the vertices of the pairs participating in the connections between the subgraphs of the cycle. The following two claims are crucial in defining a linear order.

Claim1 Let φ_C be an FO + IFP formula defining a cycle C of subgraphs of $\hat{X}(G)$ in the sense that $\varphi_C(x, y)$ iff $\{x, y\}$ is a pair participating in the connections between the subgraphs of C. Then the automorphism of $\hat{X}(G)$ which exchanges the vertices of the pairs participating in the connections between the subgraphs of C can be defined by an FO + IFP formula φ_f.

Proof of Claim1

Consider any subgraph $X(v)$ of the cycle C. Let $\{a_i, b_i\}$ and $\{a_j, b_j\}$ be the pairs of $X(v)$ participating in the connections between the subgraphs. From Lemma 3.38, we know that there is exactly one automorphism of $X(v)$ which exchanges a_i with b_i and a_j with b_j. We also know that the automorphism fixes the vertices a_k and b_k where $\{a_k, b_k\}$ is the third pair in $X(v)$, and exchanges the middle vertices connected to the same vertex of the pair $\{a_k, b_k\}$. Let us call this automorphism f_v. It follows from arguments in (Cai et al. (1992)) that for any cycle $X(v_0) \ldots X(v_n)$, the composition of mappings $f = f_{v_0} \circ \cdots \circ f_{v_n}$ is an automorphism of $\hat{X}(G)$. f is a mapping which

1. in each subgraph of C, exchanges vertices of pairs as described above
2. in each subgraph of C, exchanges middle vertices as described above
3. fixes all other vertices.

Now we give the formula φ_f which defines the automorphism f in the sense that $\varphi_f(u, v)$ iff f interchanges u and v.

$$
\begin{aligned}
\varphi_f(u,v) \equiv & \varphi_C(u,v) \\
& \vee \{ \mathit{middle}(u) \wedge \mathit{middle}(v) \wedge V_3(u,v) \\
& \quad \wedge \exists x x' y y' [\varphi_C(x,x') \wedge \neg \varphi_C(y,y') \ \wedge \ V_3(x,y) \wedge \mathit{pair}(y,y') \\
& \quad \wedge ((Eyu \ \wedge \ Eyv) \ \vee \ (Ey'u \ \wedge \ Ey'v))] \} \\
& \vee \{ (u = v) \ \wedge \ [(\neg \mathit{middle}(u) \ \wedge \ \neg \exists z (\varphi_C(u,z))) \\
& \quad \vee \ (\mathit{middle}(u) \ \wedge \ \neg \exists x x' (\varphi_C(x,x') \ \wedge \ V_3(u,x)))] \}.
\end{aligned}
$$

This proves the claim. □

Claim2 If in a copy of X, the vertices of two pairs are ordered by a relation $\leqslant$, then $\leqslant$ can be extended so that it also orders the middle vertices and the vertices of the third pair. Further, this extension can be defined by an FO formula $\varphi_{\leqslant}$.

Proof of Claim2

Suppose the two ordered pairs are $\{x, x'\}$ and $\{y, y'\}$ with, let us say, $x \leqslant x', y \leqslant y'$ and $\{x, x'\} \prec \{y, y'\}$. Let $\{z, z'\}$ be the third pair of X. Let $M = \{m_1, m_2, m_3, m_4\}$ be the set of its middle vertices. Recall that a middle vertex m is connected to exactly one of the vertices of each pair, and each vertex in a pair is connected to exactly two middle vertices. Let us say m_1 and m_2 are connected to x and m_3 and m_4 are connected to x'. Now we make $\{m_1, m_2\} \leqslant \{m_3, m_4\}$ (because $x \leqslant x'$). Further let m_1 and m_3 be connected to y and m_2 and m_4 be connected to y'. We now make $m_1 \leqslant m_2$ and $m_3 \leqslant m_4$ (because $y \leqslant y'$). This orders all the middle vertices. Now look at the third pair $\{z, z'\}$. We know that m_1 (the $\leqslant$-least of the middle vertices) is connected to exactly one of $\{z, z'\}$. Suppose it is connected to z. Then we make $z \leqslant z'$. Thus $\leqslant$ is a total ordering on the vertices of X. Further the following FO formula $\varphi_{\leqslant}$ defines $\leqslant$.

$$\varphi_{\leqslant}(u,v) \equiv u \leqslant v \ \vee \ \varphi_m(u,v) \ \vee \ [pair(u,v) \ \wedge \ \neg(u \leqslant v) \ \wedge \ \neg(v \leqslant u) \\ \wedge \exists m_1 m_2 m_3 m_4 (\varphi_m(m_1,m_2) \ \wedge \ \varphi_m(m_2,m_3) \ \wedge \ \varphi_m(m_3,m_4) \ \wedge \ Em_1u)]$$

where

$$\varphi_m(m,m') \equiv middle(m) \wedge middle(m') \ \wedge \ \exists x x' y y' \{pair(x,x') \wedge pair(y,y') \\ \wedge (x \leqslant x') \ \wedge \ (y \leqslant y') \ \wedge (x \prec y) \ \wedge \ V_3(x,y) \ \wedge \ V_3(x,m) \ \wedge \ V_3(m,m') \\ \wedge \{(Emx \ \wedge \ Em'x') \vee \\ [((Emx \ \wedge \ Em'x) \ \vee \ (Em'x' \ \wedge \ Em'x')) \ \wedge \ (Emy \ \wedge \ Em'y')]\}\}.$$

This proves the claim. □

We generate the order $\leqq$ by iterating the following steps:

Step 1. Compute a pair $\{a_i, b_i\}$ such that there is an FO + IFP definable automorphism between a_i and b_i.
Step2. Choose one of the vertices in $\{a_i, b_i\}$ (say a_i).
Step3. Order $a_i \leqq b_i$ and propagate the order such that:
if a pair is ordered then the pair connected to it is also ordered and if in a given copy of X two pairs are ordered then the middle vertices and the vertices of the third pair are also ordered.

Description of Step1. We compute a pair $\{a_i, b_i\}$ participating in a cycle C of $\hat{X}(G)$, such that the pairs connecting the subgraphs of C are all unordered.

Then the formula φ_f of Claim2 will give us an automorphism of $\hat{X}(G)$ that maps a_i to b_i. We next describe how to build C.

First we build a directed path P containing only unordered pairs $\{a_i, b_i\}$ from $\hat{X}(G)$ using the following rules:
Rule 1. $\{x_1, y_1\}$ is the first pair of P, where $\{x_1, y_1\}$ is the least unordered pair of $\hat{X}(G)$ according to the partial order $\preceq$.
Rule 2. If the pair $\{x_m, y_m\}$ of a subgraph $X(u)$ is in P and has not yet a successor then:
2a. if the pair $\{x_{m+1}, y_{m+1}\}$ that is connected to $\{x_m, y_m\}$ is not yet in the path, then add it to P as the successor of $\{x_m, y_m\}$, otherwise
2b. if no other pair of $X(u)$ is in the path, pick the first unordered pair (according to $\preceq$) of $X(u)$ and add it to P as the successor of $\{x_m, y_m\}$.

We note the following facts about P:
Remark1. P contains only unordered pairs. This is because $\{x_1, y_1\}$ is unordered and the propagation of the order $\leqq$ in Step 3 ensures that if a pair is unordered then the pair connected to it is also unordered.
Remark2. If the condition of 2b is satisfied for $\{x_m, y_m\}$, then there exists one more unordered pair in $X(u)$. This is because Step 3 ensures that no copy of X has exactly one unordered pair. Therefore Rule 2b determines a successor for $\{x_m, y_m\}$.
Remark3. Each pair of P has a unique successor, except the last pair, which is reached when both conditions 2a and 2b fail to hold.
Remark4. The successor of $\{x_m, y_m\}$ in P is either a pair in the same subgraph or it is the pair in the subgraph adjacent to $X(u)$ which is connected to $\{x_m, y_m\}$.

The directed path P can be defined by an FO + IFP formula φ_P in the sense that $\varphi_P(x, y, x'y')$ iff $\{x, y\}$ and $\{x', y'\}$ are two pairs such that for some m, $x = x_m, y = y_m, x' = x_{m+1}, y' = y_{m+1}$.

We will use the predicate *lu(x,y)* which says that $\{x, y\}$ is the least unordered pair of $\hat{X}(G)$ with respect to the partial order $\prec$. We also assume that S is some new uninterpreted relation symbol.

$$\varphi_P(x, y, x', y') \equiv (\mathbf{ifp}_{S,xyx'y'}\ \psi_P)(x, y, x', y').$$

where

$$\begin{aligned}\psi_P(x,y,x',y',S) \equiv\ & \{lu(x.y)\ \wedge\ pair(x',y')\ \wedge Exx'\ \wedge\ Eyy'\}\\ & \vee\ \{\exists uv(S\ uvxy)\ \wedge\ \neg\exists uv(S\ xyuv)\ \wedge\\ & \qquad [[pair(x',y')\ \wedge\ Exx'\ \wedge Eyy'\ \wedge\ \neg\exists uv(S\ x'y'uv)]\ \vee\\ & \qquad [\forall uv((pair(u,v)\ \wedge\ Exu\ \wedge\ Eyv)\ \Rightarrow\ \exists u_1\,v_1(S\ uvu_1\,v_1))\ \wedge\\ & \qquad \neg\exists uv(pair(u,v)\ \wedge\ V_3(x,u)\ \wedge\ \exists u_1\,v_1(S\ uvu_1\,v_1))\ \wedge\\ & \qquad pair(x',y')\ \wedge\ V_3(x,x')\ \wedge\ \neg(x'\leqq y')\ \wedge\ \neg(y'\leqq x')\ \wedge\\ & \qquad \forall uv((pair(u,v)\ \wedge\ V_3(x,u)\wedge\neg(u\leqq v)\ \wedge\ \neg(v\leqq u))\\ & \qquad\qquad\qquad \Rightarrow\ (x'\prec u))]\}\end{aligned}$$

and

$$\begin{aligned}lu(x,y)\ \equiv\ & \neg(x\leqq y)\ \wedge\ \neg(y\leqq x)\ \wedge\ pair(x,y)\\ & \wedge\forall uv[(pair(u,v)\ \wedge\ \neg(u\leqq v)\ \wedge\ \neg(v\leqq u))\ \Rightarrow\ x\prec u].\end{aligned}$$

The length of the path P is at most l where l is the number of pairs in $\hat{X}(G)$. So P reaches its last pair after at most l applications of Rule 2. Let $\{x_n, y_n\}$ be the last pair in P. Let $X(u_n)$ be the copy of X which contains the pair $\{x_n, y_n\}$. Then conditions 2a and 2b are violated at $X(u_n)$. This implies that there is at least one pair of $X(u_n)$ other than $\{x_n, y_n\}$ which is in P. Let $\{x_{m_0}, y_{m_0}\}$ be the last pair (with respect to the successor relation defined by φ_P) of $X(u_n)$ other than $\{x_n, y_n\}$ which is in P. Define CP to be $\{\{x_i, y_i\} \in P \mid m_0 \leq i \leq n\}$. Let C be the set of subgraphs containing the pairs in CP. To simplify notation let us denote the pair $\{x_i, y_i\}$ by p_i.

Claim3 C forms a cycle as given by Definition 3.41.

Proof of Claim3

We will show that each subgraph $X(v)$ in C has exactly two pairs in CP. It follows from definitions that $X(u_n)$ (the subgraph containing p_{m_0} and p_n) has exactly two pairs in CP. For each $X(v)$ in C where $v \neq u_n$, we will show that there exists an i with $m_0 < i < n$ such that the pairs p_i and p_{i+1} are in CP and no other pair of $X(v)$ is in CP.

Consider an arbitrary $X(v), v \neq u_n$. Let $p_i, m_0 < i < n$ be a pair contained in $X(v)$ (by definition there exists such a pair). Also both p_{i-1} and p_{i+1} are in CP. We will show that at least one of them is in $X(v)$. Suppose p_i is connected to p_{i-1}. Then condition 2a is violated by p_i and so p_{i+1} is

added to CP using Rule 2b, which means that p_{i+1} is in $X(v)$. Suppose p_i is not connected to p_{i-1}. Then p_i itself has been added to CP using Rule 2b, which means that p_{i-1} is in $X(v)$. Thus all the $X(v)$'s have at least two pairs in CP.

Now suppose that some $X(v)$ contains three pairs of CP. Let them be p_i, p_j, p_k and suppose $i < j < k$. So p_{i-1} is not in $X(v)$ and so it is connected to p_i by Remark4. Therefore the successor of p_i is obtained by Rule 2b and hence $j = i + 1$. But now Rule 2b cannot apply to p_{i+1} and so p_{i+2} is obtained by Rule 2a. It is the pair of an adjacent subgraph which is connected to p_{i+1}. So p_k is not the successor of p_{i+1}. It follows that p_k is the successor of p_{k-1} which is connected to it. All these considerations imply that both conditions 2a and 2b fail for p_k, which means that p_k is the last pair of P, contrary to our assumptions. This contradiction shows that none of the $X(v)$'s have more than two pairs of CP. They have exactly two. Further we can find an ordering of the subgraphs of C satisfying the conditions in Definition 3.41. Therefore C is a cycle. □

Our next aim is to find an FO + IFP formula φ_C which defines the cycle C in the sense that $\varphi_C(x, y)$ iff $\{x, y\}$ is a pair participating in the connections of the subgraphs of C. We make use of the formula φ_P defining P in the formula φ_C.

$$\begin{aligned}
\varphi_{Ptc}(x,y,x',y') \;\equiv\; & (\mathbf{ifp}_{S,xyx'y'}\ \varphi_P(x,y,x'',y'') \vee \\
& \exists x''y''(\varphi_P(x,y,x'',y'') \wedge S(x'',y'',x',y')))(x,y,x',y'). \\
\varphi_{last}(x,y) \;\equiv\; & \neg\exists uv(\varphi_P(x,y,u,v)). \\
\varphi_{Cmin}(x,y) \;\equiv\; & \exists uv(\varphi_{last}(u,v) \wedge V_3(x,u) \wedge \varphi_{Ptc}(x,y,u,v) \\
& \wedge \forall st((V_3(s,u) \wedge \varphi_{Ptc}(s,t,u,v)) \Rightarrow \varphi_{Ptc}(s,t,x,y))) \\
\varphi_C(x,y) \;\equiv\; & \exists uv(\varphi_{Cmin}(u,v) \wedge \varphi_{Ptc}(u,v,x,y)).
\end{aligned}$$

Step 1 wants us to pick an unordered pair $\{a_i, b_i\}$ such that there is an automorphism of $\hat{X}(G)$ which swaps the elements of this pair. Such a pair is very near at hand. Simplify define $\{a_i, b_i\}$ to be the least pair in C according to the order $\preceq$. The formula $\varphi(y)$ below defines the pair $\{a_i, b_i\}$ in the sense that $\varphi(y)$ holds iff y is a_i or b_i.

$\varphi(y) \equiv \exists x(\varphi_C^1(x,y) \vee \varphi_C^1(y,x))$
where
$\varphi_C^1(x,y) \equiv \varphi_C(x,y) \wedge \forall uv(\varphi_C(u,v) \Rightarrow (x \prec u))$.

From Claim1, there is an FO + IFP formula φ_f which defines an automorphism that exchanges the vertices of the pair $\{a_i, b_i\}$ defined by φ.

Description of Step2. This is the only nondeterministic step. We need to choose between either of a_i and b_i defined above. The choice set is defined by the formula φ given above. The formula which specifies the automorphisms is given by

$$\chi(u,v,x,y) \equiv \varphi(u) \wedge \varphi(v) \wedge \varphi_f(x,y).$$

Let us suppose that a_i is the element chosen by this step and that the relation T holds the chosen element at each stage.

Description of Step3. First we order the pair $\{a_i, b_i\}$ by $a_i \leqq b_i$. Then we propagate the order as described earlier. The propagation of this order is itself an iterative process and is defined by the following FO + IFP formula ψ.

$\psi(x,y) \equiv (\mathbf{ifp}_{\leqq,x,y}\ \psi_1)(x,y)$
where
$$\begin{aligned}\psi_1(x,y,\leqq) \equiv\ & (Tx \wedge pair(x,y)) \\ & \vee \exists uv(pair(u,v) \wedge pair(x,y) \wedge Eux \wedge Evy \wedge (u \leqq v)) \\ & \vee (pair(x,y) \wedge \forall uv((pair(u,v) \wedge Eux \wedge Evy) \Rightarrow \\ & \qquad (\neg(u \leqq v) \wedge \neg(v \leqq u))) \\ & \qquad \wedge \varphi_{\leqslant}(x,y))\end{aligned}$$
where $\varphi_{\leqslant}$ is the formula of Claim2.

Finally the ordering of the vertices of $\hat{X}(G)$ is obtained by iterating the three steps. It is defined by the following FO + ssc-IFP formula:

$$(\mathbf{ssc\text{-}ifp}_{\leqq xy,\ Ty}\ \psi, \varphi, \chi)(x,y).$$

We claim that when the computation of this formula reaches a fixed point, $\hat{X}(G)$ is totally ordered. For otherwise, Step 1 is still possible and the computation will continue. □

3.3 Generalized Quantifiers

In Section 3.1 we saw that FO + IFP + C does not express all PTIME queries. In particular, we constructed a class of graphs and a query q on this class which is in PTIME but not expressible by any FO + IFP + C formula. In Section 3.2 we showed that the particular query q given above is definable by an FO + ssc-IFP formula. In fact, we can define all Boolean PTIME queries on the counterexample class in FO + ssc-IFP.

In this section, we will first present a result of Hella (1996) very much in the spirit of Section 3.1. The result says that the addition of a finite set of *generalized quantifiers* to fixed point logic fails to capture PTIME. In particular, Hella constructs a class of counterexample graphs and a PTIME query q on this class which is not definable with fixed point logic augmented with finitely many generalized quantifiers. We will show using a strategy very similar to that used in Section 3.2 that we can define an order on the counterexample graphs in the logic FO + ssc-IFP. Thus there is an FO + ssc-IFP-definable query which is not expressed by any formula of FO + IFP with finitely many generalized quantifiers. We now present the details of the results. We first outline the construction of the counterexample graphs of Hella. Complete details can be found in the very readable paper by Hella. We also state some lemmas about these graphs without proof, details of which can again be found in citebib:hel.

Generalized quantifiers provide a minimal way of extending the expressive power of logics. For example, we know that the evenness query cannot be expressed in FO + IFP. The simplest way of making evenness definable is to add the associated quantifier Q_q to FO + IFP where q is a Boolean query such that for all structures $\mathfrak{A}, \mathfrak{A} \in q$ iff $|\mathrm{A}|$ is even.

Definition 3.42 The logic FO + IFP(Q_q).

Syntax

Let $\tau = \langle R_1, \ldots, R_k \rangle$ be a vocabulary where the arity of each R_i is n_i, and let q be a Boolean query on τ-structures. Then the set of formulas

of the logic $\mathrm{FO} + \mathrm{IFP}(Q_q)$ over a vocabulary σ is defined to be the least set $Form(\sigma)$ satisfying:

- if φ is an atomic formula over σ then $\varphi \in Form(\sigma)$
- if $\varphi \in Form(\sigma)$ then $\neg\varphi \in Form(\sigma)$
- if $\varphi, \psi \in Form(\sigma)$ then $\varphi \vee \psi \in Form(\sigma)$
- if $\varphi \in Form(\sigma)$ then $\forall x\varphi \in Form(\sigma)$
- Let $\varphi(\bar{x}, S) \in Form(\sigma \cup \{S\})$, where S is a relation symbol not in σ such that the length of $\bar{x}$ equals the arity of S, and let $\bar{t}$ be a tuple of terms whose length is equal to the length of $\bar{x}$. Then
$$(\mathbf{ifp}_{S,\bar{x}}\ \varphi)(\bar{t}) \in Form(\sigma)$$
- if $\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_k(\bar{x}_k, \bar{y}_k) \in Form(\sigma)$ and $\bar{y}_i$ is an n_i-tuple of distinct variables for each $1 \leq i \leq k$ and $\bar{x}$ is the union of all the variables in all the $\bar{x}_i$'s (the $\bar{x}_i$'s will be treated as parameters), then
$$\varphi(\bar{x}) \ \equiv\ Q_q\bar{y}_1 \ldots \bar{y}_k(\varphi_1(\bar{x}_1, \bar{y}_1), \ldots, \varphi_k(\bar{x}_k, \bar{y}_k)) \ \in\ Form(\sigma).$$

Semantics

We only need to define the semantics for the new clause. Let $\mathfrak{A}$ be the input structure. For all $1 \leq i \leq k$, let $\bar{a}_i$ be a tuple of elements from A such that $|\bar{a}_i| = |\bar{x}_i|$ and let $\bar{a}$ be the union of all the elements in all the $\bar{a}_i$'s. Define $\varphi_i^{\mathrm{A},\bar{a}_i} = \{\bar{b}_i \in \mathrm{A}^{n_i} \mid (\mathfrak{A}, \bar{a}_i, \bar{b}_i \models \varphi_i(\bar{x}_i, \bar{y}_i)\}$.

Now we define $(\mathfrak{A}, \bar{a}) \models \varphi(\bar{x})$ iff $\mathfrak{B} \in q$ where $\mathfrak{B}$ is the τ-structure $\langle \mathrm{A}, \varphi_1^{\mathrm{A},\bar{a}_1}, \ldots, \varphi_1^{\mathrm{A},\bar{a}_k} \rangle$. □

Let $\mathbf{Q}$ be a set of quantifiers. We define the extension of $\mathrm{FO} + \mathrm{IFP}$ with $\mathbf{Q}$ below:

Definition 3.43 The logic $\mathbf{FO} + \mathbf{IFP}(\mathbf{Q})$.

Add the syntactic and semantic rule for each quantifier $Q \in \mathbf{Q}$ simultaneously to the rules of $\mathrm{FO} + \mathrm{IFP}$. □

Definition 3.44 Type and Arity.

Let Q_q be a generalized quantifier, where q is a Boolean query on τ-structures. We say that Q_q is of *type* $\langle n_1, \ldots, n_k \rangle$ if $\tau = \langle R_1, \ldots, R_k \rangle$ and the arity of R_i is n_i for each $1 \leq i \leq k$. The *arity* of Q_q is $max\{n_1, \ldots, n_k\}$, and we say that Q_q is *n-ary* if its arity is at most n. We denote by $\mathbf{Q}_n$ the set of all *n-ary* quantifiers on finite structures. □

We again assume a knowledge of the logics $\mathcal{L}^{\omega}_{\infty\omega}$ and the various $\mathcal{L}^{k}_{\infty\omega}$'s. Just as we defined $\mathrm{FO} + \mathrm{IFP}(\mathbf{Q})$ we can define $\mathcal{L}^{\omega}_{\infty\omega}(\mathbf{Q})$ and $\mathcal{L}^{k}_{\infty\omega}(\mathbf{Q})$. We use the fact that for any set of generalized quantifiers $\mathbf{Q}$, $\mathrm{FO} + \mathrm{IFP}(\mathbf{Q}) \subseteq \mathcal{L}^{\omega}_{\infty\omega}(\mathbf{Q})$. So if we prove that a certain query q is not definable in $\mathcal{L}^{\omega}_{\infty\omega}(\mathbf{Q})$ then it immediately follows that q is not definable in $\mathrm{FO} + \mathrm{IFP}(\mathbf{Q})$.

For each k, we write $\mathfrak{A}, \bar{a} \equiv^{k,\mathbf{Q}} \mathfrak{B}, \bar{b}$ to denote the fact that for all formulas $\varphi(\bar{x}) \in \mathcal{L}^{k}_{\infty\omega}(\mathbf{Q})$,

$$\mathfrak{A}, \bar{a} \models \varphi(\bar{x}) \;\Leftrightarrow\; \mathfrak{B}, \bar{b} \models \varphi(\bar{x}).$$

We give a similar meaning to $\mathfrak{A}, \bar{a} \equiv^{\omega,\mathbf{Q}} \mathfrak{B}, \bar{b}$.

We use the fact that if q is a query such that for every $k \geq 0$, there exist structures $\mathfrak{A}_k$ and $\mathfrak{B}_k$ satisfying:

$$\mathfrak{A}_k \in q,\ \mathfrak{B}_k \notin q,\ \mathfrak{A}_k \equiv^{k,\mathbf{Q}} \mathfrak{B}_k$$

then q is not expressible in $\mathcal{L}^{\omega}_{\infty\omega}(\mathbf{Q})$.

The result by Hella vitally uses the following "building block":

Definition 3.45 Let $C = \{c_1, \ldots, c_{n+1}, d_1, \ldots, d_{n+1}\}$, where all the $2n+2$ elements $c_1, \ldots, c_{n+1}, d_1, \ldots, d_{n+1}$ are distinct, and let $\prec$ be the strict partial order of width 2 given by:
$x \prec y \Leftrightarrow x \in \{c_i, d_i\}$ and $y \in \{c_j, d_j\}$ for some $1 \leq i < j \leq n+1$.
Further let $P = \{c_1, \ldots, c_{n+1}\}$. We define two $(n+1)$-ary relations R' and R'' on C as follows:

- $R'(a_1 \ldots a_{n+1}) \Leftrightarrow a_1 \prec \cdots \prec a_{n+1}$ and $|\{i \mid a_i \notin P\}|$ is even.

- $R''(a_1 \dots a_{n+1}) \Leftrightarrow a_1 \prec \dots \prec a_{n+1}$ and $|\{i \mid a_i \notin P\}|$ is odd.

We define $\mathfrak{C}' = \langle C, R' \rangle$ and $\mathfrak{C}'' = \langle C, R'' \rangle$. □

Note that any automorphism of $\mathfrak{C}'$ or $\mathfrak{C}''$ or any isomorphism between $\mathfrak{C}'$ and $\mathfrak{C}''$ must be a bijective mapping $f : C \to C$ preserving the partial order $\prec$: $a \prec b \Leftrightarrow f(a) \prec f(b)$.

Lemma 3.46 Let $f : C \to C$ be a bijection preserving the partial order $\prec$. Then f is an automorphism of $\mathfrak{C}'$ and $\mathfrak{C}''$ iff the number

$$exc(f) = |\{i \in \{1, \dots, n+1\} \mid f(c_i) = d_i\}|$$

of c, d-exchanges of f is even. Similarly, f is an isomorphism $\mathfrak{C}' \to \mathfrak{C}''$ iff *exc(f)* is odd.

Assume that $n \geq 2$ and $\mathfrak{G} = \langle G, E^G \rangle$ is a finite, connected, undirected, $(n+1)$-regular graph. Let $<^G$ be a linear ordering of G, the set of vertices. Each vertex has degree $n+1$. Thus we can fix for each $u \in G$ a bijection $h_u : \{v \mid (u, v) \in E^G\} \to \{1, \dots, n+1\}$ such that for any two vertices v, w adjacent to u, $h_u(v) < h_u(w)$ iff $v <^G w$.

Definition 3.47 For each subset $S \subseteq G$ we define a structure
$\mathfrak{D}(\mathfrak{G}, S) = \langle D_G, R^D, E^D \rangle$ where R is $(n+1)$-ary and E is binary:
$D_G = G \times C$,
R^D is the set of all tuples $((u, a_1), \dots, (u, a_{n+1})) \in (D_G)^{n+1}$ such that either $u \notin S$ and $R'(a_1 \dots a_{n+1})$ or $u \in S$ and $R''(a_1 \dots a_{n+1})$,
E^D is the set of all pairs $((u, c_i), (v, c_j))$ and $((u, d_i), (v, d_j))$ in $(D_G)^2$ such that $(u, v) \in E^G, i = h_u(v)$ and $j = h_v(u)$. □

Thus $\mathfrak{D}(\mathfrak{G}, S)$ is obtained from the graph $\mathfrak{G}$ by replacing each vertex $u \in S$ by a copy of $\mathfrak{C}''$, each vertex $u \notin S$ by a copy of $\mathfrak{C}'$, and each edge with a double edge connecting the c-components and d-components of a pair of c, d-pairs in the corresponding copies of C.

Definition 3.48 Let u be the least element of G according to $<^G$. We define now

- $\mathfrak{A}(\mathfrak{G}) = \langle \mathfrak{D}(\mathfrak{G}, \emptyset), <^A \rangle$ and
- $\mathfrak{B}(\mathfrak{G}) = \langle \mathfrak{D}(\mathfrak{G}, \{u\}), <^B \rangle$.

where $\mathfrak{D}(\mathfrak{G}, \emptyset) = \langle A(\mathfrak{G}), R^A, E^A \rangle$ (say),
$\mathfrak{D}(\mathfrak{G}, \{u\}) = \langle B(\mathfrak{G}), R^B, E^B \rangle$ (say)
and $<^A = <^B$ is the relation:

$$(v, a) <^A (w, b) \Leftrightarrow v <^G w \ or \ (v = w \ and \ a \prec b)$$

on the set $A(\mathfrak{G}) = B(\mathfrak{G}) = G \times C$. □

Now we come to the main result in Hella (1996). The result says that for each n, there is a vocabulary σ_n and a PTIME query over σ_n structures which is not definable in $\mathcal{L}^{\omega}_{\infty\omega}(\mathbf{Q}_n)$ and hence not definable in $\mathrm{FO} + \mathrm{IFP}(\mathbf{Q}_n)$. This shows that the logic $\mathrm{FO} + \mathrm{IFP}(\mathbf{Q})$ does not capture PTIME where $\mathbf{Q}$ is a *finite* set of generalized quantifiers.

Theorem 3.49 For each $n \geq 2$ the following statement holds:

There exists a sequence of graphs $\{\mathfrak{G}_k\}, k \in \mathbb{N}$, where each $\mathfrak{G}_k$ is a finite, connected, undirected, $(n+1)$-regular graph with a linear order $<^{G_k}$ defined on it, and a PTIME computable query q_n such that for each k:

$$\mathfrak{A}(\mathfrak{G}_k) \equiv^{k,\mathbf{Q}_n} \mathfrak{B}(\mathfrak{G}_k), \ \mathfrak{A}(\mathfrak{G}_k) \in q_n, \ \mathfrak{B}(\mathfrak{G}_k) \notin q_n.$$

Thus q_n is a query not expressible in $\mathcal{L}^{\omega}_{\infty\omega}(\mathbf{Q}_n)$ and hence not definable in $\mathrm{FO} + \mathrm{IFP}(\mathbf{Q}_n)$.

Theorem 3.49 is analogous to Theorem 3.39. We now prove what can be considered an analogue of Theorem 2.24. More precisely, we show that for any finite set $\mathbf{Q}$ of generalized quantifiers, there is a vocabulary σ and a PTIME query over σ structures which cannot be expressed in $\mathrm{FO} + \mathrm{IFP}(\mathbf{Q})$, but which can be expressed in FO + ssc-IFP. In particular, the counterexample structures mentioned above can be linearly ordered in FO + ssc-IFP, which can thus express any Boolean PTIME query on those structures.

Theorem 3.50 Let $\mathfrak{X}(\mathfrak{G}) = \langle \mathfrak{D}(\mathfrak{G}, S), <^X \rangle$ where $\mathfrak{D}(\mathfrak{G}, S) = \langle X(\mathfrak{G}), R^X, E^X \rangle$ (say) for some finite, connected, undirected, $(n+1)$-regular graph $\mathfrak{G} = \langle G, E^G \rangle$ and some $S \subseteq G$. Let $<^X$ be given by:

$$(v, a) <^X (w, b) \Leftrightarrow v <^G w \text{ or } (v = w \text{ and } a \prec b).$$

There is an FO + ssc-IFP formula which defines a linear order on $\mathfrak{X}(\mathfrak{G})$.

Proof The proof is exactly identical to the proof of Theorem 3.40 except for some changes in some definitions and simpler formulas in some cases. We will content ourselves with just pointing out the relevant changes in the proof of Theorem 3.40.

The structure $\mathfrak{X}(\mathfrak{G})$ is a graph obtained when each node in $\mathfrak{G}$ is replaced by a copy of $\mathfrak{C}'$ or $\mathfrak{C}''$. We use $\mathfrak{C}$ to refer to either $\mathfrak{C}'$ or $\mathfrak{C}''$ when the difference between them does not matter. We first note that we can freely use the following predicates which are FO-definable using $<^X$:

- $pair(x, x')$ iff $x, x' \in \{(u, c_i), (u, d_i)\}$ for some $u \in \mathfrak{G}$ and some $1 \leq i \leq n+1$.
- $C(s, t)$ iff s and t are vertices of the same copy of $\mathfrak{C}$.

Note that we have no need for the predicate *middle* here. In the proof of Theorem 3.40 we use the predicate *middle* in the formulas which we wrote in Claim1 and Claim2. We will show how to write equivalent formulas there. Also note that we use the predicate name C here instead of V_3. So we will have to replace V_3 by C in all the formulas of Theorem 3.40.

The next change is in Definition 3.41. We define a *cycle* to be a set of subgraphs of $\mathfrak{X}(\mathfrak{G})$ which consists of subgraphs $\mathfrak{C}(v_0), \ldots, \mathfrak{C}(v_n)$ such that for each i between 0 and $n-1$, v_i is adjacent to v_{i+1} and v_n is adjacent to v_0.

The next change is in the following:

Claim1 Let φ_C be an FO + IFP formula defining a cycle C of subgraphs of $\mathfrak{X}(\mathfrak{G})$ in the sense that $\varphi_C(x, y)$ iff $\{x, y\}$ is a pair participating in the

connections between the subgraphs of C. Then the automorphism of $\mathfrak{X}(\mathfrak{G})$ which exchanges the vertices of the pairs participating in the connections between the subgraphs of C can be defined by an FO + IFP formula φ_f.

Proof of Claim1

The reasoning here is far simpler. It is easily seen that the function which just swaps the vertices of the pairs which participate in the connections between the subgraphs of C and leaves the other vertices fixed is in fact an automorphism of $\mathfrak{X}(\mathfrak{G})$. Such an automorphism f is defined by the formula below:

$\varphi_f(u,v) \equiv \varphi_C(u,v) \vee [(u=v) \wedge \neg\exists z(\varphi_C(u,z))]$. □

Even the statement of Claim2 is different.

Claim2 If in a copy of $\mathfrak{C}$, the vertices of all but one pair are ordered by a relation $\leqslant$, then $\leqslant$ can be extended so that it also orders the vertices of the remaining pair. Further, this extension can be defined by an FO formula $\varphi_{\leqslant}$.

Proof of Claim2

We just need some way of distinguishing between the vertices of the unordered pair. But that is easy to do. Let $x_1,\ldots,x_n$ be the first vertices in each of the ordered pair. Then exactly one vertex of the last pair holds the relation R with $x_1,\ldots,x_n$. Which of the vertices it is depends on whether the copy of $\mathfrak{C}$ in question is a copy of $\mathfrak{C}'$ or $\mathfrak{C}''$. The following formula effects the ordering:

$$\begin{aligned}\varphi_{\leqslant}(u,v) \equiv\ & (u \leqslant v) \vee [pair(u,v) \wedge \neg(u \leqslant v) \wedge \neg(v \leqslant u) \\ & \wedge [\exists x_1 \ldots x_n x'_1 \ldots x'_n (\textstyle\bigwedge_{i=1}^{n}(pair(x_i,x'_i) \wedge C(x_i,u) \wedge (x_i \leqslant x'_i)) \wedge \\ & \quad \textstyle\bigwedge_{i=1}^{n-1}(x_i \prec x_{i+1}) \wedge \\ & \quad ((u \prec x_1 \wedge Rux_1 \ldots x_n) \vee (x_n \prec u) \wedge Rx_1 \ldots x_n u) \\ & \qquad \vee \textstyle\bigvee_{j=1}^{n-1}(x_j \prec u \wedge u \prec x_{j+1} \wedge Rx_1 \ldots x_j u x_{j+1} \ldots x_n)))].\end{aligned}$$

This proves the claim. □

We also have to change Step 3 of the procedure to define the order

$\leqq$.

Step3. Order $a_i \leqq b_i$ and propagate the order such that:
if a pair is ordered then the pair connected to it is also ordered and if in a given copy of $\mathfrak{C}$ the vertices of all but one pair are ordered then the vertices of the remaining pair are also ordered.

These are all the changes to be made. Everything else is the same as in the proof of Theorem 3.40. In particular, the definitions of the formulas ψ, φ and χ are as before. The only changes are that the new definitions of φ_f, $\varphi_{\leqslant}$, and *pair* have to be used and the predicate V_3 should be replaced throughout by C. This concludes the proof. □

We end the section by formally stating the main result:

Theorem 3.51 For every $n \geq 2$, there is a vocabulary σ_n such that $\mathrm{FO} + \text{ssc-IFP}(\sigma_n)$ is not contained in $\mathrm{FO} + \mathrm{IFP}(\mathbf{Q}_n)(\sigma_n)$.

3.4 k-Reducible Structures

In Section 2.3, we restricted our logics by requiring that the formulas also specify the automorphisms of the choice set. The reason for this restriction was that we wanted to evaluate the formulas of our logic in PTIME. In this section we pursue a different approach to this problem. We study a class of structures on which the isomorphism test can be performed in PTIME. Thus instead of restricting the logic, we look at subclasses where the logic has the desired property. We start off with the key definitions.

Definition 3.52 Let $\mathfrak{A}, \mathfrak{B}$ be two σ-structures, where σ is a vocabulary which may contain constant symbols. Let $\bar{a} \in A^k, \bar{b} \in B^k$. We say that $(\mathfrak{A}, \bar{a})$ and $(\mathfrak{B}, \bar{b})$ are *k-equivalent*, denoted by $(\mathfrak{A}, \bar{a}) \equiv^k (\mathfrak{B}, \bar{b})$ if and only if for all $\mathrm{FO}(\sigma)$ formulas $\varphi(\bar{x})$ with at most k variables, we have

$$\mathfrak{A} \models \varphi(\bar{x})[\bar{a}] \Leftrightarrow \mathfrak{B} \models \varphi(\bar{x})[\bar{b}].$$

In particular, two tuples $\bar{c}, \bar{d} \in A^k$ are called k-equivalent if and only if $(\mathfrak{A}, \bar{c}) \equiv^k (\mathfrak{A}, \bar{d})$. □

Definition 3.53 A structure $\mathfrak{A}$ is called *k-reducible* if, for all $m \geq k$, any m-equivalence class is an automorphism class of $\mathfrak{A}$, i.e., for any $m \geq k$ and any two tuples $\bar{a}, \bar{b} \in A^m$, $(\mathfrak{A}, \bar{a}) \equiv^m (\mathfrak{A}, \bar{b}) \Rightarrow (\mathfrak{A}, \bar{a}) \cong (\mathfrak{A}, \bar{b})$. □

Consider the evaluation of an FO + sc-IFP formula φ on a k-reducible structure. In each choice step, instead of checking that the choice set is an automorphism class, it is now enough if we check whether the choice set is a k-equivalence class. It has been shown by Immerman and Lander (1990) that this test can be performed in PTIME. Therefore on the class of k-reducible structures, any FO + sc-IFP formula can be evaluated in PTIME.

Our next major result is to show that we can write an FO + sc-IFP formula defining a linear order on any k-reducible structure. As we remarked just before Definition 2.34 in Section 2.4, it immediately follows that we can define all PTIME queries on these structures in the logic FO + sc-IFP + I. We haven't defined this logic but it is easy to guess what it is. It is the same as FO + ssc-IFP + I except that we use the **sc-ifp** operator instead of the **ssc-ifp** operator.

The presentation of the next result is made easier by using the following variant of the logic FO + sc-IFP. Here, in each choice step, we take into account not just the present result of the **choice** operation, but also results of all the **choice** operations till the present stage.

Definition 3.54 The logic FO + sc-IFP*

Syntax

The syntax is the same as that of the logic FO + sc-IFP given in Definition 2.11 except that we use the operator **sc-ifp*** instead of **sc-ifp**.

Semantics

The semantics of the **sc-ifp*** operator is the same as for the **c-ifp** operator except for the following change:

in each choice step i, if the choice set is an automorphism class of $\langle \mathfrak{A}, S^i, T^0, \ldots, T^i \rangle$ then $T^{i+1} = \{\bar{b}\}$ where $\bar{b}$ is the element returned by the **choice** operation. If the choice set is not an automorphism class, T^{i+1} is defined to be $\emptyset$.

We can consider $\langle \mathfrak{A}, S^i, T^0, \ldots, T^i \rangle$ to be a structure on the vocabulary $\sigma \cup \langle S, \bar{w}_0, \ldots, \bar{w}_i \rangle$ where for each $1 \leq j \leq i$, $\bar{w}_j$ is a tuple of new constant symbols whose length is equal to the arity of T and which is interpreted by the only element of T^j. □

Note that on k-reducible structures, any FO + sc-IFP* formula is evaluated in PTIME.

Lemma 3.55 For each FO + sc-IFP* formula θ, there is an equivalent FO + sc-IFP formula θ'.

Proof

We will only provide a brief sketch of the details. Let $\theta \equiv (\textbf{sc-ifp}^*{}_{S\bar{x},\, T\bar{y}}\ \psi,\ \varphi)(\bar{t})$ be the given FO + sc-IFP* formula. The basic idea is to keep track of all the results of all the **choice** operation till the current stage. We can easily do that by simultaneously building in the same relation S' both the relation S and an ordered list containing the elements of $T^0, T^1, \ldots$ in that order. This can be easily achieved by standard techniques as can be found in Section 7.2 of Ebbinghaus and Flum (1995), for instance. □

The following property of k-reducibility is important to our considerations. The proof given in Gire and Hoang (1998) uses the notion of k-pebbles game. We present a more direct proof.

Proposition 3.56 Let $\mathfrak{A}$ be a structure over some signature σ, w be a constant symbol not in σ, c be any element of A, and $\mathfrak{A}'$ be a $\sigma \cup \{w\}$ structure

whose restriction to σ is $\mathfrak{A}$ such that $w^{\mathrm{A}'} = c$. If $\mathfrak{A}$ is k-reducible, then so is $\mathfrak{A}'$.

Proof

Let $(a_1, \ldots, a_m), (b_1, \ldots, b_m) \in \mathrm{A}^m$ for some $m \geq k$. Our aim is to prove that

$$\textit{if } (\mathfrak{A}', a_1, \ldots, a_m) \equiv^m (\mathfrak{A}', b_1, \ldots, b_m) \tag{3.1}$$

$$\textit{then } (\mathfrak{A}', a_1, \ldots, a_m) \cong (\mathfrak{A}', b_1, \ldots, b_m). \tag{3.2}$$

So suppose 3.1 holds. To prove 3.2 it suffices to prove the following:

$$(\mathfrak{A}, a_1, \ldots, a_m, c) \cong (\mathfrak{A}, b_1, \ldots, b_m, c). \tag{3.3}$$

Since $\mathfrak{A}$ is k-reducible and $m \geq k$ it suffices to prove:

$$(\mathfrak{A}, a_1, \ldots, a_m, c) \equiv^{m+1} (\mathfrak{A}, b_1, \ldots, b_m, c). \tag{3.4}$$

We proceed to show this now.
Since $\mathfrak{A}$ is a reduct of $\mathfrak{A}'$, 3.1 implies that

$$(\mathfrak{A}, a_1, \ldots, a_m) \equiv^m (\mathfrak{A}, b_1, \ldots, b_m).$$

which in turn implies (using the fact that $\mathfrak{A}$ is k-reducible and $m \geq k$) that

$$(\mathfrak{A}, a_1, \ldots, a_m) \equiv^{m+1} (\mathfrak{A}, b_1, \ldots, b_m). \tag{3.5}$$

The next thing we want to show is that for any $1 \leq i \leq m$,

$$(\mathfrak{A}, a_1, \ldots, a_{i-1}, c, a_{i+1}, \ldots, a_m) \equiv^{m+1} (\mathfrak{A}, b_1, \ldots, b_{i-1}, c, b_{i+1}, \ldots, b_m). \tag{3.6}$$

We first show that

$$(\mathfrak{A}', a_1, \ldots, a_{i-1}, c, a_{i+1}, \ldots, a_m) \equiv^m (\mathfrak{A}', b_1, \ldots, b_{i-1}, c, b_{i+1}, \ldots, b_m). \tag{3.7}$$

So suppose that $\mathfrak{A}' \models \varphi(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_m)[a_1, \ldots, a_{i-1}, c, a_{i+1}, \ldots, a_m]$ where φ is an $\mathrm{FO}(\sigma \cup \{w\})$ formula with at most m variables. It then follows that $\mathfrak{A}' \models \varphi'(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m)[a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_m]$ where φ' is got by replacing all free occurrences of x_i in φ by w. Note that φ' can contain other bound occurrences of x_i. So φ' is an $\mathrm{FO}(\sigma \cup \{w\})$ formula

with at most m variables and with $m-1$ free variables. Now 3.1 implies that $\mathfrak{A}' \models \varphi'(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m)[b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_m]$ which in turn implies that $\mathfrak{A}' \models \varphi(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_m)[b_1, \ldots, b_{i-1}, c, b_{i+1}, \ldots, b_m]$. The argument is clearly symmetric so we get 3.7. Now since $\mathfrak{A}$ is a reduct of $\mathfrak{A}'$ we have

$$(\mathfrak{A}, a_1, \ldots, a_{i-1}, c, a_{i+1}, \ldots, a_m) \equiv^m (\mathfrak{A}, b_1, \ldots, b_{i-1}, c, b_{i+1}, \ldots, b_m).$$

Since $m \geq k$ and $\mathfrak{A}$ is k-reducible we get 3.6.

We are now in a position to prove 3.4 using 3.5 and 3.6. Suppose that $\mathfrak{A} \models \varphi(x_1, \ldots, x_m, x_{m+1})[a_1, \ldots, a_m, c]$ where φ is an FO(σ) formula with at most $m+1$ variables. We prove that $\mathfrak{A} \models \varphi(x_1, \ldots, x_m, x_{m+1})[b_1, \ldots, b_m, c]$ by induction on the formula φ.

The case when φ is a boolean combination of subformulas is trivial. There are two nontrivial cases:

- $\varphi \equiv Qx_{m+1}(\psi)$ where Q is either $\exists$ or $\forall$. In this case x_{m+1} is not a free variable at all so $\mathfrak{A} \models \varphi(x_1, \ldots, x_m)[a_1, \ldots, a_m]$. But then 3.5 implies that $\mathfrak{A} \models \varphi(x_1, \ldots, x_m)[b_1, \ldots, b_m]$ which is equivalent to saying that $\mathfrak{A} \models \varphi(x_1, \ldots, x_m, x_{m+1})[b_1, \ldots, b_m, c]$.
- $\varphi \equiv Qx_i(\psi)$ where Q is either $\exists$ or $\forall$ and $1 \leq i \leq m$. In this case x_i is not a free variable at all so
  $\mathfrak{A} \models \varphi(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m, x_{m+1})[a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_m, c]$. But then 3.6 implies that
  $\mathfrak{A} \models \varphi(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m, x_{m+1})[b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_m, c]$ which is equivalent to saying that $\mathfrak{A} \models \varphi(x_1, \ldots, x_m, x_{m+1})[b_1, \ldots, b_m, c]$.

Since the argument is clearly symmetric, 3.4 is proved. □

Theorem 3.57 There is an FO + sc-IFP* formula (and hence an FO + sc-IFP formula) which defines a linear order on any k-reducible structure given as input. Hence the logic FO + sc-IFP + I captures PTIME on the class of all k-reducible structures.

Proof

Let $\mathfrak{A}$ be a k-reducible structure. We will first give a formula $\theta(x_1, \ldots, x_k, y_1, \ldots, y_k)$ which defines a linear ordering of the k-tuples of $\mathfrak{A}$.

The idea is simple. First, we order the k-equivalence classes of $\mathfrak{A}$ in the manner of Dawar (1993) and Abiteboul and Vianu (1991c). As each k-equivalence class is an automorphism class, we can choose some tuple $\bar{a}$ from the least k-equivalence which still contains unordered elements and make it the successor to the currently largest element. We can look at the result as a new structure $\mathfrak{A}'$ which is $\mathfrak{A}$ along with $\bar{a}$ serving as interpretation to a tuple of new constant symbols. From Proposition 3.56 $\mathfrak{A}'$ is also k-reducible. Therefore the above procedure can be repeated until all the k-tuples are ordered.

Let $lpo(\bar{x}, \bar{y})$ where $|\bar{x}| = |\bar{y}| = k$ be a formula which defines a linear preorder on the set of all k-tuples. More precisely, $\neg lpo(\bar{a}, \bar{b}) \wedge \neg lpo(\bar{b}, \bar{a})$ if and only if $\bar{a}$ and $\bar{b}$ are k-equivalent and lpo induces a linear order on the k-equivalence classes. Then the formula θ below defines an order on the k-tuples of $\mathfrak{A}$:

$$\theta(\bar{x}, \bar{y}) \equiv (\textbf{sc-ifp}^*_{\leqq \bar{x}\bar{y},\ T\bar{y}}\ \psi(\bar{x}, \bar{y}), \varphi(\bar{y}))(\bar{x}, \bar{y})$$

where

$$\psi(\bar{x}, \bar{y}) \equiv [\neg\exists\bar{u}\bar{v}(\bar{u} \leqq \bar{v}) \wedge T\bar{x} \wedge T\bar{y}] \vee [\exists\bar{u}(\bar{u} \leqq \bar{x}) \wedge \neg\exists\bar{v}(\bar{x} \leqq \bar{v}) \wedge T\bar{y}].$$

and

$$\varphi(\bar{y}) \equiv \neg\exists\bar{x}(\bar{x} \leqq \bar{y} \vee \bar{y} \leqq \bar{x}) \wedge \forall\bar{v}[\neg\exists\bar{x}(\bar{x} \leqq \bar{v} \vee \bar{v} \leqq \bar{x}) \Rightarrow (lpo(\bar{y}, \bar{v}) \vee eq(\bar{y}, \bar{v})))$$

where $eq(\bar{x}, \bar{y}) \equiv \neg lpo(\bar{x}, \bar{y}) \wedge \neg lpo(\bar{y}, \bar{x})$.

We can write an FO + sc-IFP-formula $Succ(x, y)$ based on θ which defines an ordering on A. Now we know that any PTIME query q can be defined by an FO + IFP formula φ_q which uses a linear order on the domain. The following FO + sc-IFP + I formula $\theta_q(u_1 \ldots u_m)$ defines any PTIME query q on the class of k-reducible structures.

$$\theta_q(\bar{u}) \equiv \mathbf{I}_{xy,\bar{x},\bar{z}}(Succ(x, y), R_1, \ldots, R_k, u_1 = z_1, \ldots, u_m = z_m;\ \varphi'_q)$$

where $\varphi'_q \equiv \exists! u_1(\ldots(\exists! u_m(E_1u_1 \wedge \ldots \wedge E_mu_m \wedge \varphi_q))\ldots)$. □

CHAPTER 4

CONCLUSION

We've investigated several symmetric choice based extensions of fixed point logic. We first introduced the logics FO + c-IFP, FO + sc-IFP, and FO + ssc-IFP. We showed that these logics exploit the symmetry of the input structure and define more queries then FO + IFP. The logic FO + sc-IFP has the desirable property that all queries defined in the logic are *semideterministic* and all Boolean queries are deterministic. The logic FO + ssc-IFP also enjoys the property that any formula can be evaluated in PTIME.

We also showed that on rigid structures both these logics coincide with FO + IFP. We introduced the more powerful logics FO + sc-IFP + I and FO + ssc-IFP + I which uses the *reduction operator* to gain more expressive power. In particular, the logic FO + IFP + C is included in FO + ssc-IFP + I.

We then showed some classes of structures on which the logic FO + ssc-IFP + I captures PTIME. One of the classes is a counterexample which shows that FO + IFP + C does not capture PTIME. The other is a counterexample which shows that FO + IFP($\mathbf{Q}$) does not capture PTIME. We concluded that both FO + IFP + C and FO + IFP($\mathbf{Q}$) do not subsume FO + ssc-IFP + I. Finally we introduced the class of k-reducible structures and showed that FO + sc-IFP + I captures PTIME on this class.

It still remains an open question whether FO + sc-IFP + I captures all of PTIME.

REFERENCES

[Abiteboul and Vianu (1991a)] S. Abiteboul and V. Vianu. Non-determinism in logic-based languages. *Annals of Mathematics and Artificial Intelligence* **3**, 151–186, 1991.

[Abiteboul and Vianu (1991b)] S. Abiteboul and V. Vianu. Datalog extensions for database queries and updates. *Journal of Computer and System Sciences* **43**, 62–124, 1991.

[Abiteboul and Vianu (1991c)] S. Abiteboul and V. Vianu. Generic computation and its complexity. In *Proceedings of the 23rd ACM Symposium on Theory of Computing*, pages 209–219, 1991.

[Blass et al. (1985)] A. Blass, Y. Gurevich, and D. Kozen . A Zero-One Law for Logic with a Fixed-Point Operator. *Information and Control* **67**, 70–90, 1985.

[Cai et al. (1992)] J. Cai, M. Fürer, and N. Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica* **12**, 389–410, 1992.

[Compton (1989)] K. J. Compton. 0-1 laws in logic and combinatorics. In I. Rival (ed.), *NATO Advanced Study Institute on Algorithms and Order*, pages 353–383. D. Reidel, 1989.

[Dawar (1993)] A. Dawar. *Feasible Computation through Model Theory*. PhD Thesis, University of Pennsylvania, 1993.

[Ebbinghaus and Flum (1995)] H.-D. Ebbinghaus and J. Flum: *Finite Model Theory*. Springer, 1995.

[Fagin (1974)] R. Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In R. M. Karp (ed.), *Complexity of Computation, SIAM-AMS Proceedings* 7:43–73, 1974.

[Gire and Hoang (1998)] F. Gire and K. Hoang. An Extension of Fixpoint Logic with a Symmetry-Based Choice Construct. *Information and Computation* **144**, 40–65, 1998.

[Gurevich (1988)] Y. Gurevich. Logic and the challenge of computer science. In E. Börger (ed.), *Trends in theoretical computer science*, pages 1–57. Computer Science Press, New York, 1988.

[Gurevich and Shelah (1986)] Y. Gurevich and S. Shelah. Fixed-point extensions of first-order logic. *Annals of Pure and Applied Logic*, **32**, 265–280, 1986.

[Gyssens et al. (1994)] M. Gyssens, J. Van den Bussche, and D. Van Gucht. Expressiveness of efficient semideterministic choice constructs. In *Proceedings of the 21st International Colloquium on Automata, Languages, and Programming (ICALP 94)*, volume 820 of *Lecture Notes in Computer Science*, pages 106–117. Springer, 1994.

[Hella (1996)] L. Hella. Logical Hierarchies in PTIME. *Information and Computation* **129**, 1–19, 1996.

[Immerman (1986)] N. Immerman. Relational queries computable in polynomial time. *Information and Control* **68**, 86–104, 1986.

[Immerman (1987)] N. Immerman. Languages which capture complexity classes. *SIAM Journal of Computing* **16**, 760–778, 1987.

[Immerman and Lander (1990)] N. Immerman and E. Lander. Describing graphs: a first-order approach to graph canonization. In A. Selman (ed.), *Complexity Theory Retrospective*, pages 59–81. Springer, 1990.

[Vardi (1982)] M. Y. Vardi. The complexity of relational query languages. In *Proceedings of the 14th ACM Symposium on Theory of Computing*, pages 137–146, 1982.