Algorithmic Model Theory
SS 2016

Prof. Dr. Erich Grädel and Dr. Wied Pakusa

Mathematische Grundlagen der Informatik
RWTH Aachen

Contents

5 Modal, Inflationary and Partial Fixed Points

In finite model theory, a number of other fixed-point logics, in addition to LFP, play an important role. The structure, expressive power, and algorithmic properties of these logics have been studied intensively, and we review these results in this chapter.

5.1 The Modal μ-Calculus

A fragment of LFP that is of fundamental importance in many areas of computer science (e.g. controller synthesis, hardware verification, and knowledge representation) is the modal μ-calculus (L_μ). It is obtained by adding least and greatest fixed points to propositional modal logic (ML). In this way L_μ relates to ML in the same way as LFP relates to FO.

Definition 5.1. The *modal μ-calculus L_μ* extends ML (including propositional variables $X, Y, \dots$, which can be viewed as monadic second-order variables) by the following rule for building fixed point formulae: If ψ is a formula in L_μ and X is a propositional variable that only occurs positively in ψ, then $\mu X.\psi$ and $\nu X.\psi$ are also L_μ-formulae.

The semantics of these fixed-point formulae is completely analogous to that for LFP. The formula ψ defines on G (with universe V, and with interpretations for other free second-order variables that ψ may have besides X) the monotone operator $F_\psi : \mathcal{P}(V) \to \mathcal{P}(V)$ assigning to every set $X \subseteq V$ the set $\psi^G(X) := \{v \in V : (G, X), v \models \psi\}$. The semantics of fixed-points is defined by

$$G, v \models \mu X.\psi \text{ iff } v \in \text{lfp}(F_\psi)$$
$$G, v \models \nu X.\psi \text{ iff } v \in \text{gfp}(F_\psi).$$

Example 5.2. The formula $\mu X.(\varphi \vee \langle a \rangle X)$ asserts that there exists a path along a-transitions to a node where φ holds.

The formula $\psi := \nu X.\Big((\bigvee_{a\in A} \langle a \rangle \mathit{true}) \wedge (\bigwedge_{a \in A} [a] X) \Big)$ expresses the assertion that the given transition system is deadlock-free. In other words, $G, v \models \psi$ if no path from v in G reaches a dead end (i.e. a node without outgoing transitions).

Finally, the formula $\nu X.\mu Y.\Big(\langle a \rangle \big((\varphi \wedge X) \vee Y \big) \Big)$ says that there exists a path from the current node on which φ holds infinitely often.

The embedding from ML into FO is readily extended to a translation from L_μ into LFP, by inductively replacing formulas of the form $\mu X.\varphi$ by $[\mathrm{lfp}\, Xx.\varphi^*](x)$.

Proposition 5.3. Every formula $\psi \in L_\mu$ is equivalent to a formula $\psi^*(x) \in$ LFP.

Further the argument proving that LFP can be embedded into SO also shows that L_μ is a fragment of MSO.

As for LFP, a fixed μ-calculus formula can be evaluated on a structure $\mathfrak{A}$ in time polynomial in $|\mathfrak{A}|$. The question whether evaluating μ-calculus formulas on a structure when both the formula and the structure are part of the input is in PTIME is a major open problem. On the other hand, it is not difficult to see that the μ-calculus does not suffice to capture PTIME, even in very restricted scenarios such as word structures. Indeed, as L_μ is a fragment of MSO, it can only define *regular languages*, and of course, not all PTIME-languages are regular. However, we shall see in Section 5.5 that there is a multidimensional variant of L_μ that captures the *bisimulation-invariant* fragment of PTIME. Before we do this, let us first show that L_μ is itself invariant under bisimulation. To this end, we translate L_μ formulas into formulas of *infinitary modal logic* $ML_{\infty\omega}$, similar to the embedding of LFP into $L_{\infty\omega}$.

5.1.1 Infinitary Modal Logic and Bisimulation Invariance

Infinitary modal logic extends ML in an analogous way as how infinitary first-order logic extends FO.

Definition 5.4. Let $\kappa \in \mathrm{Cn}^\infty$ be an infinite cardinal number. The *infinitary logic* $ML_{\kappa\omega}$ is inductively defined as follows.

- Predicates P_i are in $ML_{\kappa\omega}$.
- If $\varphi \in ML_{\kappa\omega}$, then also $\neg\varphi$, $\Box\varphi$, $\Diamond\varphi \in ML_{\kappa\omega}$.
- If $\Phi \subseteq ML_{\kappa\omega}$ is a set of formulae with $|\Phi| < \kappa$, then $\bigvee \Phi$, $\bigwedge \Phi \in ML_{\kappa\omega}$.

Further, we write $ML_{\infty\omega}$ to denote $\bigcup_{\kappa \in \mathrm{Cn}^\infty} ML_{\kappa\omega}$.

The semantics of $ML_{\infty\omega}$ on Kripke structures is defined analogously to the semantics of ML, with the following obvious extension for the case of infinite disjuntions and conjunctions.

- $\mathcal{K}, v \models \bigwedge \Phi$ if and only if $\mathcal{K}, v \models \varphi$ for all $\varphi \in \Phi$.
- $\mathcal{K}, v \models \bigvee \Phi$ if and only if there exists a $\varphi \in \Phi$ such that $\mathcal{K}, v \models \varphi$.

The same proof that shows invariance of ML under bisimulation works for $ML_{\infty\omega}$, because the introduction of infinite conjunctions and disjunctions does not interfere with the arguments in the proof at all.

Theorem 5.5. The logic $ML_{\infty\omega}$ is invariant under bisimulation, i.e. if $\varphi \in ML_{\infty\omega}$ is a formula and $\mathcal{K}, v \sim \mathcal{K}', v'$ are two bisimilar Kripke structures, then

$$\mathcal{K}, v \models \varphi \text{ iff } \mathcal{K}', v' \models \varphi.$$

Similarly, the proof of Theorem 5.6 can be adapted to give a translation from L_μ formulas to $ML_{\infty\omega}$, as stated below.

Theorem 5.6. Let $\kappa \in \mathrm{Cn}^\infty$. For each formula $\varphi \in L_\mu$ there exists a formula $\widehat{\varphi} \in ML_{\kappa\omega}$ such that for all transition systems $\mathcal{K}$ with $|\mathcal{K}| < \kappa$ and all $v \in \mathcal{K}$, we have $\mathcal{K}, v \models \varphi$ if and only if $\mathcal{K}, v \models \widehat{\varphi}$.

Combining these two theorems, we get bisimulation invariance of L_μ.

Corollary 5.7. The logic L_μ is invariant under bisimulation.

5.2 Inflationary Fixed-Point Logic

LFP is only one instance of a logic with an explicit operator for forming fixed points. A number of other fixed-point extensions of first-order logic (or fragments of it) have been extensively studied in finite model theory. These include inflationary, partial, non-deterministic, and alternating fixed-point logics. All of these have in common that they allow the construction of fixed points of operators that are not necessarily monotone.

An operator $G : \mathcal{P}(B) \to \mathcal{P}(B)$ is called *inflationary* if $G(X) \supseteq X$ for all $X \subseteq B$. With any operator F one can associate an inflationary operator G, defined by $G(X) := X \cup F(X)$. In particular, inflationary operators are inductive, so iterating G yields a fixed point, called the *inflationary fixed point* of F.

To be more precise, the inflationary fixed-point of any operator $F\colon \mathcal{P}(B) \to \mathcal{P}(B)$ is defined as the limit of the increasing sequence of sets (R^α) defined as $R^0 := \emptyset$, $R^{\alpha+1} := R^\alpha \cup F(R^\alpha)$, and $R^\lambda := \bigcup_{\alpha<\lambda} R^\alpha$ for limit ordinals λ. The *deflationary fixed point* of F is constructed in the dual way starting with B as the initial stage and taking intersections at successor and limit ordinals.

Remark 5.8.

(1) Monotone operators need not be inflationary, and inflationary operators need not be monotone.
(2) An inflationary operator need not have a least fixed point.
(3) The least fixed point of an inflationary operator (if it exists) may be different from the inductive fixed point.
(4) However, if F is a monotone operator, then its inflationary fixed point and its least fixed point coincide.

The logic IFP is defined with a syntax similar to that of LFP, but without the requirement that the fixed-point variable occurs only positively in the formula defining the operator, and with semantics given by the associated inflationary operator.

Definition 5.9. IFP is the extension of first-order logic by the following fixed-point formation rules. For every formula $\psi(R, \overline{x})$, every tuple $\overline{x}$ of variables, and every tuple $\overline{t}$ of terms (such that the lengths of $\overline{x}$ and $\overline{t}$ match the arity of R), we can build formulas $[\mathrm{ifp}\, R\overline{x} \,.\, \psi](\overline{t})$ and $[\mathrm{dfp}\, R\overline{x} \,.\, \psi](\overline{t})$.

Semantics. For a given structure $\mathfrak{A}$, we have that $\mathfrak{A} \models [\mathrm{ifp}\, R\overline{x} \,.\, \psi](\overline{t})$ and

$\mathfrak{A} \models [\text{dfp}\, R\bar{x} \,.\, \psi](\bar{t})$ if $\bar{t}^{\mathfrak{A}}$ is contained in the inflationary and deflationary fixed point of F_ψ, respectively.

By the last item of Remark 5.8, least and inflationary inductions are equivalent for positive formulae, and hence IFP is at least as expressive as LFP. On finite structures, inflationary inductions reach the fixed point after a polynomial number of iterations, hence every IFP-definable class of finite structures is decidable in polynomial time.

Proposition 5.10. IFP captures PTIME on ordered finite structures.

5.2.1 Least Versus Inflationary Fixed-Points

As both logics capture PTIME, IFP and LFP are equivalent on ordered finite structures. What about unordered structures? It was shown by Gurevich and Shelah that the equivalence of IFP and LFP holds on all finite structures. Their proof does not work on infinite structures, and indeed there are some important aspects in which least and inflationary inductions behave differently. For instance, there are first-order operators (on arithmetic, say) whose inflationary fixed point is not definable as the least fixed point of a first-order operator. Further, the alternation hierarchy in LFP is strict, whereas IFP has a positive normal form (see Proposition 5.17 below). Hence it was conjectured by many that IFP might be more powerful than LFP. However, Kreutzer showed recently that IFP is equivalent to LFP on arbitrary structures. Both proofs, by Gurevich and Shelah and by Kreutzer, rely on constructions showing that the *stage comparison relations* of inflationary inductions are definable by lfp inductions.

Definition 5.11. For every inductive operator $F : \mathcal{P}(B) \to \mathcal{P}(B)$, with stages X^α and an inductive fixed point X^∞, the *F-rank* of an element $b \in B$ is $|b|_F := \min\{\alpha : b \in X^\alpha\}$ if $b \in X^\infty$, and $|b|_F = \infty$ otherwise. The *stage comparison relations* of G are defined by

$$\begin{aligned} a \leq_F b \quad &\text{iff} \quad |a|_F \leq |b|_F < \infty \\ a \prec_F b \quad &\text{iff} \quad |a|_F < |b|_F. \end{aligned}$$

Given a formula $\varphi(R, \overline{x})$, we write $\leq_\varphi$ and $\prec_\varphi$ for the stage comparison relations defined by the operator F_φ (assuming that it is indeed inductive), and $\leq_\varphi^{\text{inf}}$ and $\prec_\varphi^{\text{inf}}$ for the stage comparison relations of the associated inflationary operator $G_\varphi : R \mapsto R \cup \{\overline{a} : \mathfrak{A} \models \varphi(R, \overline{a})\}$.

Example 5.12. For the formula $\varphi(T, x, y) := Exy \vee \exists z(Exz \wedge Tzy)$ the relation $\prec_\varphi$ on a graph (V, E) is distance comparison:

$$(a, b) \prec_\varphi (c, d) \text{ iff } \operatorname{dist}(a, b) < \operatorname{dist}(c, d).$$

Stage comparison theorems are results about the definability of stage comparison relations. For instance, Moschovakis proved that the stage comparison relations $\leq_\varphi$ and $\prec_\varphi$ of any positive first-order formula φ are definable by a simultaneous induction over positive first-order formulae. For results on the equivalence of IFP and LFP one needs a stage comparison theorem for IFP inductions.

We first observe that the stage comparison relations for IFP inductions are easily definable in IFP. For any formula $\varphi(T, \bar{x})$ with free variables $\bar{x}$ and free occuring predicate T, the stage comparison relation $\prec_\varphi^{\text{inf}}$ is defined by the formula

$$\psi(\overline{x}'\overline{y}') = [\text{ifp}\, \overline{w} \prec \overline{z} \,.\, \varphi[T\overline{u}/\overline{u} \prec \overline{w}](\overline{w}) \wedge \neg\varphi[T\overline{u}/\overline{u} \prec \overline{z}](\overline{z})](\overline{x}', \overline{y}').$$

Here we syntactically substitute $T, \overline{u}$ by $\overline{u} \prec \overline{w}$ in $\varphi(T\overline{x})$ and, additionally, free variables again by $\overline{w}$. (Note that $\overline{u}$ may contain free variables.) In $\neg\varphi(T, \overline{x})$, we substitute $T, \overline{u}$ by $\overline{u} \prec \overline{z}$ and, additionally, free variables again by $\overline{z}$. Thus free variables become parameter variables of the fixed-point. Now, for the first iteration, T_0 is empty as well as $\prec_0$, so the formula $\varphi(T_0, \overline{w})$ is satisfied by the same $\overline{a}$ as $\varphi(\prec_0, \overline{w})$. So in the first interation, the first components of $\prec_1$ contain the same elements as T_1. The second components of $\prec_1$ contain all other elements. In general, in the i-th iteration, $\prec_i$ consists of pairs $(\overline{a}, \overline{b})$ such that $\overline{a} \in T_i$ and $\overline{b} \notin T_i$. In the next step, precisely those $\overline{a}$ satisfy $\varphi[T\overline{u}/\overline{u} \prec \overline{w}](\prec_i)$ that satisfy $\varphi(T_i)$ (instead of $\varphi[T, \overline{u}]$ we now have $\varphi[\overline{u} \prec \overline{w}]$, i.e. $T\overline{a}$ holds if and only if $\overline{u} \prec \overline{a}$ holds if and only if $\overline{a}$ has come to T in the previous steps). So those $\overline{b}$ that do not satisfy $\varphi[T\overline{u}/\overline{u} \prec \overline{w}](\prec_i)$, satisfy $\neg\varphi[T\overline{u}/\overline{u} \prec \overline{w}](\prec_i)$.

Summing up, pairs $\overline{a}, \overline{b}$ are included to $\prec_{i+1}$ if and only if $\overline{a}$ is included into T_{i+1}, but not earlier, and $\overline{b}$ is not in T_{i+1}.

However, what we need to show is that the stage comparison relation for IFP inductions is in fact LFP-definable.

Theorem 5.13 (Inflationary Stage Comparison)**.** For any formula $\varphi(R, \overline{x})$ in FO or LFP, the stage comparison relation $\prec_{\varphi}^{\mathrm{inf}}$ is definable in LFP. On finite structures, it is even definable in positive LFP.

From this result, the equivalence of LFP and IFP follows easily.

Theorem 5.14 (Kreutzer)**.** For every IFP-formula, there is an equivalent LFP-formula.

Proof. For any formula $\varphi(R, \overline{x})$,

$$[\mathrm{ifp}\, R\overline{x} \,.\, \varphi](\overline{x}) \equiv \varphi(\{\overline{y} : \overline{y} \prec_{\varphi}^{\mathrm{inf}} \overline{x}\}, \overline{x}).$$

This holds because, by definition, an inductive fixed-point can only increase. Thus a tuple is added to it if and only if there is a stage, at which the relation R contains all previously added elements (thus $R = \{\overline{y} : \overline{y} \prec_{\varphi}^{\mathrm{inf}} \overline{x}\}$), and at that stage $\varphi(R, \overline{x})$ holds. Due to Theorem 5.13, the relation $\{\overline{y} : \overline{y} \prec_{\varphi}^{\mathrm{inf}} \overline{x}\}$ is definable in LFP, so the statement follows directly. Q.E.D.

Positive LFP. While LFP and the modal μ-calculus allow arbitrary nesting of least and greatest fixed points, and arbitrary interleaving of fixed points with Boolean operations and quantifiers, we can also ask about their more restricted forms. Let LFP_1 (sometimes also called positive LFP) be the extension of first-order logic that is obtained by taking least fixed points of positive first-order formulae (without parameters) and closing them under disjunction, conjunction, and existential and universal quantification, but *not* under negation. LFP_1 can be conveniently characterized in terms of simultaneous least fixed points, defined in the next chapter.

Theorem 5.15. A relation is definable in LFP_1 if and only if it is definable by a formula of the form $[\mathrm{lfp}\, R : S](\overline{x})$, where S is a system of update rules $R_i\overline{x} := \varphi_i(\overline{R}, \overline{x})$ with first-order formulae φ_i. Moreover, we can require, without diminishing the expressive power, that each of the formulae φ_i in the system is either a purely existential formula or a purely universal formula.

One interesting consequence of the stage comparison theorems is that on finite structures, greatest fixed points (i.e. negations of least fixed points) can be expressed in positive LFP. This gives a normal form for LFP and IFP.

Theorem 5.16 (Immerman)**.** On finite structures, every LFP-formula (and hence also every IFP-formula) is equivalent to a formula in LFP_1.

This result fails on infinite structures. On infinite structures, there exist LFP formulae that are not equivalent to positive formulae, and in fact the alternation hierarchy of least and greatest fixed points is strict. This is not the case for IFP.

Proposition 5.17. It can be proven that every IFP-formula is equivalent to one that uses ifp-operators only positively.

Proof. Assume that structures contain at least two elements and that a constant 0 is available. Then a formula $\neg[\text{ifp } R\overline{x} \,.\, \psi(R, \overline{x})]$ is equivalent to an inflationary induction on a predicate $T\overline{x}y$ which, for $y \neq 0$, simulates the induction defined by ψ, checks whether the fixed point has been reached, and then makes atoms $T\overline{x}0$ true if $\overline{x}$ is not contained in the fixed point. Q.E.D.

In finite model theory, owing to the Gurevich-Shelah Theorem, the two logics LFP and IFP have often been used interchangeably. However, there are significant differences that are sometimes overlooked. Despite the equivalence of IFP and LFP, inflationary inductions are a more powerful concept than monotone inductions. The translation from IFP-formulae to equivalent LFP-formulae can make the formulae much more complicated, requires an increase in the arity of fixed-point variables and, in the case of infinite structures, introduces alternations between least and greatest fixed points. Therefore it is often more convenient to use inflationary inductions in explicit constructions, the advantage being that one is not restricted to inductions over positive formulae. For an example, see the proof of Theorem 5.29 below. Furthermore, IFP is more robust, in the sense that inflationary fixed points remain well defined even when other non-monotone operators (e.g. generalized quantifiers) are added to the language.

5.3 Simultaneous Inductions

A more general variant of LFP permits simultaneous inductions over several formulae. A simultaneous induction is based on a system of operators of the form

$$\begin{aligned} F_1 : \mathcal{P}(B_1) \times \cdots \times \mathcal{P}(B_m) &\to \mathcal{P}(B_1) \\ &\vdots \\ F_m : \mathcal{P}(B_1) \times \cdots \times \mathcal{P}(B_m) &\to \mathcal{P}(B_m), \end{aligned}$$

forming together an operator

$$F = (F_1, \dots, F_m) : \mathcal{P}(B_1) \times \cdots \times \mathcal{P}(B_m) \to \mathcal{P}(B_1) \times \cdots \times \mathcal{P}(B_m).$$

Inclusion on the product lattice $\mathcal{P}(B_1) \times \cdots \times \mathcal{P}(B_m)$ is componentwise. Accordingly, F is monotone if, whenever $X_i \subseteq Y_i$ for all i, then also $F_i(\overline{X}) \subseteq F_i(\overline{Y})$ for all i.

Everything said above about least and greatest fixed points carries over to simultaneous induction. In particular, a monotone operator F has a least fixed point $\mathrm{lfp}(F)$ which can be constructed inductively, starting with $\overline{X}^0 = (\emptyset, \dots, \emptyset)$ and iterating F until a fixed point $\overline{X}^\infty$ is reached.

One can extend the logic LFP by a simultaneous fixed point formation rule.

Definition 5.18. *Simultaneous least fixed-point logic*, denoted by S-LFP, is the extension of first-order logic by the following rule.

Syntax. Let $\psi_1(\overline{R}, \overline{x}_1), \dots, \psi_m(\overline{R}, \overline{x}_m)$ be formulae of vocabulary $\tau \cup \{R_1, \dots, R_m\}$, with only positive occurrences of $R_1, \dots, R_m$, and, for each $i \le m$, let $\overline{x}_i$ be a sequence of variables matching the arity of R_i. Then

$$S := \begin{cases} R_1\overline{x}_1 & := \quad \psi_1 \\ & \vdots \\ R_m\overline{x}_m & := \quad \psi_m \end{cases}$$

is a *system of update rules*, which is used to build formulae $[\text{lfp}\ R_i : S](\overline{t})$ and $[\text{gfp}\ R_i : S](\overline{t})$ (for any tuple $\overline{t}$ of terms whose length matches the arity of R_i).

Semantics. On each structure $\mathfrak{A}$, S defines a monotone operator $S^{\mathfrak{A}} = (S_1, \dots, S_m)$ mapping tuples $\overline{R} = (R_1, \dots, R_m)$ of relations on A to $S^{\mathfrak{A}}(\overline{R}) = (S_1(\overline{R}), \dots, S_m(\overline{R}))$ where $S_i(\overline{R}) := \{\overline{a} : (\mathfrak{A}, \overline{R}) \models \psi_i(\overline{R}, \overline{a})\}$. As the operator is monotone, it has a least fixed point $\text{lfp}(S^{\mathfrak{A}}) = (R_1^\infty, \dots, R_m^\infty)$. Now $\mathfrak{A} \models [\text{lfp}\, R_i : S](\overline{a})$ if $\overline{a} \in R_i^\infty$. Similarly for greatest fixed points.

As in the case of LFP, one can also extend IFP and PFP (defined in the next section) by simultaneous inductions over several formulae. In all of these cases, simultaneous fixed-point logics S-LFP, S-IFP and S-PFP are not more expressive than their simple variants. This can be proven easily by taking a fixed-point over a relation R with bigger arity, e.g. one higher than the maximum arity of $R_1, \dots, R_m$. The atoms $R_i(\overline{x})$ can then be replaced by $R(c_i, \overline{x})$ for chosen m constants $c_1, \dots, c_m$. The fixed-point of R is then sufficient to describe the simultaneous fixed-point of S, yielding the following.

Theorem 5.19. For every formula $\varphi \in$ S-LFP ($\varphi \in$ S-IFP,S-PFP) there exists an equivalent formula $\varphi \in$ LFP ($\varphi \in$ IFP,PFP).

5.4 Partial Fixed-Point Logic

Another fixed-point logic that is relevant to finite structures is the partial fixed-point logic (PFP). Let $\psi(R, \overline{x})$ be an arbitrary formula defining on a finite structure $\mathfrak{A}$ a (not necessarily monotone) operator $F_\psi : R \mapsto \{\overline{a} : \mathfrak{A} \models \psi(R, \overline{a})\}$, and consider the sequence of its finite stages $R^0 := \emptyset$, $R^{m+1} = F_\psi(R^m)$.

This sequence is not necessarily increasing. Nevertheless, as $\mathfrak{A}$ is finite, the sequence either converges to a fixed point, or reaches a cycle with a period greater than one. We define the *partial fixed point* of F_ψ as the fixed point that is reached in the former case, and as the empty relation otherwise. The logic PFP is obtained by adding to first-order logic the *partial-fixed-point formation rule*, which allows us to build from any formula $\psi(R,\overline{x})$ a formula $[\text{pfp } R\overline{x} \,.\, \psi(R,\overline{x})](\overline{t})$, saying that $\overline{t}$ is contained in the partial fixed point of the operator F_ψ.

Note that if R occurs only positively in ψ, then

$$[\text{lfp } R\overline{x} \,.\, \psi(R,\overline{x})](\overline{t}) \equiv [\text{pfp } R\overline{x} \,.\, \psi(R,\overline{x})](\overline{t}),$$

so we have that LFP $\leq$ PFP. However, PFP seems to be much more powerful than LFP. For instance, while a least-fixed-point induction on finite structures always reaches the fixed point in a polynomial number of iterations, a partial-fixed-point induction may need an exponential number of stages.

Example 5.20. Consider the sequence of stages R^m defined by the formula

$$\psi(R,x) := \Big(Rx \land \exists y(y < x \land \neg Ry)\Big) \lor \Big(\neg Rx \land \forall y(y < x \rightarrow Ry)\Big) \lor \forall y Ry$$

on a finite linear order $(A,<)$. It is easily seen than the fixed point reached by this induction is the set $R = A$, but before this fixed point is reached, the induction goes in lexicographic order through all possible subsets of A. Hence the fixed point is reached at stage $2^n - 1$, where $n = |A|$.

COMPLEXITY. Although a PFP induction on a finite structure may go through exponentially many stages (with respect to the cardinality of the structure), each stage can be represented with polynomial storage space. As first-order formulae can be evaluated efficiently, it follows by a simple induction that PFP-formulae can be evaluated in polynomial space.

Proposition 5.21. For every formula $\psi \in$ PFP, the set of finite models of ψ is in PSPACE; in short: PFP $\subseteq$ PSPACE.

On ordered structures, one can use techniques similar to those used in previous capturing results, to simulate polynomial-space-bounded computation by PFP-formulae.

Theorem 5.22 (Abiteboul, Vianu, and Vardi)**.** On ordered finite structures, PFP captures PSPACE.

Proof. It remains to prove that every class $\mathcal{K}$ of finite ordered structures that is recognizable in PSPACE, can be defined by a PFP-formula.

Let M be a polynomially space-bounded deterministic Turing machine with state set Q and alphabet Σ, recognizing (an encoding of) an ordered structure $(\mathfrak{A}, <)$ if and only if $(\mathfrak{A}, <) \in \mathcal{K}$. Without loss of generality, we can make the following assumptions. For input structures of cardinality n, M requires space less than $n^k - 2$, for some fixed k. For any configuration C of M, let $\mathrm{Next(C)}$ denote its successor configuration. The transition function of M is adjusted so that $\mathrm{Next(C)} = \mathrm{C}$ if, and only if, C is an accepting configuration.

We represent any configuration of M with a current state q, tape inscription $w_1 \cdots w_m$, and head position i, by the word $\#w_1 \cdots w_{i-1}(qw_i)w_{i+1} \cdots w_{m-1}\#$ over the alphabet $\Gamma := \Sigma \cup (Q \times \Sigma) \cup \{\#\}$, where $m = n^k$ and # is merely used as an end marker to make the following description more uniform. When moving from one configuration to the next, Turing machines make only local changes. We can therefore associate with M a function $f : \Gamma^3 \to \Gamma$ such that, for any configuration $C = c_0 \cdots c_m$, the successor configuration $\mathrm{Next(C)} = \mathrm{c}_0' \cdots \mathrm{c}_\mathrm{m}'$ is determined by the rules

$$c_0' = c_m' = \# \quad \text{and} \quad c_i' = f(c_{i-1}, c_i, c_{i+1}) \text{ for } 1 \leq i \leq m-1.$$

Recall that we encode structures so that there exist first-order formulae $\beta_\sigma(\overline{y})$ such that $(\mathfrak{A}, <) \models \beta_\sigma(\overline{a})$ if and only the $\overline{a}$th symbol of the input configuration of M for input code$(()\mathfrak{A}, <)$ is σ. We now represent any configuration C in the computation of M by a tuple $\overline{\mathrm{C}} = (C_\sigma)_{\sigma \in \Gamma}$ of k-ary relations, where

$$C_\sigma := \{\overline{a} : \text{ the } \overline{a}\text{-th symbol of } C \text{ is } \sigma\}.$$

The configuration at time t is the stage $t+1$ of a simultaneous pfp induction on $(\mathfrak{A}, <)$, defined by the rules

$$C_{\#}\overline{y} := \forall z(\overline{y} \leq \overline{z}) \vee \forall \overline{z}(\overline{z} \leq \overline{y})$$

and, for all $\sigma \in \Gamma - \{\#\}$,

$$C_\sigma \overline{y} := \Big(\beta_\sigma(\overline{y}) \wedge \bigwedge_{\gamma \in \Gamma} \forall \overline{x} \neg C_\gamma \overline{x}\Big) \vee \\ \exists \overline{x} \exists \overline{z} \Big(\overline{x} + 1 = \overline{y} \wedge \overline{y} + 1 = \overline{z} \wedge \bigvee_{f(\alpha,\beta,\gamma)=\sigma} C_\alpha \overline{x} \wedge C_\beta \overline{y} \wedge C_\gamma \overline{z})\Big)$$

The first rule just says that each stage represents a word starting and ending with #. The other rules ensure that (1) if the given sequence $\overline{C}$ contains only empty relations (i.e. if we are at stage 0), then the next stage represents the input configuration, and (2) if the given sequence represents a configuration, then the following stage represents its successor configuration.

By our convention, M accepts its input if and only the sequence of configurations becomes stationary (i.e. reaches a fixed point). Hence M accepts code$(()\mathfrak{A}, <)$ if and only if the relations defined by the simultaneous pfp induction on $\mathfrak{A}$ of the rules described above are non-empty. Hence $\mathcal{K}$ is PFP-definable. Q.E.D.

5.4.1 Least Versus Partial Fixed-Point Logic

From the capturing results for PTIME and PSPACE we immediately obtain the result that PTIME = PSPACE if, and only if, LFP = PFP on ordered finite structures. The natural question arises of whether LFP and PFP can be separated on the domain of all finite structures. For a number of logics, separation results on arbitrary finite structures can be established by relatively simple methods, even if the corresponding separation on ordered structures would solve a major open problem in complexity theory. For instance, we have proved by quite a simple argument that DTC $\subsetneq$ TC, and it is also not very difficult to show that TC $\subsetneq$ LFP (indeed, TC is contained in stratified Datalog, which is also strictly contained in LFP). Further, it is trivial that LFP is less expressive than Σ_1^1 on all finite structures. However the situation is different for LFP vs. PFP.

Theorem 5.23 (Abiteboul and Vianu)**.** LFP and PFP are equivalent on finite structures if, and only if, PTIME = PSPACE.

5.5 Capturing PTIME up to Bisimulation

In mathematics, we consider isomorphic structures as identical. Indeed, it almost goes without saying that relevant mathematical notions do not distinguish between isomorphic objects. As classical algorithmic devices work on ordered *representations of structures* rather than the structures themselves, our capturing results rely on an ability to reason about canonical ordered representations of isomorphism classes of finite structures.

However, in many application domains of logic, structures are distinguished only up to equivalences coarser than isomorphism. Perhaps the best-known example is the modelling of the computational behaviour of (concurrent) programs by transition systems. The meaning of a program is usually not captured by a unique transition system. Rather, transition systems are distinguished only up to appropriate notions of behavioural equivalence, the most important of these being *bisimulation*.

In such a context, the idea of a logic capturing PTIME gets a new twist. One would like to express in a logic precisely those properties of structures that are

(1) decidable in polynomial time, and
(2) invariant under the notion of equivalence being studied.

A class S of rooted transition systems or Kripke structures is *invariant under bisimulation* if, whenever $\mathcal{K}, v \in S$ and $\mathcal{K}, v \sim \mathcal{K}', v'$, then also $\mathcal{K}', v' \in S$. We say that a class S of finite rooted transition systems is in *bisimulation-invariant PTIME* if it is invariant under bisimulation, and if there exists a polynomial-time algorithm deciding whether a given pair $\mathcal{K}, v$ belongs to S. A logic L is invariant under bisimulation if all L-definable properties of rooted transition systems are.

Clearly, $L_\mu \subseteq$ bisimulation-invariant PTIME. However, as pointed out in Section 5.1, L_μ is far too weak to *capture* this class, mainly because it is essentially a monadic logic. Instead, we have to consider a *multidimensional* variant L_μ^ω of L_μ.

But before we define this logic, we should explain the main technical step, which relies on definable canonization, but of course with respect to bisimulation rather than isomorphism. For simplicity of notation, we consider only Kripke structures with a single transition relation E. The extension to the case of several transition relations E_a is straightforward.

With a rooted Kripke structure $\mathcal{K} = (V, E, (P_b)_{b \in B}), u$, we associate a new transition system

$$\mathcal{K}_u^\sim := (V_u^\sim, E^\sim, (P_b^\sim)_{b \in B}),$$

where $V_u^\sim$ is the set of all $\sim$-equivalence classes $[v]$ of nodes $v \in V$ that are reachable from u. More formally, let $[v]$ denote the bisimulation equivalence class of a node $v \in V$. Then

$$\begin{aligned} V_u^\sim &:= \{[v] : \text{there is a path in } G \text{ from } u \text{ to } v\} \\ P_b^\sim &:= \{[v] \in V_u^\sim : v \in P_b\} \\ E^\sim &:= \{([v],[w]) : (v,w) \in E\}. \end{aligned}$$

The pair $\mathcal{K}_u^\sim, [u]$ is, up to isomorphism, a *canonical representant* of the bisimulation equivalence class of $\mathcal{K}, u$. To see this one can prove that (1) $(\mathcal{K}, u) \sim (\mathcal{K}_u^\sim, [u])$, and (2) if $(\mathcal{K}, u) \sim (\mathcal{G}, v)$, then $(\mathcal{K}_u^\sim, [u]) \cong (\mathcal{G}_v^\sim, [v])$.

It follows that a class S of rooted transition systems is bisimulation-invariant if and only if $S = \{(\mathcal{K}, u) : (\mathcal{K}_u^\sim, [u]) \in S\}$. Let $\mathcal{CR}^\sim$ be the domain of canonical representants of finite transition systems, i.e.

$$\mathcal{CR}^\sim := \{\mathcal{K}, u \mid (\mathcal{K}_u^\sim, [u]) \cong (\mathcal{K}, u)\}.$$

Proposition 5.24. $\mathcal{CR}^\sim$ admits LFP-definable linear orderings, i.e. for every vocabulary $\tau = \{E\} \cup \{P_b : b \in B\}$, there exists a formula $\psi(x, y) \in \mathrm{LFP}(\tau)$ which defines a linear order on every transition system in $\mathcal{CR}^\sim(\tau)$.

Proof. Recall that bisimulation equivalence on a transition system is a greatest fixed point. Its complement, bisimulation inequivalence, is a least fixed point, which is the limit of an increasing sequence $\not\sim_i$ defined as follows: $u \not\sim_0 v$ if u and v do not have the same atomic type, i.e. if there exists some b such that one of the nodes u, v has the property P_b and the other does not. Further, $u \not\sim_{i+1} v$ if the sets of $\sim_i$-classes that are reachable in one step from u and v are different. The idea is to refine this inductive process, by defining relations $\prec_i$ that order the $\sim_i$-classes. On the transition system itself, these relations are pre-orders. The inductive limit $\prec$ of the pre-orders $\prec_i$ defines a linear order of the bisimulation equivalence classes. But in transition systems in $\mathcal{CR}^{\sim}$, bisimulation classes have only one element, so $\prec$ actually defines a linear order on the set of nodes.

To make this precise, we choose an order on B and define $\prec_0$ by enumerating the $2^{|B|}$ atomic types with respect to the propositions P_b, i.e.

$$x \prec_0 y := \bigvee_{b \in B} \Big(\neg P_b x \wedge P_b y \wedge \bigwedge_{b' < b} P_{b'} x \leftrightarrow P_{b'} y \Big).$$

In other words, there is some b such that P_b separates x from y and for the least such b, P_b holds on y and not on x.

In what follows, $x \sim_i y$ can formally be taken as an abbreviation for $\neg(x \prec_i y \vee y \prec_i x)$, and similarly for $x \sim y$. We define $x \prec_{i+1} y$ by the condition that either $x \prec_i y$, or $x \sim_i y$ and the set of $\sim_i$-classes reachable from x is lexicographically smaller than the set of $\sim_i$-classes reachable from y. Note that this inductive definition of $\prec$ is not monotone, so it cannot be directly captured by an LFP-formula. However, as we know that LFP $\equiv$ IFP, we can use an IFP-formula instead. Explicitly, $\prec$ is defined by $[\mathrm{ifp}\, x \prec y \,.\, \psi(\prec, x, y)](x, y)$, where

$$\begin{aligned}\psi(\prec, x, y) := x \prec_0 y \vee \Big(& x \sim y \wedge \\ & (\exists y' \,.\, Eyy')\Big((\forall x' \,.\, Exx')x' \not\sim y' \wedge \\ & \qquad (\forall z . z \prec y')\big(\exists x''(Exx'' \wedge x'' \sim z) \leftrightarrow \\ & \qquad\qquad \exists y''(Eyy'' \wedge y'' \sim z)\big)\Big)\Big).\end{aligned}$$

Q.E.D.

Corollary 5.25. On the domain $\mathcal{CR}^{\sim}$, LFP captures PTIME.

Since LFP is not invariant under bisimulation, we will strengthen the above result and capture bisimulation-invariant PTIME in terms of a natural logic, the multidimensional μ-calculus L_μ^ω.

Definition 5.26. The syntax of the *k-dimensional μ-calculus* L_μ^k (for transition systems $\mathcal{K} = (V, E, (P_b)_{b\in B})$) is the same as the syntax of the usual μ-calculus L_μ with modal operators $\langle i\rangle$, $[i]$, and $\langle\sigma\rangle$, $[\sigma]$ for every substitution $\sigma : \{1, \dots, k\} \to \{1, \dots, k\}$. Let $S(k)$ be the set of all these substitutions.

The semantics is different, however. A formula ψ of L_μ^k is interpreted on a transition system $\mathcal{K} = (V, E, (P_b)_{b\in B})$ at node v by evaluating it as a formula of L_μ on the modified transition system

$$\mathcal{K}^k = (V^k, (E_i)_{1\le i\le k}, (E_\sigma)_{\sigma\in S(k)}, (P_{b,i})_{b\in B, 1\le i\le k})$$

at node $\underline{v} := (v, v, \dots, v)$. Here $V^k = V \times \cdots \times V$ and

$$\begin{aligned} E_i &:= \{(\overline{v}, \overline{w}) \in V^k \times V^k : (v_i, w_i) \in E \text{ and } v_j = w_j \text{ for } j \neq i\} \\ E_\sigma &:= \{(\overline{v}, \overline{w}) \in V^k \times V^k : w_i = v_{\sigma(i)} \text{ for all } i\} \\ P_{b,i} &:= \{\overline{v} \in V^k : v_i \in P_b\} \end{aligned}$$

That is, $\mathcal{K}, v \models_{L_\mu^k} \psi$ iff $\mathcal{K}^k, (v, \dots, v) \models_{L_\mu} \psi$. The *multidimensional μ-calculus* is $L_\mu^\omega = \bigcup_{k<\omega} L_\mu^k$.

Remark. Instead of evaluating a formula $\psi \in L_\mu^k$ at single nodes v of G, we can also evaluate it at k-tuples of nodes: $\mathcal{K}, \overline{v} \models_{L_\mu^k} \psi$ iff $\mathcal{K}^k, \overline{v} \models_{L_\mu} \psi$.

Example 5.27. Bisimulation is definable in L_μ^2 (in the sense of the remark just made). Let

$$\psi^{\sim} := \nu X \,.\, \big(\bigwedge_{b\in B} (P_{b,1} \leftrightarrow P_{b,2}) \land [1]\langle 2\rangle X \land [2]\langle 1\rangle X\big).$$

For every transition system $\mathcal{K}$, we have that $\mathcal{K}, v_1, v_2 \models \psi^{\sim}$ if, and only if, v_1 and v_2 are bisimilar in $\mathcal{K}$. Further, we have that

$$\mathcal{K}, v \models \mu Y . \langle 2 \rangle (\psi^{\sim} \vee \langle 2 \rangle Y)$$

if, and only if, there exists in $\mathcal{K}$ a point w that is reachable from v (by a path of length ≥ 1) and bisimilar to v.

One can see that L_{μ}^{ω} is invariant under bisimulation (because if $\mathcal{K}, v_i \sim \mathcal{G}, u_i$ for all i then also $\mathcal{K}^k, \overline{v} \sim \mathcal{G}, \overline{u}$) and that L_{μ}^{ω} can be embedded in LFP. This establishes the easy direction of the desired result: $L_{\mu}^{\omega} \subseteq$ bisimulation-invariant PTIME.

For the converse, it suffices to show that LFP and L_{μ}^{ω} are equivalent on the domain $\mathcal{CR}^{\sim}$. Let S be a class of rooted transition systems in bisimulation-invariant PTIME. For any $\mathcal{K}, u$, we have that $\mathcal{K}, u \in S$ if its canonical representant $\mathcal{K}_u^{\sim}, [u] \in S$. If LFP and L_{μ}^{ω} are equivalent on $\mathcal{CR}^{\sim}$, then there exists a formula $\psi \in L_{\mu}^{\omega}$ such that $\mathcal{K}_u^{\sim}, [u] \models \psi$ iff $\mathcal{K}_u^{\sim}, [u] \in S$. By the bisimulation invariance of ψ, it follows that $\mathcal{K}, u \models \psi$ iff $\mathcal{K}, u \in S$.

The *width* of an LFP-formula φ is the maximal number of free variables occuring in a subformula of φ.

Proposition 5.28. On the domain $\mathcal{CR}^{\sim}$, LFP $\leq L_{\mu}^{\omega}$. More precisely, for each formula $\psi(x_1, \dots, x_k) \in$ LFP of width $\leq k$, there exists a formula $\psi^* \in L_{\mu}^{k+1}$ such that for each $\mathcal{K}, u \in \mathcal{CR}^{\sim}$, we have that $\mathcal{K} \models \psi(u, \overline{v})$ iff $\mathcal{K}, u, \overline{v} \models \psi^*$.

Note that although, ultimately, we are interested only in formulae $\psi(x)$ with just one free variable, we need more general formulae, and evaluation of L_{μ}^{k}-formulae over k-tuples of nodes, for the inductive treatment. In all formulae, we shall have at least x_1 as a free variable, and we always interpret x_1 as u (the root of the transition system). We remark that, by an obvious modification of the formula given in Example 5.27, we can express in L_{μ}^{k} the assertion that $x_i \sim x_j$ for any i, j.

Atomic formulae are translated from LFP to L_{μ}^{ω} according to

$$(x_i = x_j)^* := x_i \sim x_j$$
$$(P_b x_i)^* := P_{b,i} \overline{x}$$

$$\begin{aligned}(Ex_ix_j)^* &:= \langle i\rangle x_i \sim x_j \\ (Xx_{\sigma(1)}\cdots x_{\sigma(r)})^* &:= \langle\sigma\rangle X.\end{aligned}$$

Boolean connectives are treated in the obvious way, and *quantifiers* are translated by use of fixed points. To find a witness x_j satisfying a formula ψ, we start at u (i.e. set $x_j = x_1$), and search along transitions (i.e. use the μ-expression for reachability). That is, let $j/1$ be the substitution that maps j to 1 and fixes the other indices, and translate $\exists x_j\psi(\overline{x})$ into

$$\langle j/1\rangle\mu Y\,.\,\psi^* \vee \langle j\rangle Y.$$

Finally, *fixed points* are first brought into normal form so that variables appear in the right order, and then they are translated literally, i.e. $[\text{lfp}\, X\overline{x}\,.\,\psi](\overline{x})$ translates into $\mu X\,.\,\psi^*$.

The proof that the translation has the desired property is a straightforward induction, which we leave as an exercise. Altogether we have established the following result.

Theorem 5.29 (Otto)**.** The multidimensional μ-calculus captures bisimulation-invariant PTIME.

Otto has also established capturing results with respect to other equivalences. For finite structures $\mathfrak{A}, \mathfrak{B}$, we say that $\mathfrak{A} \equiv_k \mathfrak{B}$ if no first-order sentence of width k can distinguish between $\mathfrak{A}$ and $\mathfrak{B}$. Similarly, $\mathfrak{A} \equiv_k^C \mathfrak{B}$ if $\mathfrak{A}$ and $\mathfrak{B}$ are indistinguishable by first-order sentences of width k with counting quantifiers of the form $\exists^{\geq i}x$, for any $i \in \mathbb{N}$.

Theorem 5.30 (Otto)**.** There exist logics that effectively capture $\equiv_2$-invariant PTIME and $\equiv_2^C$-invariant PTIME on the class of all finite structures.