4

Back and Forth Between Logic and Games

Erich Grädel
RWTH Aachen University

Abstract

In this chapter we discuss relationships between logic and games, focusing on first-order logic and fixed-point logics, and on reachability and parity games. We discuss the general notion of model-checking games. While it is easily seen that the semantics of first-order logic can be captured by reachability games, more effort is required to see that parity games are the appropriate games for evaluating formulae from least fixed-point logic and the modal μ-calculus. The algorithmic consequences of this result are discussed. We also explore the reverse relationship between games and logic, namely the question of how winning regions in games are definable in logic. Finally the connections between logic and games are discussed for more complicated scenarios provided by inflationary fixed-point logic and the quantitative μ-calculus.

4.1 Introduction

The idea that logical reasoning can be seen as a dialectic game, where a proponent attempts to convince an opponent of the truth of a proposition is very old. Indeed, it can be traced back to the studies of Zeno, Socrates, and Aristotle on logic and rhetoric. Modern manifestation of this idea are the presentation of the semantics of logical formulae by means of ***model-checking games*** and the algorithmic evaluation of logical statements via the ***synthesis of winning strategies*** in such games.

model-checking games are two-player games played on an arena which is formed as the product of a structure $\mathfrak{A}$ and a formula ψ where one player, called the Verifier, attempts to prove that ψ is true in $\mathfrak{A}$ while the other player, the Falsifier, attempts to refute this. In contrast to common definitions of the meaning of logical formulae which proceed bottom-up, from atomic formulae through the application of logical operators (such as connectives and quantifiers) to more complicated ones, game-theoretic semantics proceed top-down. Starting with a complicated sentence, Verifier and Falsifier try to justify their claims by moves to supposedly simpler assertions. By playing such an evaluation game the two players thus produce a sequence of formulae that ends when they reach an atomic statement, which cannot be simplified further. The Verifier has succeeded to justify her original claim if the atomic formula reached at the end of the sequence of is true, and Falsifier has won if it is false. We thus assume that the truth of atomic statements can be readily determined, for instance by a look-up in a table.

model-checking games permit us to evaluate logical formulae by solving ***algorithmic problems on games*** such as the computation of ***winning regions*** and the construction of ***winning strategies***. For the most common logical systems, such as ***first-order logic*** (FO) or ***propositional modal logic*** (ML), the construction of the associated model-checking games is straightforward, and the games are simple in several senses. First of all, the goals of the players are the simplest conceivable objectives in such games, namely ***reachability objectives***: each player tries to force the play to a terminal position where she has won (like check mate). Secondly, it is the case that in each move, the formula is strictly simplified, so that every play terminates after a number of moves that is bounded by the nesting depth of logical operators in the formula. In particular there are no infinite plays, and this holds no matter whether the structure on which the formula is evaluated is finite or infinite. Finally, in the case of finite game graphs, the winner of such a reachability game can be determined in linear time (with respect to the size of the game graph). Thus, algorithms for solving reachability games can be applied to evaluate first-order formulae, and give us a detailed complexity analysis for the model-checking problem of first-order logic on finite structures.

But life is not always that simple. For expressing properties of finite structures, and for defining combinatorial problems on classes of structures (such as graphs), first-order logic is rather limited. For many tasks arising in computer science there is thus a need of other kinds of logical systems, such as temporal logics, dynamic logics, game logics, transitive closure logics, fixed-point logics and so on, which extend a basic formalism like FO and ML by more powerful operators.

The natural model-checking games for such logics are more complicated than reachability games. In particular, they admit infinite plays. Essential ingredients in the description of such games are the winning conditions for infinite plays. Among the simplest of these are recurrence (or Büchi) conditions, which require that certain good states must occur infinitely often in a play, or eventual safety conditions, which impose that from some point onwards the play must stay outside a bad region. Of special importance for us are ***parity games***. These are games of possibly infinite duration where we assign to each position a natural number, and the winner of an infinite play is determined according to whether the least number seen infinitely often in the play is even or odd. The importance of parity games is due to several reasons.

(1) Many classes of games arising in practical applications admit ***reductions to parity games*** (over larger game graphs). This is the case for games modelling reactive systems, with winning conditions specified in some temporal logic or in monadic second-order logic over infinite paths (S1S), for Muller games, but also for games with partial information appearing in the synthesis of distributed controllers.
(2) Parity games are ***positionally determined***. This means that from every position, one of the two players has a winning strategy whose moves depend only on the current position, not on the history of the play. This property is fundamental for the algorithmic synthesis of winning strategies.
(3) Parity games arise as the model-checking games for ***fixed-point logics*** such as the modal μ-calculus or LFP, the extension of first-order logic by least and greatest fixed-points. Conversely, winning regions of parity games (with a bounded number of priorities) are definable in both LFP and the μ-calculus. Parity games are also of crucial importance in the analysis of structural properties of fixed-point logics.

The last point, the intimate relationship between parity games and fixed-point logic is a central theme of this chapter.

We shall start with an introduction to basic notions on reachability and parity games, explain the notions of winning strategies and winning regions and discuss algorithmic questions related to games. We study connections between logic and games for the special case of reachability games. In particular, we shall present in detail the model-checking games for first-order logic. After that, we introduce logics with least and greatest fixed points, such as LFP and the modal μ-calculus, and explain why parity games are appropriate evaluation games for these logics. We shall also discuss the algorithmic consequences of this result. Then, the reverse relationship between games and logic is explored, namely the question of how winning regions in games are definable in logic. We shall see that for parity games with a bounded number of priorities, winning regions are definable in both LFP and the modal μ-calculus. For parity games with an unbounded number of priorities it is not known whether the winning regions are LFP-definable. We show that this problem is intimately related to the open question of whether parity games are solvable in polynomial time. In the last two sections we shall discuss the relationship between logic and games for more complicated scenarios provided by ***inflationary fixed-point logic*** and the ***quantitative*** ***μ-calculus***. In both cases, we can indeed find generalisations of parity games with a balanced two-way relationship to the associated fixed-point logic. On the one hand, we obtain appropriate evaluation games for all formulae in the logic, and on the other hand, the winning regions in the games are definable in the logic

4.2 Reachability games and parity games

We consider turn-based games where two players move a token through a directed graph, tracing out a finite or infinite path. Such a ***graph game*** is specified by a directed graph $G = (V, E)$, with a partition $V = V_0 \cup V_1$ of the nodes into positions of Player 0 and positions of Player 1. In case $(v, w) \in E$ we call w a successor of v and we denote the set of all successors of v by vE. A ***play*** in $\mathcal{G}$ is a finite or infinite path $v_0 v_1 \ldots$ formed by the two players starting from a given initial position v_0. Whenever the current position v_i belongs to V_0, then Player 0 chooses a successor $v_{i+1} \in v_i E$, if $v_i \in V_1$, then $v_{i+1} \in v_i E$ is selected by Player 1.

For ***reachability games*** we define the winning condition either by saying that Player σ loses at positions $v \in V_\sigma$ where no moves are possible, or by explicitly including the sets T_0, T_1 of winning terminal positions for each player into the description of the game. A play that is not won by any of the two players is called a ***draw***. In reachability games, infinite plays are draws.

It is often convenient to have games without draws, so that Player 1 wins every play that is not won by Player 0, and vice versa. As the complement of a reachability condition is a ***safety*** condition this leads to a ***reachability-safety game***: the winning condition is given by a set $T \subseteq V$; Player 0 wins a play if it reaches T, and Player 1 wins if it remains inside $V \setminus T$.

There is an extensive theory of games with more general winning conditions for infinite plays that are specified either by logical formulae from some logic on infinite sequences such as temporal logic (LTL), first-order logic (FO), or monadic second-order logic (S1S), or by automata-theoretic conditions such as Muller conditions, Streett–Rabin conditions, or parity conditions (see the contributions by Christof Löding and Marcin Jurdziński to this book). In this chapter, only parity conditions will be used.

A ***parity game*** is given by a game graph $\mathcal{G} = (V, V_0, V_1, E)$ together with a ***priority function*** $\Omega : V \to \omega$ assigning to each position a natural number. An infinite play $\pi = v_0 v_1 \ldots$ is won by Player 0 if the least priority appearing infinitely often in π is even, or no priority appears infinitely often (which may only happen if the range of Ω is infinite).

Winning strategies, winning regions, and determinacy. A (deterministic) ***strategy*** for Player σ is a partial function $f : V^* V_\sigma \to V$ that assigns to finite paths through $\mathcal{G}$ ending in a position $v \in V_\sigma$ a successor $w \in vE$. A play $v_0 v_1 \cdots \in V^\omega$ is ***consistent*** with f if, for each initial segment $v_0 \ldots v_i$ with $v_i \in V_\sigma$, we have that $v_{i+1} = f(v_0 \ldots v_i)$. We say that such a strategy f is ***winning*** from position v_0 if every play that starts at v_0 and that is consistent with f is won by Player σ. The ***winning region*** of Player σ, denoted W_σ, is the set of positions from which Player σ has a winning strategy.

A game $\mathcal{G}$, without draws, is called ***determined*** if $W_0 \cup W_1 = V$, i.e., if from each position one of the two players has a winning strategy. For games with draws, it is appropriate to define determinacy in a slightly different way: we call a game with draws determined if from each position, either one of the two players has a winning strategy, or both players have a strategy to achieve at least a draw. To put it differently, this means that from every position $v \in V \setminus W_\sigma$, Player $1 - \sigma$ has a strategy to guarantee that Player σ does not win. It has been known for almost 100 years that chess is determined in this sense, see Zermelo [1913]. However, we still do not know which of the three possibilities holds for the initial position of chess: whether White has a winning strategy, whether Black has one, or whether both players can guarantee a draw.

There is a large class of games that are known to be determined, including all games for which the winning condition is a Borel set (Martin [1975]). One can show (based on the Boolean Prime Ideal Theorem, which is a weak form of the the Axiom of Choice) that non-determined games exist. However, all games considered in this chapter are determined in a strong sense.

Computing winning regions of reachability games. To solve a game algorithmically means to compute the winning regions for the two players. When considering algorithmic problems of this kind, we always assume that game graphs are finite. For reachability games, the winning regions can easily be computed in polynomial time. Denote by W_σ^n the set of positions from which Player σ has a strategy to win the game in at most n moves. Then $W_\sigma^0 = \{v \in V_{1-\sigma} : vE = \emptyset\}$ is the set of winning terminal positions for Player σ, and we can compute the sets W_σ^n inductively by using

$$W_\sigma^{n+1} := W_\sigma^n \cup \{v \in V_0 : vE \cap W_\sigma^n \neq \emptyset\} \cup \{v \in V_1 : vE \subseteq W_\sigma^n\}$$

until $W_\sigma^{n+1} = W_\sigma^n$.

With a more sophisticated algorithm, which is a clever variant of depth-first search, one can actually compute the winning regions of both players in linear time $O(|V| + |E|)$ (see e.g., Grädel [2007]).

Theorem 4.1 *Winning regions of finite reachability games, and hence also reachability-safety games, can be computed in linear time.*

Further, the problem of computing winning regions of reachability games is complete for PTIME (see Greenlaw et al. [1995]).

Positional determinacy and complexity of parity games. Winning strategies can be very complicated objects since they may depend on the entire history of a play. However, for many important games, including reachability, safety, and parity games, it suffices to consider ***positional strategies***, which are strategies that depend only on the current position, not on the history of the play. A game is ***positionally determined***, if it is determined, and each player has a positional winning strategy on her winning region.

The positional determinacy of reachability games – and reachability-safety games – is obvious since the winning condition itself is purely positional. For parity games the positional determinacy is a non-trivial and fundamental result. It has been established independently by Emerson and Jutla [1991] and Mostowski [1991] for parity games with a finite game graph. This is generalised by Zielonka [1998] to infinite game graphs with a finite number of priorities. Finally positional determinacy has been extended by Grädel and Walukiewicz [2006] to parity games with $\mathrm{rng}(\Omega) = \omega$.

Theorem 4.2 *Every parity game is positionally determined.*

In a parity game $\mathcal{G} = (V, V_0, V_1, E, \Omega)$, a positional strategy for Player σ, defined on $W \subseteq V$, can be represented by a subgraph $H = (W, S) \subseteq (V, E)$ such that there is precisely one outgoing S-edge from each node $v \in V_\sigma \cap W$ and $vS = vE$ for each node $v \in V_{1-\sigma} \cap W$. On a finite game graph, such a strategy is winning on W if, and only if, the least priority on every cycle in (W, S) has the same parity as σ.

Hence, given a finite parity game $\mathcal{G}$ and a positional strategy (W, S) it can be decided in polynomial time, whether the strategy is winning on W. To decide winning regions we can therefore just guess winning strategies, and verify them in polynomial time.

Corollary 4.3 *Winning regions of parity games (on finite game graphs) can be decided in* NP $\cap$ Co-NP.

In fact, Jurdziński [1998] proved that the problem is in UP $\cap$ Co-UP, where UP denotes the class of NP-problems with unique witnesses. The best known deterministic algorithm has complexity $n^{O(\sqrt{n})})$ (Jurdziński et al. [2006]). For parity games with a number d of priorities the progress measure lifting algorithm by Jurdziński [2000] computes winning regions in time $O(dm \cdot (2n/(d/2))^{d/2}) = O(n^{d/2+O(1)})$, where m is the number of edges, giving a polynomial-time algorithm when d is bounded. The two approaches can be combined to achieve a worst-case running time of $O(n^{n/3+O(1)})$ for solving parity games with d priorities. These, and other, algorithms, are explained in detail in Jurdziński's contribution to this book.

4.3 Reachability games and logic

We now discuss connections between logic and games for the special case of reachability games. We assume that the reader is familiar with first-order logic.

(1) Computing winning regions of reachability games is equivalent, under very simple reductions, to computing minimal models for propositional Horn formulae.
(2) The model-checking games for first-order logic are reachability games.

We will then discuss the definability problem for winning regions of reachability games and see that more powerful formalisms than first-order logic are needed.

4.3.1 Games and Horn formulae

Recall that a ***propositional Horn formula*** is a conjunction of implication clauses of the form $Z \leftarrow X_1 \wedge \cdots \wedge X_k$ where $X_1, \ldots X_k$ are propositional variables, forming the ***body*** of the clause, and Z, the ***head*** of the clause, is either also a propositional variable, or the constant 0. Notice that the body of the clause can also be empty, in which case the clause takes the form

$Z \leftarrow 1$. (Indeed if a Horn formula contains no clause of this form, then it is trivially satisfiable by setting all variables to false.)

It is well known that SAT-HORN, the satisfiability problem for propositional Horn formulae, is PTIME-complete (see Greenlaw et al. [1995]) and solvable in linear time (Dowling and Gallier [1984], Itai and Makowsky [1987]). Hence its computational properties are very similar to those of reachability games. Actually there is a simple way of going back and forth between solving reachability games and finding satisfying assignments for Horn formulae, so that the two problems are solved by essentially the same algorithms.

From reachability games to Horn formulae: Given a finite game graph $\mathcal{G} = (V, V_0, V_1, E)$, we can construct in linear time a propositional Horn formula $\psi_{\mathcal{G}}$ consisting of the clauses $u \leftarrow v$ for all edges $(u, v) \in E$ with $u \in V_0$, and the clauses $u \leftarrow v_1 \wedge \cdots \wedge v_m$ for all nodes $u \in V_1$, where $uE = \{v_1, \ldots, v_m\}$. It is easy to see that the winning region W_0 for Player 0 in $\mathcal{G}$ coincides with the minimal model for $\psi_{\mathcal{G}}$. Hence $v \in W_0$ if the Horn formula $\psi_{\mathcal{G}} \wedge (0 \leftarrow v)$ is unsatisfiable.

From Horn formulae to reachability games: With a Horn formula $\psi = \bigwedge_{i \in I} C_i$ with propositional variables $X_1, \ldots, X_n$ and Horn clauses C_i of the form $Z_i \leftarrow X_{i_1} \wedge \cdots X_{i_m}$ we associate a game $\mathcal{G}_\psi$ as follows. The positions of Player 0 are the initial position 0 and the propositional variables $X_1, \ldots, X_n$, and the positions of Player 1 are the clauses C_i of ψ. Player 0 can move from a position X to any clause C_i with head X, and Player 1 can move from a clause C_i to any variable occurring in the body of C_i. Formally, $\mathcal{G}_\psi = (V, E)$, $V = V_0 \cup V_1$ with $V_0 = \{0\} \cup \{X_1, \ldots, X_n\}$, $V_1 = \{C_i : i \in I\}$, and

$$\begin{aligned} E = &\{(X, C) \in V_0 \times V_1 : X = \text{head}(C)\} \cup \\ &\{(C, X) \in V_1 \times V_0 : X \in \text{body}(C)\}. \end{aligned}$$

Player 0 has a winning strategy for $\mathcal{G}_\psi$ from position X if, and only if, $\psi \models X$. In particular, ψ is unsatisfiable if, and only if, Player 0 wins from position 0.

4.3.2 model-checking games for first-order logic

For a logic L and a domain $\mathcal{D}$ of structures, the ***model-checking problem*** asks, given a structure $\mathfrak{A} \in \mathcal{D}$ and a formula $\psi \in L$, whether it is the case that $\mathfrak{A} \models \psi$. Model-checking problems can be reformulated in game-theoretic terms using appropriate model-checking games. With a sentence ψ, a structure $\mathfrak{A}$ (of the same vocabulary as ψ), we associate a ***model-checking game*** $\mathcal{G}(\mathfrak{A}, \psi)$. It is played by two players, ***Verifier*** and ***Falsifier***. Verifier (also called Player 0) tries to prove that $\mathfrak{A} \models \psi$, whereas Falsifier (also called

Player 1) tries to establish that the sentence is false. For first-order logic, the evaluation games are simple, in the sense that (1) all plays are finite (regardless of whether the input structure is finite or infinite) and (2) winning conditions are defined in terms of reachability.

Let us assume that $\mathfrak{A} = (A, R_1, \dots, R_m)$ is a relational structure and ψ is a first-order sentence in negation normal form, i.e., built up from atoms and negated atoms by means of the propositional connectives $\wedge, \vee$ and the quantifiers $\exists, \forall$. Obviously, any first-order formula can be converted in linear time into an equivalent one in negation normal form. The model-checking game $\mathcal{G}(\mathfrak{A}, \psi)$ has positions $\varphi(\overline{a})$ where $\varphi(\overline{x})$ is a subformula of ψ which is instantiated by a tuple $\overline{a}$ of elements of A. The initial position of the game is the formula ψ.

Verifier (Player 0) moves from positions associated with disjunctions and with formulae starting with an existential quantifier. From a position $\varphi \vee \vartheta$, she moves to either φ or ϑ. From a position $\exists y \varphi(\overline{a}, y)$, Verifier can move to any position $\varphi(\overline{a}, b)$, where $b \in A$. Dually, Falsifier (Player 1) makes corresponding moves for conjunctions and universal quantifications. At atoms or negated atoms, i.e., positions $\varphi(\overline{a})$ of the form $a = a'$, $a \neq a'$, $R\overline{a}$, or $\neg R\overline{a}$, the game is over. Verifier has won the play if $\mathfrak{A} \models \varphi(\overline{a})$; otherwise, Falsifier has won.

Model-checking games are a way of defining the semantics of a logic. The equivalence to the standard definition can be proved by a simple induction.

Theorem 4.4 *Verifier has a winning strategy from position $\varphi(\overline{a})$ in the game $\mathcal{G}(\mathfrak{A}, \psi)$ if, and only if, $\mathfrak{A} \models \varphi(\overline{a})$.*

This suggests a game-based approach to model-checking: given $\mathfrak{A}$ and ψ, construct the game $\mathcal{G}(\mathfrak{A}, \psi)$ and decide whether Verifier has a winning strategy from the initial position.

4.3.3 Complexity of first-order model-checking

A model-checking problem has two inputs: a structure and a formula. We can measure the complexity in terms of both inputs, and this is what is commonly referred to as the ***combined complexity*** of the model-checking problem (for L and $\mathcal{D}$). However, in many cases, one of the two inputs is fixed, and we measure the complexity only in terms of the other. If we fix the structure $\mathfrak{A}$, then the model-checking problem for L on this structure amounts to deciding $\mathrm{Th}_L(\mathfrak{A}) := \{\psi \in L : \mathfrak{A} \models \psi\}$, the ***$L$-theory*** of $\mathfrak{A}$. The complexity of this problem is called the ***expression complexity*** of the model-checking problem (for L on $\mathfrak{A}$). Especially in finite model theory, one often considers model-checking problems for a fixed formula ψ, which amounts to deciding the ***model class*** of ψ inside $\mathcal{D}$, $\text{Mod}_{\mathcal{D}}(\psi) := \{\mathfrak{A} \in \mathcal{D} : \mathfrak{A} \models \psi\}$. Its complexity is the ***structure complexity*** of the model-checking problem (for ψ on $\mathcal{D}$).

Since reachability games can be solved in linear time, the size of the game graph directly gives us an upper bound for the time complexity for first-order model-checking. The size of the model-checking game $\mathcal{G}(\mathfrak{A}, \psi)$ is the number of different instantiations of the subformulae of ψ with elements from $\mathfrak{A}$. It depends on several parameters, including the cardinality of the structure $\mathfrak{A}$, the number of subformulae of ψ (which is of course bounded by the length ψ) and the ***width*** of ψ which is defined as the maximal number of free variables in subformulae of ψ. Clearly, $|\mathcal{G}(\mathfrak{A}, \psi)| \leq |\psi| \cdot |A|^{\text{width}(\psi)}$, so the crucial parameter is the width of the formula: if we have subformulae with many free variables, then the number of instantiations, and thus the size of the game, becomes very large. In general the combined complexity and the expression complexity of first-order model-checking problem are PSPACE-complete. In turn, the game graphs have polynomial size for any class of first-order formulae with bounded width.

Theorem 4.5 *The model-checking problem for first-order logic is* PSPACE-*complete. For any fixed $k \geq 2$, the model-checking problem for first-order formulae of width at most k is* PTIME-*complete.*

Exercise 4.1 Prove the hardness results. Reduce QBF, the problem of evaluating quantified Boolean formulae, to the model-checking problem for first-order logic on a fixed structure with two elements. Reduce the problem of solving reachability games to the model-checking problem for formulae of width 2.

By applying the game-based analysis of model-checking to the case of a fixed sentence ψ, we see that the structure complexity of first-order logic is much lower than the expression or combined complexity. In particular, the evaluation problem for any fixed first-order sentence can be computed deterministically in logarithmic space.

For a detailed study of the complexity of first-order model-checking, giving precise complexity bounds in terms of deterministic and alternating complexity classes, the reader may consult Grädel [2007].

4.3.4 Definability of winning regions

Let $\mathcal{S}$ be a class of games, represented as structures of some fixed vocabulary. We say that ***winning regions on $\mathcal{S}$ are definable in a logic L*** if there exist formulae $\psi_0(x)$ and $\psi_1(x)$ of L that define, on each game $\mathcal{G} \in \mathcal{S}$, the winning regions W_0 and W_1 for the two players. This means that, for each game $\mathcal{G} \in \mathcal{S}$, and $\sigma = 0, 1$

$$W_\sigma = \{v \in \mathcal{G} : \mathcal{G} \models \psi_\sigma(v)\}.$$

We can view a logic L and class $\mathcal{S}$ of games as ***balanced***, if on the one hand, $\mathcal{S}$ provides model-checking games for L, and on the other hand, the winning regions for games in $\mathcal{S}$ are definable in L.

While reachability games are appropriate model-checking games for first-order logic, the reverse relationship does not hold. Indeed it is well-known that the expressive power of first-order logic, for defining properties of finite or infinite structures, is rather limited. A general result making this precise is ***Gaifman's Theorem***, saying that first-order logic can express only ***local properties***. For an exact statement and proof of this fundamental result, we refer to Ebbinghaus and Flum [1999]. Perhaps the simplest query that is not local, and hence not first-order definable, is ***reachability***: Given a directed graph $G = (V, E)$ and a starting node v, determine the set of all nodes that are reachable from v. This also implies that first-order logic is too weak for reachability games; indeed the reachability problem can be viewed as the problem of computing winning regions in the special case of one-player reachability games.

Theorem 4.6 *Winning regions of reachability games are not first-order definable.*

Thus, already for reachability games, and even more so for parity games, more powerful logics are required to define the winning regions. Appropriate logics for this are fixed-point logics that we are going to study in the next section. In particular, we shall see that LFP and parity games (with a bounded number of priorities) are balanced.

4.4 Logics with least and greatest fixed-points

Consider a formula $\psi(R, \overline{x})$ of vocabulary $\tau \cup \{R\}$ where $\overline{x}$ is a tuple of variables whose length matches the arity of R. Such a formula defines, for every τ-structure $\mathfrak{A}$, an ***update operator*** $F_\psi : \mathcal{P}(A^k) \to \mathcal{P}(A^k)$ on the class of k-ary relations on A, by

$$F_\psi : R \mapsto \{\overline{a} : (\mathfrak{A}, R) \models \psi(R, \overline{a})\}.$$

A ***fixed-point*** of F_ψ is a relation R for which $F_\psi(R) = R$. ***Fixed-point logics*** extend a basic logical formalism (such as first-order logic, conjunctive queries, or propositional modal logic) by formulae defining *fixed-points of relational operators.* Notice that, in general, fixed-points of F_ψ need not exist, or there may exist many of them. We therefore consider special kinds of fixed-points, such as least and greatest, and later inflationary and deflationary fixed-points, and we impose additional conditions on the relational operators to guarantee that these fixed-points exist.

We shall now describe some basic facts of fixed-point theory for powerset lattices $(\mathcal{P}(B), \subseteq)$, where B is an arbitrary (finite or infinite) set. An operator $F : \mathcal{P}(B) \to \mathcal{P}(B)$ is ***monotone***, if it preserves inclusion, i.e., $F(X) \subseteq F(Y)$ whenever $X \subseteq Y$. A fixed-point X of F is called the ***least fixed-point*** of F if $X \subseteq Y$ for all fixed-points Y of F. Similarly, if all fixed-points of F are subsets of a fixed-point X, then X is the ***greatest fixed-point*** of F.

Theorem 4.7 (Knaster–Tarski) *Every monotone operator $F : \mathcal{P}(B) \to \mathcal{P}(B)$ has a least fixed-point* $\mathbf{lfp}(F)$ *and a greatest fixed-point* $\mathbf{gfp}(F)$*. Further, these fixed-points may be written in the form*

$$\begin{aligned}\mathbf{lfp}(F) &= \bigcap\{X : F(X) = X\} = \bigcap\{X : F(X) \subseteq X\}\\ \mathbf{gfp}(F) &= \bigcup\{X : F(X) = X\} = \bigcup\{X : F(X) \supseteq X\}.\end{aligned}$$

A proof can be found in any standard exposition on fixed-point theory or fixed-point logics (see e.g., Grädel [2007]). Least fixed-points can also be constructed inductively. We call an operator $F : \mathcal{P}(B) \to \mathcal{P}(B)$ ***inductive*** if the sequence of its ***stages*** X^α (where α ranges over the ordinals), defined by

$$\begin{aligned}X^0 &:= \emptyset,\\ X^{\alpha+1} &:= F(X^\alpha), \text{ and}\\ X^\lambda &:= \bigcup_{\alpha<\lambda} X^\alpha \text{ for limit ordinals } \lambda,\end{aligned}$$

is increasing, i.e., if $X^\beta \subseteq X^\alpha$ for all $\beta < \alpha$. Obviously, monotone operators are inductive. The sequence of stages of an inductive operator eventually reaches a fixed-point, which we denote by X^∞. The least ordinal β for which $X^\beta = X^{\beta+1} = X^\infty$ is called the ***closure ordinal*** of F.

Exercise 4.2 Prove that the *cardinality* of the closure ordinal of every inductive operator $F : \mathcal{P}(B) \to \mathcal{P}(B)$ is bounded by the cardinality of B. However, the closure ordinal itself can be larger than $|B|$. Prove this by an example.

Theorem 4.8 *For monotone operators, the inductively constructed fixed-point coincides with the least fixed-point:* $X^\infty = \mathbf{lfp}(F)$*.*

Proof As X^∞ is a fixed-point, $\mathbf{lfp}(X) \subseteq X^\infty$. For the converse, we show by induction that $X^\alpha \subseteq \mathbf{lfp}(F)$ for all α. As $\mathbf{lfp}(F) = \bigcap\{Z : F(Z) \subseteq Z\}$, it suffices to show that X^α is contained in all Z for which $F(Z) \subseteq Z$.

For $\alpha = 0$, this is trivial. By monotonicity and the induction hypothesis, we have $X^{\alpha+1} = F(X^\alpha) \subseteq F(Z) \subseteq Z$. For limit ordinals λ with $X^\alpha \subseteq Z$ for all $\alpha < \lambda$ we also have $X^\lambda = \bigcup_{\alpha<\lambda} \subseteq Z$. □

The greatest fixed-point can be constructed by a dual induction, starting with $Y^0 = B$, by setting $Y^{\alpha+1} := F(Y^\alpha)$ and $Y^\lambda = \bigcap_{\alpha<\lambda} Y^\alpha$ for limit ordinals. The *decreasing* sequence of these stages then eventually converges to the greatest fixed-point $Y^\infty = \mathbf{gfp}(F)$.

The least and greatest fixed-points are dual to each other. For every monotone operator F, the dual operator $F^d : X \mapsto \overline{F(\overline{X})}$ (where $\overline{X}$ denotes the complement of X) is also monotone, and we have that

$$\mathbf{lfp}(F) = \overline{\mathbf{gfp}(F^d)} \text{ and } \mathbf{gfp}(F) = \overline{\mathbf{lfp}(F^d)}.$$

4.4.1 Least fixed-point logic and reachability games

Least fixed-point logic (LFP) is defined by adding to the syntax of first-order logic the following *least fixed-point formation rule*: If $\psi(R, \overline{x})$ is a formula of vocabulary $\tau \cup \{R\}$ with only positive occurrences of R, if $\overline{x}$ is a tuple of variables, and if $\overline{t}$ is a tuple of terms (such that the lengths of $\overline{x}$ and $\overline{t}$ match the arity of R), then also

$$[\mathbf{lfp} R\overline{x} \,.\, \psi](\overline{t}) \text{ and } [\mathbf{gfp} R\overline{x} \,.\, \psi](\overline{t})$$

are formulae of vocabulary τ. The free first-order variables of these formulae are those in $(\text{free}(\psi) - \{x : x \text{ in } \overline{x}\}) \cup \text{free}(\overline{t})$.

Semantics. Since R occurs only positive in ψ, the update operator F_ψ, defined by ψ on any τ-structure $\mathfrak{A}$ (providing interpretations for all free variables in the formula) is monotone. We define that $\mathfrak{A} \models [\mathbf{lfp} R\overline{x} \,.\, \psi](\overline{t})$ if, and only if, $\overline{t}^{\mathfrak{A}}$ (the tuple of elements of $\mathfrak{A}$ interpreting $\overline{t}$) is contained in $\mathbf{lfp}(F_\psi)$. The definition for greatest fixed-points is analogous.

Obviously, LFP is a fragment of second-order logic. Indeed, by the Tarski–Knaster Theorem,

$$[\mathbf{lfp} R\overline{x} \,.\, \psi(R, \overline{x})](\overline{y}) \equiv \forall R((\forall \overline{x}(\psi(R, \overline{x}) \to R\overline{x})) \to R\overline{y})$$
$$[\mathbf{gfp} R\overline{x} \,.\, \psi(R, \overline{x})](\overline{y}) \equiv \exists R((\forall \overline{x}(R\overline{x} \to \psi(R, \overline{x})) \land R\overline{y}).$$

Perhaps the simplest example of a problem that is expressible in LFP, but not in first-order logic, is ***reachability***: Given a graph $G = (V, E)$ and a starting point v, find the set of nodes that are reachable by a path from v. It is definable in LFP, by the formula

$$\psi(x) := [\mathbf{lfp} Rx \,.\, x = v \vee \exists z(Rz \wedge Ezx)](x).$$

Indeed, in any graph (G, v), the set $\psi^{G,v} := \{w : G, v \models \psi(w)\}$ is precisely the set of nodes reachable from w.

Exercise 4.3 Prove that the LFP-sentence

$$\psi := \forall y \exists z Fyz \wedge \forall y[\mathbf{lfp} Ry \,.\, \forall x(Fxy \rightarrow Rx)](y)$$

is an infinity axiom, i.e., it is satisfiable but does not have a finite model.

We have noticed above that winning regions of reachability and safety games are not first-order definable. However it not difficult to generalise the LFP-definition of reachability to ***LFP-definitions for the winning regions of reachability (and safety) games***. Consider reachability-safety games $\mathcal{G} = (V, V_0, V_1, E, T)$ where Player 0 wants to reach T and Player 1 tries to stay outside of T. On such games, the winning region W_0 of Player 0 is uniformly definable by the LFP-formula $\psi_0(x) := [\mathbf{lfp}\, Wx \,.\, \varphi](x)$ with

$$\varphi(W, x) := Tx \vee (V_0 x \wedge \exists y(Exy \wedge Wy)) \vee (V_1 \wedge \forall y(Exy \rightarrow Wy)).$$

The complement of W_0, which is the winning region for Player 1 for her associated safety condition, is defined by a greatest fixed-point formula $\psi_1(x) := [\mathbf{gfp}\, Wx \,.\, \eta(W, x)](x)$ with

$$\eta(W, x) := \neg Tx \wedge (V_0 x \rightarrow \forall y(Exy \rightarrow Wy)) \wedge (V_1 \rightarrow \exists y(Exy \wedge Wy)).$$

This is just a special case of the ***duality between least and greatest fixed-points*** which implies that for any formula φ,

$$[\mathbf{gfp} R\overline{x} \,.\, \varphi](\overline{t}) \equiv \neg[\mathbf{lfp} R\overline{x} \,.\, \neg\varphi[R/\neg R]](\overline{t}),$$

where $\varphi[R/\neg R]$ is the formula obtained from φ by replacing all occurrences of R-atoms by their negations. (As R occurs only positively in φ, the same is true for $\neg\varphi[R/\neg R]$.) Because of this duality, greatest fixed-points are sometimes omitted in the definition of LFP. However, for studying the relationship between LFP and games it is much more convenient to keep the greatest fixed-points, and to use the duality (and De Morgan's laws) to translate LFP-formulae to *negation normal form*, i.e., to push negations all the way to the atoms.

4.4.2 Capturing polynomial time

Let φ be a formula such that, for any given structure $\mathfrak{A}$, the update operator $F_\varphi : \mathcal{P}(A^k) \to \mathcal{P}(A^k)$ is monotone and computable in polynomial time (with respect to $|A|$). Then also the fixed-points $\mathbf{lfp}(F_\varphi)$ and $\mathbf{gfp}(F_\varphi)$ are polynomial-time computable since the inductive constructions of least and greatest fixed points terminate after at most $|A|^k$ iterations of F_φ. Together with the fact that first-order operations are polynomial-time computable we can conclude, by induction, that every LFP-definable property of finite structures is computable in polynomial time.

Theorem 4.9 *Let ψ be a sentence in* LFP*. It is decidable in polynomial time whether a given finite structure $\mathfrak{A}$ is a model of ψ. In short,* LFP $\subseteq$ PTIME*.*

Further, we have already seen that LFP can define properties that are actually PTIME-complete, such as winning regions in reachability games. This leads to the question of whether LFP can express *all* properties of finite structures that are computable in polynomial time.

This is indeed the case when we consider *ordered finite structures.* We say that a logic L ***captures a complexity class*** C on a domain $\mathcal{D}$ of finite structures, if (1) for every fixed sentence $\psi \in L$, the complexity of evaluating ψ on structures from $\mathcal{D}$ is a problem in the complexity class C, and (2) every property of structures in $\mathcal{D}$ that can be decided with complexity C is definable in the logic L. For any finite vocabulary τ, we write $\mathrm{Ord}(\tau)$ for the class of all structures $(\mathfrak{A}, <)$, where $\mathfrak{A}$ is a finite τ-structure and $<$ is a linear order on (the universe of) $\mathfrak{A}$. It is one of the most influential results of finite model theory that for every model class $\mathcal{K} \subseteq \mathrm{Ord}(\tau)$ that is decidable in polynomial time, there exists a sentence $\psi \in$ LFP such that $\mathcal{K} = \{\mathfrak{A} \in \mathrm{Ord}(\tau) : \mathfrak{A} \models \psi\}$.

Theorem 4.10 (Immerman and Vardi) *On ordered finite structures, least fixed-point logic captures polynomial time.*

However, in the absence of a linear ordering, LFP fails to express all PTIME-properties. Indeed, there are quite trivial queries on unordered finite structures that are not LFP-definable. A simple example is the question of whether a given finite structure has an even number of elements.

The question of whether there exists a logic that captures PTIME on arbitrary finite structures, originally posed by Chandra and Harel [1982], is the most important open problem of finite model theory. The most promising candidates are suitable extensions of LFP (or other fixed-point logics). However, many people conjecture that no logic whatsoever can capture PTIME on the domain of arbitrary finite structures. Since there exist logics for NP this would imply that P $\neq$ NP.

4.4.3 model-checking games for least fixed-point logic

We now construct evaluation games for LFP-formulae. We make the following assumptions:

(1) Fixed-point formulae do not contain parameters. This means that in a subformula $[\mathbf{fp}\, R\overline{x}\,.\,\varphi]$ (where **fp** means either **lfp** or **gfp**), the formula $\varphi(R,\overline{x})$ contains no free first-order variables besides those in $\overline{x}$. This is no loss of generality since one can always eliminate parameters, but it may affect the complexity of model-checking algorithms.

(2) Formulae are in negation normal form, i.e., negations apply to atoms only. Due to the standard dualities of first-order operators and the duality of least and greatest fixed points, this is no loss of generality.

(3) Every fixed-point variable is bound only once and the free relation variables are distinct from the fixed-point variables. For every fixed-point variable T occurring in ψ, we write φ_T for the unique subformula in ψ of the form $[\mathbf{fp}\, T\overline{x}\,.\,\eta(T,\overline{x})]$.

(4) Finally, we require that each occurrence of a fixed-point variable T in φ_T is inside the scope of a quantifier. Again, this is no loss of generality.

For two fixed-point variables S, T, we say that S ***depends*** on T if T occurs free in φ_S. The transitive closure of this dependency relation is called the ***dependency order***, denoted by $\sqsubset_\psi$. The ***alternation level*** $al_\psi(T)$ of T in ψ is the maximal number of alternations between least and greatest fixed-point variables on the $\sqsubset_\psi$-paths from T. The ***alternation depth*** $ad(\psi)$ of a fixed-point formula ψ is the maximal alternation level of its fixed-point variables.

For a structure $\mathfrak{A}$ and an LFP-sentence ψ, the arena of the model-checking game $\mathcal{G}(\mathfrak{A},\psi)$ is defined as for first-order model-checking games, with additional moves for fixed-point formulae. The positions are subformulae of ψ instantiated by elements of $\mathfrak{A}$. The moves are as in the first-order game, except for the positions associated with fixed-point formulae and with fixed-point atoms. At such positions there is a unique move (by Falsifier, say) to the formula defining the fixed-point. For each fixed-point variable T in ψ, there is a unique subformula $[\mathbf{fp}\; T\overline{x}\;.\;\varphi(T,\overline{x})](\overline{y})$ of ψ. From position $[\mathbf{fp}\; T\overline{x}\;.\;\varphi(T,\overline{x})](\overline{b})$, Falsifier moves to $\varphi(T,\overline{b})$, and from any fixed-point atom $T\overline{c}$, she moves to the position $\varphi(T,\overline{c})$.

Notice that if ψ does not contain fixed-points, this game is the model-checking game for first-order logic. However, if we have fixed-points the games may now admit infinite plays. The winning condition for infinite plays will be a parity condition. To motivate the priority assignment let us discuss some special cases:

Consider a formula with just one **lfp**-operator, applied to a first-order formula. The intuition is that from position $[\mathbf{lfp}\ T\overline{x}\ .\ \varphi(T,\overline{x})](\overline{b})$, Verifier tries to establish that $\overline{b}$ enters T at some stage α of the fixed-point induction defined by φ on $\mathfrak{A}$. The game goes to $\varphi(T,\overline{b})$ and from there, as φ is a first-order formula, Verifier can either win the φ-game in a finite number of steps, or force it to a position $T\overline{c}$, where $\overline{c}$ enters the fixed-point at some stage $\beta < \alpha$. The game then resumes at position $\varphi(\overline{c})$. As any descending sequence of ordinals is finite, Verifier will win the game in a finite number of steps. If the formula is not true, then Falsifier can either win in a finite number of steps or force the play to go through infinitely many positions of the form $T\overline{c}$. Hence, these positions should be assigned priority 1 (and all other positions higher priorities) so that such a play will be won by Falsifier. For **gfp**-formulae, the situation is reversed. Verifier wants to force an infinite play, going infinitely often through positions $T\overline{c}$, so **gfp**-atoms are assigned priority 0.

In the general case, we have a formula ψ with nested least and greatest fixed-points, and in an infinite play of $\mathcal{G}(\mathfrak{A},\psi)$ one may see different fixed-point variables infinitely often. But one of these variables is then the smallest with respect to the dependency order $\sqsubset_\psi$. It can be shown that $\mathfrak{A} \models \psi$ if, and only if, this smallest variable is a **gfp**-variable (provided the players play optimally).

Hence, the priority labelling is defined as follows.

(1) Even priorities are assigned to **gfp**-atoms and odd priorities to **lfp**-atoms.
(2) If $S \sqsubset_\psi T$ and S,T are fixed-point variables of different kinds, then S-atoms should get a lower priority than T-atoms.
(3) All positions that are not fixed-point atoms, get a maximal (i.e., most irrelevant) priority.

This completes the definition of the game $\mathcal{G}(\mathfrak{A},\psi)$. Note that the number of priorities in $\mathcal{G}(\mathfrak{A},\psi)$ is essentially the alternation depth of ψ.

We want to prove that $\mathcal{G}(\mathfrak{A},\psi)$ is indeed a correct model-checking game for ψ in $\mathfrak{A}$. The proof proceeds by induction on $\mathfrak{A}$. The interesting case concerns fixed-point formulae $\psi(\overline{a}) := [\mathbf{gfp} T\overline{x}\,.\,\varphi(\overline{x})](\overline{a})$. By the inductive construction of greatest fixed-points, $\mathfrak{A} \models [\mathbf{gfp} T\overline{x}\,.\,\varphi(\overline{x})](\overline{a})$ if, and only if, $(\mathfrak{A},T^\alpha) \models \varphi(\overline{a})$ for all stages T^α of the **gfp**-induction of φ on $\mathfrak{A}$. Further, by the induction hypothesis, we know that, for every interpretation T_0 of T, $(\mathfrak{A}, T_0) \models \varphi(\overline{a})$ if, and only if, Player 0 has a winning strategy for the game $\mathcal{G}((\mathfrak{A}, T_0), \varphi(\overline{a}))$.

It suffices therefor to show that Player 0 wins the game $\mathcal{G} := \mathcal{G}(\mathfrak{A}, \psi(\overline{a}))$ if, and only if, she wins all games $\mathcal{G}((\mathfrak{A}, T^\alpha), \varphi(\overline{a})$. But this follows from a general fact on parity game, the so-called ***Unfolding Lemma***.

The unfolding of a parity game. Let $\mathcal{G} = (V, V_0, V_1, E, \Omega)$ be a parity game that has at least one node with priority 0 and in which every node v with priority 0 has a unique successor $s(v)$ (i.e., $vE = \{s(v)\}$). This condition holds for the game $Gg(\mathfrak{A}, \psi(\overline{a}))$, since the positions of minimal priority are the fixed-point atoms $T\overline{b}$ which have unique successors $\varphi(\overline{b})$.

Let Z be the set of nodes with priority 0 and let $\mathcal{G}^-$ be the game obtained by deleting from $\mathcal{G}$ all edges $(v, s(v)) \in E \cap (Z \times V)$ so that the nodes in Z become terminal positions. The ***unfolding*** of $\mathcal{G}$ is a sequence $\mathcal{G}^\alpha$ (where α ranges over the ordinals) which all coincide with $\mathcal{G}^-$ up to the winning conditions for the terminal positions $v \in Z$. For every α, we define a decomposition $Z = Z_0^\alpha \cup Z_1^\alpha$, where Z_σ^α is the set of terminal positions $v \in Z$ at which we declare, for the game $\mathcal{G}^\alpha$, that Player σ has won. Further, for every α, we define W_σ^α to be winning region of Player σ in the game $\mathcal{G}^\alpha$. Note that W_σ^α depends of course on the decomposition $Z = Z_0^\alpha \cup Z_1^\alpha$ (also for positions outside Z). In turn, the decomposition of Z for $\alpha + 1$ depends on the winning sets W_σ^α in $\mathcal{G}^\alpha$. We set

$$\begin{aligned} Z_0^0 &:= Z \\ Z_0^{\alpha+1} &:= \{v \in Z : s(v) \in W_0^\alpha\} \\ Z_0^\lambda &:= \bigcap_{\alpha<\lambda} Z_0^\alpha \text{ for limit ordinals } \lambda. \end{aligned}$$

By determinacy, $V = W_0^\alpha \cup W_1^\alpha$ for all α, and with increasing α, the winning sets of Player 0 are decreasing and the winning sets of Player 1 are increasing:

$$\begin{aligned} &W_0^0 \supseteq W_0^1 \supseteq \cdots W_0^\alpha \supseteq W_0^{\alpha+1} \supseteq \cdots \\ &W_1^0 \subseteq W_1^1 \subseteq \cdots W_1^\alpha \subseteq W_1^{\alpha+1} \subseteq \cdots \ . \end{aligned}$$

Hence there exists an ordinal α (whose cardinality is bounded by the cardinality of V) for which $W_0^\alpha = W_0^{\alpha+1} =: W_0^\infty$ and $W_1^\alpha = W_1^{\alpha+1} =: W_1^\infty$. The crucial result on unfoldings of parity games states that these fixed-points coincide with the winning regions W_0 and W_1 of the original game $\mathcal{G}$.

Lemma 4.11 (Unfolding Lemma) $W_0 = W_0^\infty$ *and* $W_1 = W_1^\infty$.

For a proof, see Grädel [2007].

By ordinal induction, one can easily see that the games $\mathcal{G}((\mathfrak{A}, T^\alpha), \varphi(\overline{a}))$ associated with the **gfp**-induction of φ in $\mathfrak{A}$ coincide with the unfolding of the game $\mathcal{G} = \mathcal{G}(\mathfrak{A}, \psi(\overline{a}))$. By the Unfolding Lemma, we conclude that Player 0 wins the game $\mathcal{G}(\mathfrak{A}, \psi(\overline{a}))$ if, and only if, she wins all games $\mathcal{G}((\mathfrak{A}, T^\alpha), \varphi(\overline{a}))$. By the induction hypothesis this holds if, and only if, $(\mathfrak{A}, T^\alpha) \models \varphi(\overline{a})$ for all α, which is equivalent to $\mathfrak{A} \models \psi(\overline{a})$.

For least fixed-point formulae we can dualize the arguments.

Theorem 4.12 *Let ψ be a well-named and parameter-free* LFP*-formula in negation normal form, and let $\mathfrak{A}$ be a relational structure. Then Player 0 has a winning strategy from position $\psi(\overline{a})$ in the game $\mathcal{G}(\mathfrak{A}, \psi(\overline{a}))$ if, and only if, $\mathfrak{A} \models \psi(\overline{a})$.*

For future reference we note that the model-checking games $\psi(\mathfrak{A}, \psi)$ can not only be easily constructed from $\mathfrak{A}$ and ψ, but are also easily (i.e., first-order) definable inside $\mathfrak{A}$.

Theorem 4.13 *For every structure $\mathfrak{A}$ with at least two elements, and every formula $\varphi(\bar{x}) \in$* LFP *the model-checking game $\mathcal{G}(\mathfrak{A}, \varphi)$ is first-order interpretable in $\mathfrak{A}$.*

For finite structures, the size of the game $\mathcal{G}(\mathfrak{A}, \psi(\overline{a}))$ (and the time complexity of its construction) is bounded by $|\psi| \cdot |A|^{\mathrm{width}(\psi)}$. Hence, for LFP-formulae of bounded width, the size of the game is polynomially bounded.

Corollary 4.14 *The model-checking problem for* LFP*-formulae of bounded width (and without parameters) is in* NP ∩ Co-NP*, in fact in* UP ∩ Co-UP.

By the complexity results for parity games mentioned at the end of Section 4.2, we obtain complexity bounds for LFP model-checking which are polynomial with respect to the size of the structure, but exponential in the width and the *alternation depth* of the formula.

Corollary 4.15 *The model-checking problem for* LFP*-formulae of bounded width and bounded alternation depth is solvable in polynomial time.*

We have imposed the condition that the fixed-point formulae do not contain parameters. If parameters are allowed, then, at least with a naive definition of width, Corollary 4.14 is no longer true (unless UP = PSPACE). The intuitive reason is that with parameters one can 'hide' first-order variables in fixed-point variables. Indeed, by Dziembowski [1996] the evaluation problem for quantified Boolean formulae can be reduced to the evaluation of LFP-formulae with two first-order variables (but an unbounded number of monadic fixed-point variables) on a fixed structure with three elements. Hence the expression complexity of evaluating such formulae is PSPACE-complete.

For LFP-formulae of unbounded width, our analysis in terms of model-checking games only gives only exponential time bound. This cannot be improved, even for very simple LFP-formulae (Vardi [1982]).

Theorem 4.16 *The model-checking problem for* LFP*-formulae (of unbounded width) is* EXPTIME*-complete, even for formulae with only one fixed-point operator, and on a fixed structure with only two elements.*

4.4.4 *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) rather than to FO. In other words L_μ relates to ML in the same way as LFP relates to FO.

Modal logics such as ML and the μ-calculus are evaluated on transition systems (alias Kripke structures, alias coloured graphs) at a particular node. Given a formula ψ and a transition system G, we write $G, v \models \psi$ to denote that G holds at node v of G. Recall that formulae of ML, for reasoning about ***transition systems*** $G = (V, (E_a)_{a\in A}, (P_b)_{b\in B})$, are built from atomic propositions P_b by means of the usual propositional connectives and the modal operators $\langle a \rangle$ and $[a]$. That is, if ψ is a formula and $a \in A$ is an action, then we can build the formulae $\langle a \rangle\psi$ and $[a]\psi$, with the following semantics:

$$G, v \models \langle a\rangle\psi \text{ iff } G, w \models \psi \text{ for } \textit{some } w \text{ such that } (v, w) \in E_a,$$
$$G, v \models [a]\psi \text{ iff } G, w \models \psi \text{ for } \textit{all } w \text{ such that } (v, w) \in E_a.$$

If there is only one transition relation, i.e., $A = \{a\}$, then we simply write $\Box$ and $\Diamond$ for $[a]$ and $\langle a\rangle$, respectively.

ML can be viewed as an extension of propositional logic. However, in our context it is more convenient to view it as a simple fragment of first-order logic. A modal formula ψ defines a query on transition systems, associating with G a set of nodes $\psi^G := \{v : G, v \models \psi\}$, and this set can be defined equivalently by a first-order formula $\psi^*(x)$. This translation maps atomic propositions P_b to atoms $P_b x$, it commutes with the Boolean connectives, and it translates the modal operators by use of quantifiers as follows:

$$(\langle a\rangle\psi)^*(x) := \exists y(E_a xy \land \psi^*(y))$$
$$([a]\psi)^*(x) := \forall y(E_a xy \rightarrow \psi^*(y)).$$

Note that the resulting formula has width 2 and can thus be written with only two variables.

Theorem 4.17 *For every formula $\psi \in$ ML, there exists a first-order formula $\psi^*(x)$ of width 2, which is equivalent to ψ in the sense that $G, v \models \psi$ iff $G \models \psi^*(v)$.*

The ***modal μ-calculus*** L_μ extends ML 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 $F_\psi : \mathcal{P}(V) \rightarrow \mathcal{P}(V)$ assigning to every set $X \subseteq V$ the set $\psi^G(X) := \{v \in V : (G, X), v \models \psi\}$. Now,

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

Example 4.18 The formula $\mu X.\varphi \lor \langle a\rangle X$ asserts that there exists a path along a-transitions to a node where φ holds. The formula $\nu X.\mu Y.\langle a\rangle((\varphi \land X) \lor Y)$ says that there exists a path from the current node on which φ holds infinitely often.

Exercise 4.4 Prove that the formulae in Example 4.18 do indeed express the stated properties.

The translation from ML into FO extends to a translation from L_μ into LFP.

Theorem 4.19 *Every formula $\psi \in L_\mu$ is equivalent to a formula $\psi^*(x) \in$ LFP of width two.*

Further the argument proving that LFP can be embedded into second-order logic also shows that L_μ is a fragment of ***monadic second-order logic*** (MSO).

The model-checking games for LFP easily translate into ***games for the μ-calculus***. Given a formula $\psi \in L_\mu$ and a transition system $\mathcal{K}$, we obtain a parity game $\mathcal{G}(\mathcal{K}, \psi)$, with positions (φ, v) where φ is a subformula of ψ and

v is a node of $\mathcal{K}$, such that $\mathcal{K}, v \models \varphi$ if, and only if, Player 0 has a winning strategy in $\mathcal{G}(\mathcal{K}, \psi)$ from position (φ, v). As a consequence, an efficient algorithm for solving parity games would also solve the model-checking problem for L_μ.

Since L_μ-formulae can be seen as LFP-formulae of width two, the bounds established in the previous section apply: The model-checking problem for L_μ is in UP $\cap$ Co-UP, and it is a major open problem whether it can be solved in polynomial time. For L_μ-formulae of bounded alternation depths, the associated parity games have a bounded number of priorities and can therefore be solved in polynomial time.

Also the structure complexity can be settled easily. Since L_μ is a fragment of LFP, all properties expressible in L_μ are decidable in polynomial time. Further, there exist $\psi \in L_\mu$ for which the model-checking problem is PTIME-complete. Indeed, winning regions of reachability games are definable not only in LFP, but also in the μ-calculus. In a game $G = (V, V_0, V_1, E)$, Player 0 has a winning strategy from v if, and only if, $G, v \models \mu X.((V_0 \wedge \Diamond X) \vee (V_1 \wedge \Box X))$.

Despite this result, the μ-calculus is far away from a logic that would capture PTIME. Since L_μ is a fragment of MSO, all word languages definable in L_μ are *regular languages*, and of course, not all PTIME-languages are regular.

4.5 Definability of winning regions in parity games

We have seen that the model-checking problem for the LFP and the modal μ-calculus can be reduced to the problem of computing winning regions in parity games. We now discuss the question of whether, and under what conditions, winning regions of parity games are definable in LFP and the μ-calculus.

To study questions of logical definability for parity games (V, V_0, V_1, E, Ω) we need to represent the games as relational structures. We distinguish between two cases.

For fixed d, we consider parity games where the range of the priority function Ω is in $\{0, \dots, d-1\}$ as structures $\mathcal{G} = (V, V_0, V_1, E, P_0, \dots, P_{d-1})$ where $P_0, \dots, P_{d-1}$ are pairwise disjoint unary relations such that P_i is the set of positions v with $\Omega(v) = i$. We denote this class of structures by $\mathcal{PG}_d$.

On the other hand, to consider classes of parity games with an *unbounded number of priorities*, we consider them as structures

$$\mathcal{G} = (V, V_0, V_1, E, \prec, \mathrm{Odd})$$

where $u \prec v$ means that u has a smaller priority than v, and Odd is the set of nodes with an odd priority. We denote this class of structures by $\mathcal{PG}$.

In each case, when we say that winning regions of parity games are definable in a logic L, we mean that there is is a formula ψ_0 and ψ_1 of L such that for any structure $\mathcal{G} \in \mathcal{PG}$ (resp. $\mathcal{PG}_d$), ψ_σ is true in exactly those nodes in $\mathcal{G}$ from which Player σ has a winning strategy.

4.5.1 Parity games with a bounded number of priorities

For any fixed d, the winning regions of parity games in $\mathcal{PG}_d$ are definable by LFP-formulae with d nested fixed-point operators. For Player 0, the formula is

$$\psi_0^d(x) := [\mathbf{gfp}\ R_0 x\,.\,[\mathbf{lfp}\ R_1 x\,.\,\ldots[\mathbf{fp}\ R_{d-1} x\,.\,\varphi(x, R_0, \ldots, R_{d-1})](x)\ldots](x),$$

where

$$\varphi(x, R_0, \ldots, R_{d-1}) := \bigvee_{i<d} ((V_0 x \wedge P_i x \wedge \exists y\,(Exy \wedge R_i y)) \vee \\ (V_1 x \wedge P_i x \wedge \forall y\,(Exy \rightarrow R_i y))).$$

The fixed-point operators alternate between **gfp** and **lfp**, and hence **fp** = **gfp** if d is odd, and **fp** = **lfp** if d is even.

Theorem 4.20 *For every $d \in \mathbb{N}$, the formula ψ_0^d defines the winning region of Player 0 in parity games with priorities* $0, \ldots, d-1$.

Proof In general, LFP-formulae are hard to understand, especially if they have many alternations between least and greatest fixed-points. However, in this case have an elegant argument based on model-checking games to prove that, for every parity game $\mathcal{G} = (V, V_0, V_1, P_0, \ldots, P_{d-1})$ and every position $v \in V$,

$$\mathcal{G} \models \psi_0^d(v) \iff \text{Player 0 has a winning strategy for } \mathcal{G} \text{ from } v.$$

Let $\mathcal{G}^*$ be the model-checking game for the formula $\psi_0^d(v)$ on $\mathcal{G}$ and identify Verifier with Player 0 and Falsifier with Player 1. Hence, Player 0 has a winning strategy for $\mathcal{G}^*$ if, and only if, $\mathcal{G} \models \psi_0^d(v)$.

By the construction of model-checking games, $\mathcal{G}^*$ has positions of the form $\eta(u)$, where $u \in V$ and η is a subformula of ψ_0^d. The priority of a position $R_i u$ is i, and when $\eta(u)$ is not of this form, then its priority is d.

We claim that the game $\mathcal{G}^*$ is essentially, i.e., up to elimination of stupid moves (which would lead to a loss within one or two moves) and up to contraction of several consecutive moves into one, the same as the original game $\mathcal{G}$. To see this, we compare playing $\mathcal{G}$ from a current position $u \in V_0 \cup P_i$ with playing $\mathcal{G}^*$ from any position $\vartheta_k(u)$, where $\vartheta_k(x)$ is the subformula $[\textbf{gfp}\ R_k \dots]$ or $[\textbf{lfp}\ R_k \dots]$ of $\psi_0^d(x)$. In $\mathcal{G}$, Player 0 selects at position u a successor $w \in uE$, and the play proceeds from w. In $\mathcal{G}^*$, the play goes from $\vartheta_k(u)$ through the positions $\vartheta_{k+1}(u), \dots, \vartheta_{d-1}(u)$ to the inner formula $\varphi(u, R_0, \dots, R_{d-1})$.

This formula is a disjunction, so Verifier (Player 0) decides how to proceed. But her only reasonable choice at this point is to move to the position $(V_0 u \wedge P_i u \wedge \exists y(Euy \wedge R_i y)$, since with any other choice she would lose one move later. But from there, the only reasonable move of Falsifier (Player 1) is to go to position $\exists y(Euy \wedge R_i y)$, and it is now the turn of Player 0 to select a successor $w \in vE$ and move to $(Euw \wedge R_i w)$. This forces Player 1 to move to $R_i w$ from which the play proceeds to $\vartheta_i(w))$.

Thus one move from u to w in $\mathcal{G}$ corresponds to a sequence of moves in $\mathcal{G}^*$ from $\vartheta_k(u)$ to $\vartheta_i(w)$, but the only genuine choice is the move from $\exists y(Euy \wedge R_i y)$ to $(Euw \wedge R_i w)$, i.e., the choice of a successor $w \in uE$. In $\mathcal{G}$, the position u has priority i, and in $\mathcal{G}^*$ the minimal, and hence relevant, priority that is seen in the sequence of moves from $\vartheta_k(u)$ to $\vartheta_i(w)$ is that of $R_i w$ which is also i. The situation for positions $u \in V_1 \cap P_i$ is the same, except that the play in $\mathcal{G}^*$ now goes through $\forall y(Exy \rightarrow R_i y)$ and it is Player 1 who selects a successor $w \in uE$ and forces the play to $R_i w$.

Hence the (reasonable) choices that have to be made by the players in $\mathcal{G}^*$ and the relevant priorities that are seen are the same as in a corresponding play of $\mathcal{G}$. Thus, Player 0 has a winning strategy for $\mathcal{G}$ from v if, and only if, Player 0 has a winning strategy for $\mathcal{G}^*$ from position $\psi_0^d(v)$. But since $\mathcal{G}^*$ is the model-checking game for $\psi_0^d(v)$ on $\mathcal{G}$ this is the case if, and only if, $\mathcal{G} \models \psi_0^d(v)$. □

The formula ψ_1^d defining the winning region for Player 1 is defined similarly. Notice that the formula ψ_σ^d has width two. An analogous construction can be carried out in the μ-calculus. The corresponding formulae are

$$\mathrm{Win}_d = \nu X_0 \mu X_1 \nu X_2 \dots \lambda X_{d-1} \bigvee_{j=0}^{d-1} \big((V_0 \wedge P_j \wedge \Diamond X_j) \vee (V_1 \wedge P_j \wedge \Box X_j) \big).$$

Corollary 4.21 *The following three problems are algorithmically equivalent, in the sense that if one of them admits a polynomial-time algorithm, then all of them do.*

(1) Computing winning regions in parity games.

(2) The model-checking problem for LFP*-formulae of width at most k, for any $k \geq 2$.*

(3) The model-checking problem for the modal μ-calculus.

4.5.2 Alternation hierarchies

The formulae Win_d also play an important role in the study of the ***alternation hierarchy*** of the modal μ-calculus. Clearly, Win_d has alternation depth d and it has been shown that this cannot be avoided. As a consequence the alternation hierarchy of the μ-calculus is strict, a result due to Bradfield [1998] and Arnold [1999].

Sometimes, a slightly stronger formulation of this result is needed, for parity games on finite and strongly connected graphs. This easily follows from the general result by the finite model property of the μ-calculus and by a straightforward reduction to strongly connected games.

Theorem 4.22 *Winning regions in parity games in $\mathcal{PG}_d$ are not definable by formulae in the μ-calculus with alternation depth $< d$, even under the assumption that the game graphs are finite and strongly connected.*

For LFP the strictness of the alternation hierarchy also applies, even on certain fixed infinite structures, such as arithmetic $\mathcal{N} = (\mathbb{N}, +, \cdot)$.

However, on ***finite structures***, the interleaving of least and greatest fixed points (or of **lfp**-operators and negation) can be completely avoided, at the expense of increasing the arity of fixed-point operators. Indeed, a single application of an **lfp**-operator to a first-order formula suffices to express any LFP-definable property (see Immerman [1986] or Ebbinghaus and Flum [1999]).

Theorem 4.23 *On finite structures, every* LFP*-formula is equivalent to a formula of the form $\exists y[\mathbf{lfp} R\overline{x} \,.\, \varphi(R, \overline{x})](y, \dots, y)$.*

This result can be strengthened further. Notice that the model-checking game of a formula $\exists y[\mathbf{lfp} R\overline{x} \,.\, \varphi(R, \overline{x})](y, \dots, y)$ is actually a reachability-safety game. The winning region for Player 0 is this definable by an LFP-formula of a particularly simple form, where the **lfp**-operator is applied to a Δ_2-formula.

Theorem 4.24 (Dahlhaus [1987]) *Every* LFP*-definable property of finite structures can be reduced, by a quantifier-free translation, to the problem of computing winning regions in reachability games.*

Hence even the problem of computing winning regions in reachability games is ***complete for LFP*** via this logical notion of reduction.

4.5.3 Parity games with an unbounded number of priorities

We now show that winning regions of parity games are not definable in LFP when the game graph may be infinite.

Theorem 4.25 *Winning regions in $\mathcal{PG}$ are not definable in* LFP*, even under the assumptions that the game graph is countable and the number of priorities is finite.*

Proof Suppose that $\mathrm{Win}(x) \in \mathrm{LFP}$ defines the winning region of Player 0 on $\mathcal{PG}$. We use this formula to solve the model-checking problem for LFP on $\mathfrak{N} = (\omega, +, \cdot)$.

Recall that, for any $\varphi(x) \in \mathrm{LFP}$, we have a parity game $\mathcal{G}(\mathfrak{N}, \varphi)$ such that, for all n

$$\mathfrak{N} \models \varphi(n) \iff \mathcal{G}(\mathfrak{N}, \varphi) \models \mathrm{Win}(v_n)$$

(where v_n is the initial position associated with $\varphi(n)$)

Further, the model-checking game $\mathcal{G}(\mathfrak{N}, \varphi)$ is first-order interpretable in $\mathfrak{N}$. Hence the formula $\mathrm{Win}(x)$ is mapped, via a first-order translation $\mathfrak{I}_\varphi$, into another LFP-formula $\mathrm{Win}_\varphi(x)$ such that

$$\mathcal{G}(\mathfrak{N}, \varphi) \models \mathrm{Win}(v_n) \iff \mathfrak{N} \models \mathrm{Win}_\varphi(n).$$

The first-order translation $\mathrm{Win}(x) \mapsto \mathrm{Win}_\varphi(x)$ depends on φ, but does not increase the alternation depth. Hence, on arithmetic, every formula $\varphi(x)$ would be equivalent to one of fixed alternation depth:

$$\mathfrak{N} \models \varphi(n) \iff \mathfrak{N} \models \mathrm{Win}_\varphi(n).$$

However, it is known that the alternation hierarchy of LFP on arithmetic is strict. □

Definability on finite graphs. On finite game graphs, the definability issues are different and closely related to complexity. One of the most interesting questions is whether the winning regions are definable in fixed-point logics such as LFP or the μ-calculus.

It is not difficult to see that the μ-calculus is not sufficient (no matter how one would precisely define a μ-calculus on $\mathcal{PG}$). Again, this is a consequence of the strictness of the alternation hierarchy. A μ-calculus formula defining the winning region of Player 0 on $\mathcal{PG}$ could be translated to formulae of the usual μ-calculus on structures $\mathcal{PG}_d$ (for any fixed d) with just a bounded increase of the alternation depth. But this would mean that, for any d, the winning regions of parity games with d priorities can be expressed by a μ-calculus formula with a fixed alternation level, which would contradict the strictness of the alternation hierarchy of L_μ. For details, we refer to Dawar and Grädel [2008]

We now turn to the least fixed-point logic LFP. Clearly, a proof that winning regions of parity games in $\mathcal{PG}$ are LFP-definable would imply that parity games are solvable in polynomial time. Surprisingly, it turns out that also the converse direction holds, despite the fact that LFP is weaker than PTIME.

To prove this, we use a result by Otto [1999] saying that the multi-dimensional μ-calculus, which is a fragment of LFP, captures precisely the *bisimulation-invariant* part of PTIME. See also [Grädel et al., 2007, Section 3.5.3] for an exposition of this result.

Winning positions in parity games are of course invariant under the usual notion of bisimulation (e.g., as structures in $\mathcal{PG}_d$). However, to apply Otto's Theorem for parity games with an unbounded number of priorities, we have to consider bisimulation on structures of the form $\mathcal{G} = (V, V_0, V_1, E, \prec, \mathrm{Odd})$. Let $\tau = \{V_0, V_1, E, \prec, \mathrm{Odd}, v\}$ be the vocabulary of parity games with a starting node, and let $\mathrm{Str}(\tau)$ denote the class of all structures of this vocabulary. If we have two such structures that are indeed parity games, then bisimilarity as τ-structures coincides with the usual notion of bisimilarity in $\mathcal{PG}_d$, for appropriate d. However, not all structures in $\mathrm{Str}(\tau)$ are parity games, and the class of parity games is not closed under bisimulation. An efficient procedure for deciding whether a structure is bisimilar to a parity game is to compute its quotient under bisimulation and checking whether it is a parity game.

For a structure $(\mathcal{G}, v) \in \mathrm{Str}(\tau)$ consider the bisimulation relation $a \sim b$ on elements of $\mathcal{G}$ defined with respect to the binary relations E, $\prec$ and $\prec^{-1}$. That is to say $\sim$ is the largest relation satisfying:

- if $a \sim b$ then a and b agree on the unary relations V_0, V_1 and Odd;
- for every $x \in aE$ there is a $y \in bE$ such that $x \sim y$, and conversely;
- for every x with $a \prec x$ there is a y with $b \prec y$ and $x \sim y$ and conversely; and finally
- for every $x \prec a$ there is a $y \prec b$ such that $x \sim y$, and conversely.

We write $(\mathcal{G}, v)^\sim$ for the *bisimulation quotient* of $(\mathcal{G}, v)$, i.e., the structure whose elements are the equivalence classes in $\mathcal{G}$ with respect to $\sim$ with the relations $V_0, V_1, E, \prec, \mathrm{Odd}$ defined in the natural way and $[v]$ as the starting vertex.

Exercise 4.5 Prove that a structure $(\mathcal{G}, v) \in \mathrm{Str}(\tau)$ is bisimilar to a parity game if, and only if, its bisimulation quotient is a parity game, i.e., $(\mathcal{G}, v)^{\sim} \in \mathcal{PG}$.

Theorem 4.26 *Let $\mathcal{C}$ be any class of parity games on finite game graphs, such that winning positions on its bisimulation quotients are decidable in polynomial time. Then, on $\mathcal{C}$, winning positions are* LFP*-definable.*

Proof Let $\mathrm{Win}\mathcal{C}$ be the class of parity games $(\mathcal{G}, v)$, such that $(\mathcal{G}, v) \in \mathcal{C}$, and Player 0 wins from initial position v. It suffices to construct a bisimulation-invariant class X of structures $(\mathcal{H}, u)$ such that

(1) X is decidable in polynomial time,
(2) $X \cap \mathcal{C} = \mathrm{Win}\mathcal{C}$.

Indeed, by Otto's Theorem X is then definable by an LFP-formula $\psi(x)$, such that, given any parity game $(\mathcal{G}, v) \in \mathcal{C}$ we have

$$\mathcal{G}, v \in \mathrm{Win}\mathcal{C} \iff \mathcal{G}, v \in X \iff \mathcal{G} \models \psi(v).$$

By assumption, there exists a polynomial time algorithm A which, given a parity game $(\mathcal{G}, v) \in \mathcal{C}^{\sim}$, decides whether Player 0 wins $\mathcal{G}$ from v. It is not important what the algorithm returns for quotients outside $\mathcal{C}^{\sim}$, as long as it is isomorphism-invariant and halts in polynomial time. Finally, let B be the algorithm which, given any finite structure in $\mathrm{Str}(\tau)$, first computes its bisimulation quotient, and then applies algorithm A.

Clearly B is a polynomial time algorithm, since bisimulation quotients are efficiently computable. Further the class X of structures accepted by B is invariant under bisimulation. Indeed, let $\mathcal{H}$ and $\mathcal{H}'$ be two bisimilar structures. Then their bisimulation quotients are isomorphic and are therefore either both accepted or both rejected by A. Finally, $X \cap C = \mathrm{Win}\mathcal{C}$. Indeed, given a parity game $\mathcal{G}, v \in \mathcal{C}$, then it has the same winner as its bisimulation quotient which is therefore correctly decided by the algorithm B. □

Corollary 4.27 *On the class $\mathcal{PG}$ of all finite parity games, winning regions are* LFP*-definable if, and only if, they are computable in polynomial time.*

For further results on the definability of winning regions in parity games, we refer to Dawar and Grädel [2008].

4.6 Inflationary fixed-point logic and backtracking games

LFP and the modal μ-calculus are not the only logics based on fixed-point operators. In the context of finite model theory, a rich variety of fixed-point operators has been studied due to the close connection that the resulting logics have with complexity classes. One of the most prominent fixed-point logics is IFP, the logic of *inflationary fixed points*. In finite model theory the logics IFP and LFP have often been used interchangeably as it has long been known that they have equivalent expressive power on finite structures. More recently, it has been shown by Kreutzer [2004] that the two logics are equally expressive even without the restriction to finite structures. However, it has also been proved by Dawar et al. [2004] that MIC, the extension of propositional modal logic by inflationary fixed-points, is vastly more expressive than the modal μ-calculus L_μ and that LFP and IFP have very different structural properties even when they have the same expressive power. This exploration of the different nature of the fixed-point operators leads naturally to the question of what an appropriate model-checking game for IFP might look like.

Our analysis of why parity games are the appropriate model-checking games for LFP logics relied on the *well-foundedness* of the inductive definition of a least fixed-point. The Verifier who is trying to prove that a certain tuple $\overline{a}$ belongs to a least fixed-point relation R, needs to present a well-founded justification for its inclusion. That is, the inclusion of $\overline{a}$ in R may be based on the inclusion of other elements in R whose inclusion in turn needs to be justified but the entire process must be well-founded. On the other hand, the justification for including an element in a greatest fixed-point may well be circular. This interaction between sequences that are required to be finite and those that are required to be infinite provides the structural correspondence with parity games.

A key difference that arises when we consider inflationary fixed points (and, dually, deflationary fixed-points) is that the stage at which a tuple $\overline{a}$ enters the construction of the fixed-point R may be an important part of the justification for its inclusion. In the case of least and greatest fixed-points, the operators involved are monotone. Thus, if the inclusion of $\overline{a}$ can be justified at some stage, it can be justified at all later stages. In contrast, in constructing an inflationary fixed-point, if $\overline{a}$ is included in the set, it is on the basis of the immediately preceding stage of the iteration. It may be possible to reflect this fact in the game setting by including the iteration stage as an explicit component of the game position. However, this would blow up the game enormously, since we would have to take a separate copy of the arena for each ordinal. Our aim is to leave the notion of the game arena unchanged as the product of the structure and the formula. We wish only to change the rules of the game to capture the nature of the inflationary fixed-point operator.

The change we introduce to parity games is that either player is allowed to *backtrack* to an earlier position in the game, effectively to force a *countback* of the number of stages. That is, when a backtracking move is played, the number of positions of a given priority that are backtracked are counted and this count plays an important role in the succeeding play. The precise definition is given in Section 4.6.2 below. The backtracking games we define are more complex than parity games. Winning strategies are necessarily more complicated, requiring unbounded memory, in contrast to the positional strategies that work for parity games. Furthermore, deciding the winner is Pspace-hard and remains hard for both NP and Co-NP even when games have only two priorities. In contrast, parity games are known to be decidable in NP $\cap$ Co-NP and in Ptime when the number of priorities is fixed. We will explain how the model-checking problem for IFP can be represented in the form of backtracking games. The construction allows us to observe that a simpler form of backtracking game suffices which we call *simple* backtracking games, and we will see that the winning regions of simple backtracking games are definable in IFP. Thus, we obtain a tight correspondence between the game and the logic, as exists between LFP and parity games.

4.6.1 Inflationary fixed-point logic

The ***inflationary fixed-point*** of any operator $F \,.\, \mathcal{P}(A^k) \to \mathcal{P}(A^k)$ is defined as the limit of the increasing sequence of sets $(R^\alpha)_{\alpha\in\mathrm{Ord}}$ 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 A^k as the initial stage and taking intersections at successor and limit ordinals.

Inflationary fixed-point logic (IFP) is obtained from FO by allowing formulae of the form $[\mathbf{ifp} R\overline{x} \,.\, \varphi(R,\overline{x})](\overline{x})$ and $[\mathbf{dfp} R\overline{x} \,.\, \varphi(R,\overline{x})](\overline{x})$, for arbitrary φ, defining the inflationary and deflationary fixed-point of the operator induced by φ.

To illustrate the power of IFP, we present here a few examples of situations where inflationary and deflationary fixed-points arise.

Bisimulation. Let $\mathcal{K} = (V, E, P_1, \dots, P_m)$ be a transition system with a binary transition relation E and unary predicates P_i. Bisimilarity on $\mathcal{K}$ is the maximal equivalence relation $\sim$ on V such that any two equivalent nodes satisfy the same unary predicates P_i and have edges into the same equivalence classes. To put it differently, $\sim$ is the greatest fixed-point of the refinement operator $F : \mathcal{P}(V \times V) \to \mathcal{P}(V \times V)$ with

$$\begin{aligned} F : Z \mapsto \Big\{ (u,v) \in V \times V : \bigwedge\nolimits_{i \le m} & P_i u \leftrightarrow P_i v \\ & \wedge \forall u'(Euu' \to \exists v'(Evv' \wedge Zu'v')) \\ & \wedge \forall v'(Evv' \to \exists u'(Euu' \wedge Zu'v)) \Big\}. \end{aligned}$$

For some applications (one of which will appear in Section 4.6.4) one is interested in having not only the bisimulation relation $\sim$ but also a linear order on the bisimulation quotient $\mathcal{K}/_\sim$. That is, we want to define a pre-order $\preceq$ on $\mathcal{K}$ such that $u \sim v$ iff $u \preceq v$ and $v \preceq u$. We can again do this via a fixed-point construction, by defining a sequence $\preceq_\alpha$ of pre-orders (where α ranges over ordinals) such that $\preceq_{\alpha+1}$ refines $\preceq_\alpha$ and $\preceq_\lambda$, for limit ordinals λ, is the intersection of the pre-orders $\preceq_\alpha$ with $\alpha < \lambda$. Let

$$u \preceq_1 v :\Longleftrightarrow \bigwedge_{i \le m} P_i u \to \Big(P_i v \vee \bigvee_{j<i} (\neg P_j u \wedge P_i v) \Big)$$

(i.e., if the truth values of the P_i at u are lexicographically smaller than or equal to those at v), and for any α, let

$$u \sim_\alpha v :\Longleftrightarrow u \preceq_\alpha v \wedge v \preceq_\alpha u.$$

To define the refinement, we say that the $\sim_\alpha$-class C *separates* two nodes u and v, if precisely one of the two nodes has an edge into C. Now, let $u \preceq_{\alpha+1} v$ if, and only if, $u \preceq_\alpha v$ and there is an edge from v (and hence none from u) into the smallest $\sim_\alpha$-class (w.r.t. $\preceq_\alpha$) that separates u from v (if it exists). Since the sequence of the pre-orders $\preceq_\alpha$ is decreasing, it must indeed reach a fixed-point $\preceq$, and it is not hard to show that the corresponding equivalence relation is precisely the bisimilarity relation $\sim$.

The point that we want to stress here is that $\preceq$ is a deflationary fixed-point of a non-monotone induction. Indeed, the refinement operator on pre-orders is not monotone and does not, in general, have a greatest fixed-point. We remark that is not difficult to give an analogous definition of this order by an inflationary, rather than deflationary induction.

The lazy engineer: iterated relativisation. Let $\varphi(x)$ be a specification that should be satisfied by all states a of a system, which we assume to be described as a relational structure $\mathfrak{A}$. Now, suppose that the engineer notices that the system he designed is faulty, i.e., that there exist elements $a \in \mathfrak{A}$

where φ does not hold. Rather than redesigning the system, he tries to just throw away all bad elements of $\mathfrak{A}$, i.e., he relativises $\mathfrak{A}$ to the substructure $\mathfrak{A}|_\varphi$ induced by $\{a : \mathfrak{A} \models \varphi(a)\}$. Unfortunately, it need not be the case that $\mathfrak{A}|_\varphi \models \forall x\varphi(x)$. Indeed, the removal of some elements may have the effect that others no longer satisfy φ. But the lazy engineer can of course iterate this relativisation procedure and define a (possibly transfinite) sequence of substructures $\mathfrak{A}^\beta$, with $\mathfrak{A}^0 = \mathfrak{A}$, $\mathfrak{A}^{\beta+1} = \mathfrak{A}^\beta|_\varphi$ and $\mathfrak{A}^\lambda = \bigcap_{\beta<\lambda} \mathfrak{A}^\beta$ for limit ordinals λ. This sequence reaches a fixed-point $\mathfrak{A}^\infty$ which satisfies $\forall x\varphi(x)$ – but it may be empty.

This process of iterated relativisation is definable by a fixed-point induction in $\mathfrak{A}$. Let $\varphi|_Z$ be the syntactic relativisation of φ to a new set variable Z, obtained by replacing inductively all subformulae $\exists y\alpha$ by $\exists y(Zy \wedge \alpha)$ and $\forall y\alpha$ by $\forall y(Zy \rightarrow \alpha)$. Iterated relativisation means repeated application of the operator

$$F : Z \mapsto \{a : \mathfrak{A}|_Z \models \varphi(a)\} = \{a : \mathfrak{A} \models Za \wedge \varphi|_Z(a)\}$$

starting with $Z = A$ (the universe of $\mathfrak{A}$). Note that F is deflationary but not necessarily monotone.

In logics with inflationary and deflationary fixed points (the universe of) $\mathfrak{A}^\infty$ is uniformly definable in $\mathfrak{A}$ by a formula of the form $[\mathbf{dfp}Zx\,.\,\varphi|_Z](x)$. Since IFP and LFP have the same expressive power, $\mathfrak{A}^\infty$ is also LFP-definable. However, the only known way to provide such a definition is by going through the proof of Kreutzer's Theorem (see Kreutzer [2004]). There seems to be no simple direct definition based on least and greatest fixed-points only.

4.6.2 Parity games with backtracking

Backtracking games are essentially parity games with the addition that, under certain conditions, players can jump back to an earlier position in the play. This kind of move is called backtracking. A backtracking move from position v to an earlier position u is only possible if v belongs to a given set B of backtrack positions, if u and v have the same priority and if no position of smaller priority has occurred between u and v. With such a move, the player who backtracks not only resets the play back to u, she also commits herself to a backtracking distance d, which is the number of positions of priority $\Omega(v)$ that have been seen between u and v. After this move, the play ends when d further positions of priority $\Omega(v)$ have been seen, unless this priority is "released" by a lower priority.

For finite plays we have the winning condition that a player wins if her opponent cannot move. For infinite plays, the winner is determined according to the parity condition, i.e., Player 0 wins a play π if the least priority seen infinitely often in π is even, otherwise Player 1 wins.

Thus the arena $\mathcal{G} := (V, E, V_0, V_1, B, \Omega)$ of a backtracking game is a defined as for parity games, extended by a subset $B \subseteq V$ of backtrack positions. A *play* of $\mathcal{G}$ from initial position v_0 is formed as follows. If, after n steps the play has gone through positions $v_0 v_1 \dots v_n$ and reached a position $v_n \in V_\sigma$, then Player σ can select a successor $v_{n+1} \in v_n E$; this is called an ordinary move. But if $v_n \in B$ is a backtrack position, of priority $\Omega(v_n) = q$, say, then Player σ may also choose to backtrack; in that case she selects a number $i < n$ subject to the conditions that $\Omega(v_i) = q$ and $\Omega(v_j) \geq q$ for all j with $i < j < n$. The play then proceeds to position $v_{n+1} = v_i$ and we set $d(q) = |\{k : i \leq k < n \wedge \Omega(v_k) = q\}|$. This number $d(q)$ is relevant for the rest of the game, because the play ends when $d(q)$ further positions of priority q have been seen without any occurrence of a priority $< q$. Therefore, a play is not completely described by the sequence $v_0 v_1 \dots$ of the positions that have been visited. For instance, if a player backtracks from v_n in $v_0 \dots v_i \dots v_j \dots v_n$, it matters whether she backtracks to i or j, even if $v_i = v_j$ because the associated numbers $d(p)$ are different. For a more formal description of how backtracking games are played we refer to Dawar et al. [2006].

It is easy to see that backtracking games are Borel games, so by Martin's Theorem, they are determined. Since parity games are positionally determined the question arises whether this also holds for backtracking games. However, simple examples show that this is not the case and, indeed, no fixed amount of finite memory suffices for winning strategies.

Theorem 4.28 *Backtracking games in general do not admit finite-memory winning strategies.*

Exercise 4.6 Find an example proving this.

Thus, winning strategies for backtracking games are more complex than the strategies needed for parity games. Also the computational complexity of computing winning regions is higher for backtracking games than for parity games. While it is known that winning regions of parity games can be decided in NP $\cap$ Co-NP (and it is conjectured by many that this problem is actually solvable in polynomial time), the corresponding problem for backtracking games is PSPACE-hard. Further, for any fixed number of priorities, parity games can be decided in PTIME, but backtracking games with just two priorities are already NP-hard (see Dawar et al. [2006]).

4.6.3 Games for IFP

We restrict attention to finite structures. The model-checking game for an IFP-formula ψ on a finite structure $\mathfrak{A}$ is a backtracking game $\mathcal{G}(\mathfrak{A}, \psi) = (V, E, V_0, V_1, B, \Omega)$. As in the games for LFP, the positions are subformulae of ψ, instantiated by elements of $\mathfrak{A}$. We only describe the modifications.

We always assume that formulae are in negation normal form, and write $\overline{\vartheta}$ for the negation normal form of $\neg\vartheta$. Consider any **ifp**-formula $\varphi^*(\overline{x}) := [\mathbf{ifp} T\overline{x} \,.\, \varphi(T, \overline{x})](\overline{x})$ in ψ. In general, φ can have positive or negative occurrences of the fixed-point variable T. We use the notation $\varphi(T, \overline{T})$ to separate positive and negative occurrences of T. To define the set of positions we include also all subformulae of $T\overline{x} \vee \varphi$ and $\overline{T}\overline{x} \wedge \overline{\varphi}$. Note that an **ifp**-subformula in φ is translated into a **dfp**-subformula in $\overline{\varphi}$, and vice versa. To avoid conflicts we have to change the names of the fixed-point variables when doing this, i.e., a subformula $[\mathbf{ifp} R\overline{y} \,.\, \vartheta(R, \overline{R}, \overline{y})](\overline{y})$ in φ will correspond to a subformula $[\mathbf{dfp} R'\overline{y} \,.\, \overline{\vartheta}(\overline{R'}, R', \overline{y})](\overline{y})$ of $\overline{\varphi}$ where R' is a new relation variable, distinct from R.

From a position $\varphi^*(\overline{a})$ the play proceeds to $T\overline{a} \vee \varphi(T, \overline{a})$. When a play reaches a position $T\overline{c}$ or $\overline{T}\overline{c}$ the play proceeds back to the formula defining the fixed-point by a regeneration move. More precisely, the regeneration of an **ifp**-atom $T\overline{c}$ is $T\overline{c} \vee \varphi(T, \overline{c})$, the regeneration of $\overline{T}\overline{c}$ is $\overline{T}\overline{c} \wedge \overline{\varphi}(T, \overline{c})$. Verifier can move from $T\overline{c}$ to its regeneration, Falsifier from $\overline{T}\overline{c}$. For **dfp**-subformulae $\vartheta^*(\overline{x}) := [\mathbf{dfp} R\overline{x} \,.\, \vartheta(R, \overline{x})](\overline{x})$, dual definitions apply. Verifier moves from $\overline{R}\overline{c}$ to its regeneration $\overline{R}\overline{c} \vee \overline{\vartheta}(R, \overline{c})$, and Falsifier can make regeneration moves from $R\overline{c}$ to $R\overline{c} \wedge \vartheta(R, \overline{c})$. The priority assignment associates with each **ifp**-variable T an odd priority $\Omega(T)$ and with each **dfp**-variable R an even priority $\Omega(R)$, such that for any two distinct fixed-point variables S, S', we have $\Omega(S) \neq \Omega(S')$, and whenever S' depends on S, then $\Omega(S) < \Omega(S')$. Positions of the form $S\overline{c}$ and $\overline{S}\overline{c}$ are called S-positions. All S-positions get priority $\Omega(S)$, all other formulae get a higher priority. The set B of backtrack positions is the set of S-positions, where S is any fixed-point variable.

Let us focus on IFP-formulae with a single fixed-point, $\psi := [\mathbf{ifp} T\overline{x} \,.\, \varphi](\overline{a})$ where $\varphi(T, \overline{x})$ is a first-order formula. When the play reaches a position $T\overline{c}$ Verifier can make a regeneration move to $T\overline{c} \vee \varphi(T, \overline{c})$ or backtrack. Dually, Falsifier can regenerate from positions $\overline{T}\overline{c}$ or backtrack. However, since we have only one fixed-point, all backtrack positions have the same priority and only one backtrack move can occur in a play.

In this simple case, the rules of the backtracking game ensure that infinite plays (which are plays without backtracking moves) are won by Falsifier, since **ifp**-atoms have odd priority. However, if one of the players backtracks after the play has gone through α T-positions, then the play ends when α further T-positions have been visited. Falsifier has won, if the last of these is of form $T\overline{c}$, and Verifier has won if it is of form $\overline{T}\overline{c}$.

The differences between IFP model-checking and LFP model-checking are in fact best illustrated with this simple case. We claim that Verifier has a winning strategy for the game $\mathcal{G}(\mathfrak{A}, \psi)$ if $\mathfrak{A} \models \psi$ and Falsifier has a winning strategy if $\mathfrak{A} \not\models \psi$.

We look at the first-order formulae φ^α defining the stages of the induction. Let $\varphi^0(\overline{a}) = \mathit{false}$ and $\varphi^{\alpha+1}(\overline{a}) = \varphi^\alpha(\overline{a}) \vee \varphi[T/\varphi^\alpha, \overline{T}/\overline{\varphi}^\alpha](\overline{x})$. On finite structures $\psi(\overline{a}) \equiv \bigvee_{\alpha<\omega} \varphi^\alpha(\overline{a})$.

The first-order game $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\overline{a}))$ can be seen as an unfolding of the game $\mathcal{G}(\mathfrak{A}, \psi(\overline{a}))$. Every position in $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\overline{a}))$ corresponds to a unique position in $\mathcal{G}(\mathfrak{A}, \psi(\overline{a}))$, and conversely, for a pair (p, β) where p is a position of $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\overline{a}))$ and $\beta \leq \alpha$ is an ordinal, there is a unique associated position p_β of the unfolded game $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\overline{a}))$. When a play in $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\overline{a}))$ reaches a position $T\overline{c}$, it is regenerated to either $T\overline{c}$ or $\varphi(T, \overline{c})$ and such a regeneration move decrements the associated ordinal. The corresponding play in $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\overline{a}))$ proceeds to position $\varphi^\beta(\overline{c})$ or $\varphi[T/\varphi^\beta, \overline{T}/\overline{\varphi}^\beta](\overline{c})$. We can use this correspondence to translate strategies between the two games. Notice that the lifting of a positional strategy f in the unfolded game $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\overline{a}))$ will produce a non-positional strategy f^* in the original game $\mathcal{G}(\mathfrak{A}, \psi)$: start with $\beta = \alpha$ and, for any position p, let $f^*(p) := f(p_\beta)$; at regeneration moves, the ordinal β is decremented.

Consider now a play in $\mathcal{G}(\mathfrak{A}, \psi)$ after a backtracking move prior to which β T-positions have been visited, and suppose that $\mathfrak{A} \models \varphi^\beta(\overline{a})$. Then Verifier has a winning strategy in the first-order game $\mathcal{G}(\mathfrak{A}, \varphi^\beta(\overline{a}))$ (from position $\varphi^\beta(\overline{a})$) which translates into a (non-positional) strategy for the game $\mathcal{G}(\mathfrak{A}, \psi)$ with the following properties: Any play that is consistent with this strategy will either be winning for Verifier before β T-positions have been seen, or the β-th T-position will be negative.

Similarly, if $\mathfrak{A} \not\models \varphi^\beta(\overline{a})$ then Falsifier has a winning strategy for $\mathcal{G}(\mathfrak{A}, \varphi^\beta(\overline{a}))$, and this strategy translates into a strategy for the game $\mathcal{G}(\mathfrak{A}, \psi)$ by which Falsifier forces the play (after backtracking) from position $\psi(\overline{a})$ to a positive β-th T-position, unless she wins before β T-positions have been seen. We hence have established the following fact.

Lemma 4.29 *Suppose that a play on $\mathcal{G}(\mathfrak{A}, \psi)$ has been backtracked to the initial position $\psi(\overline{a})$ after β T-positions have been visited. Verifier has a winning strategy for the remaining game if, and only if, $\mathfrak{A} \models \varphi^\beta(\overline{a})$.*

From this we obtain the desired result.

Theorem 4.30 *If $\mathfrak{A} \models \psi(\overline{a})$, then Verifier wins the game $\mathcal{G}(\mathfrak{A}, \psi(\overline{a}))$ from position $\psi(\overline{a})$. If $\mathfrak{A} \not\models \psi(\overline{a})$, then Falsifier wins the game $\mathcal{G}(\mathfrak{A}, \psi(\overline{a}))$ from position $\psi(\overline{a})$.*

Proof Suppose first that $\mathfrak{A} \models \psi(\overline{a})$. Then there is some ordinal $\alpha < \omega$ such that $\mathfrak{A} \models \varphi^\alpha(\overline{a})$. We construct a winning strategy for Verifier in the game $\mathcal{G}(\mathfrak{A}, \psi(\overline{a}))$ starting at position $\psi(\overline{a})$.

From $\psi(\overline{a})$ the game proceeds to $(T\overline{a} \vee \varphi(\overline{a}))$. At this position, Verifier repeatedly chooses the node $T\overline{a}$ until this node has been visited α-times. After that, she backtracks and moves to $\varphi(\overline{a})$. By Lemma 4.29 and since $\mathfrak{A} \models \varphi^\alpha(\overline{a})$, Verifier has a strategy to win the remaining play.

Now suppose that $\mathfrak{A} \not\models \psi(\overline{a})$. If, after α T-positions, one of the players backtracks, then Falsifier has a winning strategy for the remaining game, since $\mathfrak{A} \not\models \varphi^\alpha(\overline{a})$. Hence, the only possibility for Verifier to win the game in a finite number of moves is to avoid positions $\overline{Tb}$ where Falsifier can backtrack. Consider the formulae φ^α_f, with $\varphi^0_f = \mathit{false}$ and $\varphi^{\alpha+1}_f(\overline{x}) = \varphi[T/\varphi^\alpha_f, \overline{T}/\mathit{false}](\overline{x})$. They define the stages of $[\mathbf{ifp} T\overline{x} \,.\, \varphi[T, \mathit{false}](\overline{x})]$, obtained from ψ by replacing negative occurrences of T by *false*. If Verifier could force a finite winning play, with $\alpha - 1$ positions of the form $T\overline{c}$ and without positions $\overline{T\overline{c}}$, then she would in fact have a winning strategy for the model-checking game $\mathcal{G}(\mathfrak{A}, \varphi^\alpha_f(\overline{a}))$. Since ψ^α_f implies φ^α, it would follow that $\mathfrak{A} \models \varphi^\alpha(\overline{a})$. But this is impossible. $\square$

The extension of the proof of Theorem 4.30 to arbitrary IFP-formulae poses no major difficulties. Proceeding by induction on the number of nested fixed-point formulae, one has to combine the argument just given (applied to the outermost fixed-point) with the correctness proof for the LFP-model-checking games. Notice that the essential differences between backtracking games and parity games are in the effects of backtracking moves. Backtracking moves impose a finiteness condition on one priority (unless it is later released by smaller priority) and the effect of such a move remains essentially the same in the general case as in the case of formulae with a single fixed-point. On the other hand, an infinite play in an IFP-model-checking game is a play in which the backtracking moves do not play a decisive role. The winner of such a play is determined by the parity condition and the analysis of such plays closely follows the proof that parity games are the model-checking games for LFP-formulae.

4.6.4 Definability of winning regions in backtracking games

We have seen that backtracking games can be used as model-checking games for IFP. We will now identify a natural subclass of backtracking games, which we call *simple* which is balanced with IFP. This means that for every formula $\varphi \in$ IFP and finite structure $\mathfrak{A}$, the game $\mathcal{G}(\mathfrak{A}, \varphi)$ can trivially be modified to fall within this class and, on the other hand, for every $d \in \mathbb{N}$ there is a formula $\varphi \in$ IFP defining the winning region for Player 0 in any simple backtracking game with at most d priorities. In this sense, simple backtracking games precisely capture IFP model-checking.

Consider the model-checking game $\mathcal{G}(\mathfrak{A}, \varphi)$ and the way backtracking was used there: if Player 0 wanted to backtrack it was always after opening a fixed-point, say $[\mathbf{ifp} R\overline{x} \,.\, R\overline{x} \vee \varphi]$. She then looped α times through the $R\overline{x}$ sub-formula and backtracked. By choosing the α she essentially picked a stage of the fixed-point induction on φ and claimed that $\overline{x} \in \varphi^{\alpha}$. From this observation we can derive two important consequences. As every inflationary fixed-point induction must close after polynomially many steps in the size of the structure $\mathfrak{A}$ and therefore in linearly many steps in terms of the game graph, there is no need for Player 0 to backtrack more than n steps, where n is the size of the game graph. Further, the game can easily be modified such that instead of having the nodes for the disjunction $R\overline{x} \vee \varphi$ and the sub-formula $R\overline{x}$, we simply have a node for φ with a self-loop. In this modified game graph, not only is it sufficient for Player 0 to backtrack no more than n steps, we can, in addition, require that whenever she backtracks from a node v, it must be to v again, i.e., when she decides to backtrack from a node corresponding to the formula φ, she loops α times through φ and then backtracks α steps to φ again. The same is true for Player 1 and her backtracking.

We call a strategy in a backtracking game $\mathcal{G}$ ***local*** if all backtracking moves from any node v are to a previous occurrence of v. Given a function $f \,.\, \mathbb{N} \to \mathbb{N}$, we call a strategy f-backtracking if all backtracking moves made by the strategy have distance at most $f(|\mathcal{G}|)$. The strategy is called ***linear*** in case $f(n) = n$ and ***polynomial*** if f is a polynomial in n.

A backtracking game $\mathcal{G} := (V, E, V_0, V_1, B, \Omega)$ is ***simple***, if every node in B has a self-loop and both players have local linear winning strategies on their winning regions.

Theorem 4.31 *For any* IFP*-formula ψ and every finite structure $\mathfrak{A}$, the model-checking game $\mathcal{G}(\mathfrak{A}, \varphi)$, as defined in Section 4.6.3, is simple.*

We now want to show that the logic IFP is balanced with the class of simple backtracking games, i.e., the winning regions of simple backtracking games are IFP-definable.

Since backtracking games are extensions of parity games we start with the formula defining winning regions in parity games. We take this formula as a starting point for defining an IFP-formula deciding the winner of backtracking games. To define strategies involving backtracking, we first need some preparation. In particular, in order to measure distances we need an ordering on the arenas.

It is easily seen that backtracking games are invariant under bisimulation. Thus, it suffices to consider arenas where no two distinct nodes are bisimilar (we refer to such arenas as *bisimulation minimal*). The next step is to define an ordering on the nodes in an arena. This is done by ordering the bisimulation types realised in it.

Indeed, we have seen above that there is a formula $\varphi_{\text{ord}}(x, y) \in$ IFP defining on every bisimulation minimal arena a linear order. As a result, we can assume that the backtracking games are ordered and that we are given an arithmetical predicate for addition with respect to the order defined above.

Theorem 4.28, saying that there exist backtracking games whose winning strategies require infinite memory, also applies to games with local strategies. In general, the reason for the increased memory consumption is that when the decision to backtrack is made, it is necessary to know which nodes have been seen in the past, i.e., to which node a backtracking move is possible. Furthermore, after a backtracking move occurred, both players have to remember the backtracking distance, as this determines their further moves. However, since here we consider strategies with local backtracking only, it suffices to know the distance of the backtracking moves that are still active, i.e., have not yet been released, whereas the history of the play in terms of nodes visited may safely be forgotten. Thus we can capture all the relevant information about a partial play π ending in position v by the tuple $(v, d_\pi(0), \dots, d_\pi(k-1))$, where d_π denotes the distance function.

In a backtracking game with priorities $0, \dots, k-1$, a ***configuration*** is a pair $(v, \overline{d})$ consisting of a node v and a tuple $\overline{d} \in (\mathbb{N} \cup \{\infty\})^k$. Let π be a (partial) play ending in node v. The configuration of π is defined as the tuple $(v, d_\pi(0), \dots, d_\pi(k))$.

Recall that in a simple backtracking game the distance of all backtracking moves is at most $n := |\mathcal{G}|$. Furthermore we can assume that we are given a linear order on the nodes of the game graph. Thus the configuration of any (partial) play π in a simple game can be represented by a pair $(v, \overline{d})$ where

$\overline{d} \in \{0, \dots, n, \infty\}^k$ and we can use nodes in the game graph to represent the values of the d_i.

The structure of the formulae ψ_0^k defining the winning region for Player 0 in backtracking games with priorities $< k$ is similar to the structure of the corresponding LFP-formula for parity games. It has the form

$$\psi_0^k(x) := [\mathbf{gfp}\ R_0 x\overline{d}\,.\,[\mathbf{lfp}\ R_1 x\overline{d}\,.\ \dots [\mathbf{fp} R_{k-1} x\overline{d}\,.\,\varphi(\overline{R}, x, \overline{d})] \dots](x, \infty, \dots \infty)$$

with k nested fixed-points, applied to a formula φ which is first-order, up to the IFP-subformula defining the order of the bisimulation types. In its various nested fixed-points the formula builds up sets of configurations $(x, d_0, \dots, d_{k-1})$ such that if $(x, d_0, \dots, d_k) \in R_{\Omega(x)}$, then Player 0 can extend any partial play π, ending in node x with $d_\pi(j) = d_j$ for all $j < k$, to a winning play.

We do not give an explicit construction here, but explain the idea. For details, we refer to Dawar et al. [2006].

First of all the formula $\varphi(\overline{R}, x, \overline{d})$ states that for some i, the priority of x is i and the tuple $(d_0, \dots, d_{k-1})$ has ∞ at all positions greater than i (which corresponds to the fact that a node of priority i releases all backtracking moves on higher priorities). Further, if x is a position of Player 0, then she can win from configuration $(x, \overline{d})$ if she can move to a successor y of x from which she wins the play. Winning from y means that the configuration $(y, \overline{d}')$ reached from $(x, \overline{d})$ by moving to y is in $R_{\Omega(y)}$. Thus the formula must define what it means for $(y, \overline{d}')$ to be the configuration reached from x when moving to y.

This involves a case distinction.

If $d_i = \infty$, Player 0 can either do an ordinary move or, in case $x \in B$, a backtracking move. After an ordinary move to a successor node y of priority j the play will have the configuration $(y, \overline{d}')$ which satisfies $\overline{d}' = (d_0, \dots, d_j, \infty, \dots, \infty))$ and which must be in R_j. After a backtracking move, we will have, for some $m \neq \infty$, a configuration $(x, d_0, \dots, d_{i-1}, m, \infty, \dots, \infty)$ which must be in R_i

In the case that $d_i = m \leq |\mathcal{G}|$, the formulae must express that Player 0 wins the m-step game on priority i from x. This game is won by Player 0 if there is a successor y of x from which she wins and either the priority j of y is less than i, i.e., all backtracking moves on priorities greater than j are released ($d_l = \infty$ for all $l > j$), or the priority j of y equals i and Player 0 wins the $m - 1$ step game from y (and all d_l with $l < i$ are left unchanged), or the priority j of y is greater than i, in which case the play continues with the configuration $(y, d_0, \dots, d_i, \infty, \dots, \infty)$, i.e., all active backtracking moves (whose distances are stored in $d_0, \dots, d_i$) remain unchanged and the play continues on priority j without any active backtracking moves on priorities greater than i.

It is not difficult to express all this in first-order logic, provided an ordering on priorities is available. For nodes where Player 1 moves the construction is very similar.

Exercise 4.7 Make the construction of the formulae ψ_σ^k explicit, and prove that they indeed define the winning region for Player σ.

Theorem 4.32 *Winning regions of simple backtracking games are definable in* IFP.

4.7 Logic and games in a quantitative setting

Common logical formalisms are two-valued and express qualitative properties. There have been a number of proposals to extend logics such as propositional modal logic ML, the temporal logics LTL and CTL, and the modal μ-calculus L_μ to ***quantitative*** formalisms where formulae can take, at a given state of a system, not just the values *true* and *false*, but quantitative values, for instance real numbers. There are several scenarios and applications where it is desirable to replace purely qualitative statements by quantitative ones which can be of very different nature: we may be interested in the probability of an event, the value that we assign to an event may depend on to how late it occurs, we can ask for the number of occurrences of an event in a play, and so on. We can consider transition structures, where already the atomic propositions take numeric values, or we can ask about the 'degree of satisfaction' of a property.

While there certainly is ample motivation to extend qualitative specification formalisms to quantitative ones, there are also problems. It has been observed in many areas of mathematics, engineering and computer science where logical formalisms are applied, that quantitative formalisms in general lack the clean and clear mathematical theory of their qualitative counterparts, and that many of the desirable mathematical and algorithmic properties tend to get lost. Also, the definitions of quantitative formalisms are often ad hoc and do not always respect the properties that are relevant for logical methodologies.

Here we discuss to what extent the relationship between the μ-calculus and parity games can be extended to a quantitative μ-calculus and appropriate quantitative model-checking games. The extension is not straightforward and requires that one defines the quantitative μ-calculus in the 'right' way, so as to ensure that it has appropriate closure and duality properties (such as closure under negation, De Morgan equalities, quantifier and fixed-point dualities) to make it amenable to a game-based approach. But once this is done, one can indeed construct a quantitative variant of parity games, and prove that they are the appropriate model-checking games for the quantitative μ-calculus. As in the classical setting the correspondence goes both ways: The value of a formula in a structure coincides with the value of the associated model-checking game, and conversely, the value of quantitative parity games (with a bounded number of priorities) are definable in the quantitative μ-calculus. However, the mathematical properties of quantitative parity games are different from their qualitative counterparts. In particular, they are, in general, not positionally determined, not even up to approximation, and the proof that the quantitative model-checking games correctly describe the value of the formulae is considerably more difficult than for the classical case.

As in the classical case, model-checking games lead to a better understanding of the semantics and expressive power of the quantitative μ-calculus. Further, the game-based approach also sheds light on the consequences of different choices in the design of the quantitative formalism, which are far less obvious than for classical logics.

4.7.1 Quantitative transition systems and quantitative μ-calculus

We write $\mathbb{R}^+$ for the set of non-negative real numbers, and $\mathbb{R}^+_\infty := \mathbb{R}^+ \cup \{\infty\}$. ***Quantitative transition systems (QTS)*** are directed graphs equipped with quantities at states and with discounts of edges. They have the form $\mathcal{K} = (V, E, \delta, \{P_i\}_{i\in I})$ where (V, E) is a directed graph, with a discount function $\delta : E \to \mathbb{R}^+ \setminus \{0\}$ and functions $P_i : V \to \mathbb{R}^+_\infty$, that assign to each state the values of the predicates at that state. A transition system is ***qualitative*** if all functions P_i assign only the values 0 or ∞, where 0 stands for *false* and ∞ for *true*, and it is ***non-discounted*** if $\delta(e) = 1$ for all $e \in E$.

Given predicate functions $\{P_i\}_{i\in I}$, discount factors $d \in \mathbb{R}^+$ and constants $c \in \mathbb{R}^+$, the ***quantitative μ-calculus*** $Q\mu$ is built in a similar way to the modal μ-calculus, with the following two modifications.

(1) Atomic formulae have the form $|P_i - c|$.
(2) If φ is a $Q\mu$-formula, then so is $d \cdot \varphi$,

Boolean connectives $\wedge$, $\vee$, modal operators $\Diamond$ and $\Box$, and fixed-point operators μ, ν are used as in the syntax of L_μ. The semantics, however, is quite different.

Formulae of $Q\mu$ are interpreted over quantitative transition systems $\mathcal{K} =$

$(V, E, \delta, (P_i)_{i\in I})$. The meaning of a formula φ in $\mathcal{K}$ is a function $[\![\varphi]\!]^{\mathcal{K}} : V \to \mathbb{R}^+_\infty$. We write $\mathcal{F}$ for the set of functions $f : V \to \mathbb{R}^+_\infty$, with $f_1 \leq f_2$ if $f_1(v) \leq f_2(v)$ for all v. Then $(\mathcal{F}, \leq)$ forms a complete lattice with the constant functions $f = \infty$ as $f = 0$ as top and bottom elements.

The interpretation $[\![\varphi]\!]^{\mathcal{K}} : V \to \mathbb{R}^+_\infty$ is defined as follows:

(1) $[\![|P_i - c|]\!]^{\mathcal{K}}(v) := |P_i(v) - c|$,
(2) $[\![\varphi_1 \wedge \varphi_2]\!]^{\mathcal{K}} := \min\{[\![\varphi_1]\!]^{\mathcal{K}}, [\![\varphi_2]\!]^{\mathcal{K}}\}$ and
$[\![\varphi_1 \vee \varphi_2]\!]^{\mathcal{K}} := \max\{[\![\varphi_1]\!]^{\mathcal{K}}, [\![\varphi_2]\!]^{\mathcal{K}}\}$,
(3) $[\![\Diamond\varphi]\!]^{\mathcal{K}}(v) := \sup_{v'\in vE} \delta(v, v') \cdot [\![\varphi]\!]^{\mathcal{K}}(v')$ and
$[\![\Box\varphi]\!]^{\mathcal{K}}(v) := \inf_{v'\in vE} \frac{1}{\delta(v,v')} [\![\varphi]\!]^{\mathcal{K}}(v')$,
(4) $[\![d \cdot \varphi]\!]^{\mathcal{K}}(v) := d \cdot [\![\varphi]\!]^{\mathcal{K}}(v)$,
(5) $[\![\mu X.\varphi]\!]^{\mathcal{K}} := \inf\{f \in \mathcal{F} : f = [\![\varphi]\!]^{\mathcal{K}[X\leftarrow f]}\}$, and
$[\![\nu X.\varphi]\!]^{\mathcal{K}} = \sup\{f \in \mathcal{F} : f = [\![\varphi]\!]^{\mathcal{K}[X\leftarrow f]}\}$.

When interpreted over qualitative transition systems $Q\mu$ coincides with the classical μ-calculus and we say that $\mathcal{K}, v$ is a model of φ, $\mathcal{K}, v \models \varphi$ if $[\![\varphi]\!]^{\mathcal{K}}(v) = \infty$. For discounted systems we take the natural definition for $\Diamond$ and use the dual one for $\Box$ which motivates the $\frac{1}{\delta}$ factor. It has been proved by Fischer et al. [2009] that this is the only definition for which there is a well-behaved negation operator (with $[\![\neg\varphi]\!]^{\mathcal{K}} = 1/[\![\varphi]\!]^{\mathcal{K}}$) and which gives us the dualities that are needed for natural model-checking games.

Note that all operators in $Q\mu$ are monotone. This guarantees the existence of the least and greatest fixed-points, and their inductive definition according to the Knaster–Tarski Theorem: Given a formula $\mu X.\varphi$ and a quantitative transition system $\mathcal{K}$, we obtain the inductive sequence of functions g_α (for ordinals α) where $g_0 := 0$, $g_{\alpha+1} := [\![\varphi]\!]^{\mathcal{K}[X\leftarrow g_\alpha]}$, and $g_\lambda := \lim_{\alpha<\lambda} [\![\varphi]\!]^{\mathcal{K}[X\leftarrow g_\alpha]}$ for limit ordinals λ. Then $[\![\mu X.\varphi]\!]^{\mathcal{K}} = g_\gamma$ for the minimal ordinal γ with $g_\gamma = g_{\gamma+1}$. For $\nu X.\varphi$ the dual induction applies, starting with $g_0 := \infty$.

4.7.2 Quantitative parity games

Quantitative parity games are modest modifications of classical parity games. Quantitative values are assigned to final positions, where they are interpreted as the payoff for Player 0 at that position, and to moves, where they are interpreted as discounts to the payoff when the play goes through that move.

A ***quantitative parity game*** is a tuple $\mathcal{G} = (V, V_0, V_1, E, \delta, \lambda, \Omega)$ extending a classical parity game by two functions $\delta : E \to \mathbb{R}^+$, assigning to every move a ***discount factor***, and $\lambda : \{v \in V : vE = \emptyset\} \to \mathbb{R}^+_\infty$ assigning to every terminal position a ***payoff value***. The ***outcome*** $p(\pi)$ of a finite play

$\pi = v_0 \dots v_k$, ending at a terminal position v_k is computed by multiplying all discount factors seen in the play with the payoff value at the final node,

$$p(v_0 v_1 \dots v_k) = \delta(v_0, v_1) \cdot \delta(v_1, v_2) \cdot \ldots \cdot \delta(v_{k-1}, v_k) \cdot \lambda(v_k).$$

The outcome of an infinite play depends only on the lowest priority seen infinitely often. We assign the value 0 to every infinite play in which the lowest priority seen infinitely often is odd, and ∞ to those in which it is even. Player 0 wants to maximise the outcome whereas Player 1 wants to minimise it.

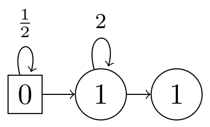
Determinacy. A quantitative game is ***determined*** if, for each position v, the highest outcome that Player 0 can enforce from v and the lowest outcome Player 1 can assure converge,

$$\sup_{\sigma \in \Gamma_0} \inf_{\rho \in \Gamma_1} p(\pi_{\sigma,\rho}(v)) = \inf_{\rho \in \Gamma_1} \sup_{\sigma \in \Gamma_0} p(\pi_{\sigma,\rho}(v)) =: \mathrm{val}\mathcal{G}(v),$$

where Γ_0, Γ_1 are the sets of all possible strategies for Player 0, Player 1. The outcome defined in this way is the ***value*** of $\mathcal{G}$ at v.

One of the fundamental properties of qualitative parity games is the ***positional determinacy***. Unfortunately, this does not generalise to quantitative parity games. Example 4.33 shows that there are simple quantitative games where no player has a positional winning strategy. In the depicted game there is no optimal strategy for Player 0, and even if one fixes an approximation of the game value, Player 0 needs infinite memory to reach this approximation, because she needs to loop in the second position as long as Player 1 looped in the first one to make up for the discounts. (By convention, we depict positions of Player 0 with a circle and of Player 1 with a square and the number inside is the priority for non-terminal positions and the payoff in terminal ones.)

Example 4.33



4.7.3 model-checking games for $Q\mu$

Given a quantitative transition system $\mathcal{K} = (S, T, \delta_S, P_i)$ and a $Q\mu$-formula ψ in negation normal form, we define the model-checking game $\mathrm{MC}[\mathcal{K}, \psi] = (V, V_0, V_1, E, \delta, \lambda, \Omega)$, as a quantitative parity game.

Positions and moves. As in games for L_μ, positions are the pairs (φ, s), where φ is a subformula of φ, and $s \in S$ is a state of the $\mathcal{K}$; in addition we have two special positions (0) and (∞). Positions $(|P_i - c|, s), (0)$, and (∞) are terminal positions. Moves are defined as in the games for L_μ, with the following modifications: positions of the form $(\Diamond\psi, s)$ have either a single successor (0), in case s is a terminal state in $\mathcal{K}$, or one successor (ψ, s') for every $s' \in sT$. Analogously, positions of the form $(\Box\psi, s)$ have a single successor (∞), if $sT = \emptyset$, or one successor (ψ, s') for every $s' \in sT$ otherwise. Positions of the form $(d \cdot \psi, s)$ have a unique successor (ψ, s'). Priorities are assigned in the same way as in the model-checking games for L_μ.

Discounts and payoffs. The discount of an edge is d for transitions from positions $(d \cdot \psi, s)$, it is $\delta_S(s, s')$ for transitions from $(\Diamond\psi, s)$ to (ψ, s'), it is $1/\delta_S(s, s')$ for transitions from $(\Box\psi, s)$ to (ψ, s'), and 1 for all outgoing transitions from other positions. The payoff function λ assigns $|[\![P_i]\!](s) - c|$ to all positions $(|P_i - c|, s)$, ∞ to position (∞), and 0 to position (0).

To prove that $\mathrm{MC}(\mathcal{K}, \psi)$ is indeed an appropriate model-checking game it must be shown that the value of the game starting from v coincides with the value of the formula evaluated on $\mathcal{K}$ at v'. In the qualitative case, that means, that ψ holds in $\mathcal{K}, v'$ if Player 0 wins in $\mathcal{G}$ from v.

Theorem 4.34 *For every formula ψ in $Q\mu$, every quantitative transition system $\mathcal{K}$, and $v \in \mathcal{K}$, the game $MC[\mathcal{K}, \psi]$ is determined and*

$$valMC[\mathcal{K}, \psi](\psi, v) = [\![\psi]\!]^{\mathcal{K}}(v).$$

This is shown by Fischer et al. [2009] using a generalisation of the unfolding method for parity games.

Example 4.35 A model-checking game for $\varphi = \mu X.(P \vee 2 \cdot \Diamond X)$ on the QTS $\mathcal{Q}$ shown in Figure 4.1(a), with $P(1) = 0$, $P(2) = 1$, is depicted in Figure 4.1(b). The nodes are labelled with the corresponding subformulae of φ, and the state of $\mathcal{Q}$. Only the edges with discount factor different from 1 are labelled.

Note that in this game only Player 0 is allowed to make any choices. When we start at the top node, corresponding to an evaluation of φ at 1 in $\mathcal{Q}$, the only choice she has to make is either to keep playing (by looping), or to end the game by moving to a terminal position.

4.7.4 Defining game values in $Q\mu$

As in the case of parity games and LFP (and $L\mu$), also the connection between quantitative parity games and quantitative μ-calculus goes back

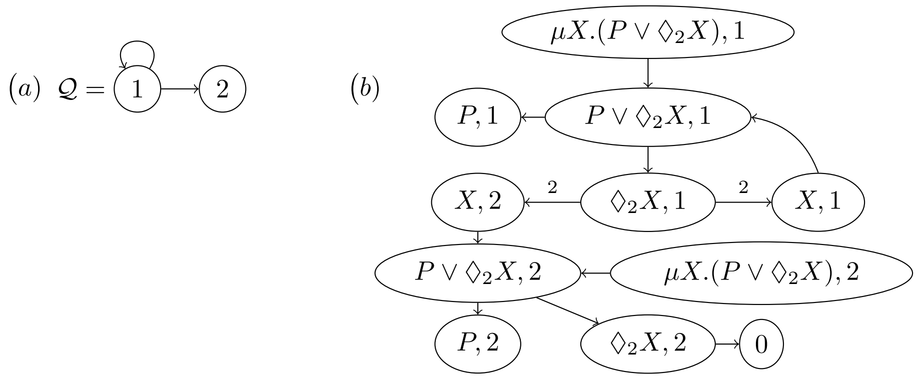


Figure 4.1 Example (a) QTS and (b) model-checking game for $\mu X.(P \vee 2 \cdot \Diamond X)$

and forth. We have seen that quantitative parity games are appropriate model-checking games for the evaluation of $Q\mu$-formulae on quantitative transition systems. For the other direction, we now show that *values* of positions in quantitative parity games (with a bounded number of priorities) are definable in $Q\mu$. It is convenient to describe a quantitative parity game $\mathcal{G} = (V, V_0, V_1, E, \delta_G, \lambda, \Omega_G)$ with priorities in $\{0, \dots d-1\}$ by a quantitative transition system $\mathcal{Q}_\mathcal{G} = (V, E, \delta, V_0, V_1, \Lambda, \Omega)$, where $V_i(v) = \infty$ for positions of Player i, and $V_i(v) = 0$ otherwise, where $\Omega(v) = \Omega_G(v)$ when $vE \neq \emptyset$ and $\Omega(v) = d$ otherwise, with discount function

$$\delta(v, w) = \begin{cases} \delta_G(v, w) & \text{for } v \in V_0, \\ (\delta_G(v, w))^{-1} & \text{for } v \in V_1 \end{cases}$$

and with payoff predicate $\Lambda(v) := \lambda(v)$ in case $vE = \emptyset$ and is $\Lambda(v) = 0$ otherwise.

We then modify the L_μ-formulae Win_d constructed in Section 4.5.1 to quite similar $Q\mu$-formulae

$$\mathrm{QWin}d = \nu X_0 \mu X_1 \nu X_2 \dots \lambda X_{d-1} \bigvee_{j=0}^{d-1} ((V_0 \wedge P_j \wedge \Diamond X_j) \vee (V_1 \wedge P_j \wedge \Box X_j)) \vee \Lambda,$$

where $P_i := \neg(\mu X.(2 \cdot X \vee |\Omega - i|))$. Note that $P_i(v) = \infty$ when $\Omega(v) = i$ and $P_i(v) = 0$ otherwise. The formula QWin_d is therefore analogous to the formula Win_d in the qualitative case.

Theorem 4.36 *For every $d \in \mathbb{N}$, the value of any quantitative parity game with priorities in $\{0, \dots d-1\}$ coincides with the value of $QWin_d$ on the associated transition system.*

Exercise 4.8 Adapt the proof of Theorem 4.20 to get a proof of Theorem 4.36.

References

A. Arnold. The mu-calculus alternation-depth is strict on binary trees. *RAIRO Informatique Théorique et Applications*, 33:329–339, 1999.

J. Bradfield. The modal μ-calculus alternation hierarchy is strict. *Theoretical Computer Science*, 195:133–153, 1998.

A. Chandra and D. Harel. Structure and complexity for relational queries. *Journal of Computer and System Sciences*, 25:99–128, 1982.

E. Dahlhaus. Skolem normal forms concerning the least fixed point. In E. Börger, editor, *Computation Theory and Logic*, number 270 in Lecture Notes in Computer Science, pages 101–106. Springer Verlag, 1987.

A. Dawar and E. Grädel. The descriptive complexity of parity games. In *Proceedings of 22th Annual Conference of the European Association for Computer Science Logic CSL 2008*, pages 354–368, 2008.

A. Dawar, E. Grädel, and S. Kreutzer. Inflationary fixed points in modal logic. *ACM Transactions on Computational Logic*, 5:282 – 315, 2004.

A. Dawar, E. Grädel, and S. Kreutzer. Backtracking games and inflationary fixed points. *Theoretical Computer Science*, 350:174–187, 2006.

W. F. Dowling and J. H. Gallier. Linear-time algorithms for testing the satisfiability of propositional horn formulae. *Journal of Logic Programming*, 1(3):267–284, 1984.

S. Dziembowski. Bounded-variable fixpoint queries are PSPACE-complete. In *10th Annual Conference on Computer Science Logic CSL 96. Selected papers*, Lecture Notes in Computer Science Nr. 1258, pages 89–105. Springer, 1996.

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

A. Emerson and C. Jutla. Tree automata, mu-calculus and determinacy. In *Proc. 32nd IEEE Symp. on Foundations of Computer Science*, pages 368–377, 1991.

D. Fischer, E. Grädel, and L. Kaiser. Model checking games for the quantitative μ-calculus. *Theory of Computing Systems*, 2009.

E. Grädel. Finite Model Theory and Descriptive Complexity. In *Finite Model Theory and Its Applications.* Springer-Verlag, 2007.

E. Grädel and I. Walukiewicz. Positional determinacy of games with infinitely many priorities. *Logical Methods in Computer Science*, 2006.

E. Grädel, P. G. Kolaitis, L. Libkin, M. Marx, J. Spencer, M. Y. Vardi, Y. Venema, and S.Weinstein. *Finite Model Theory and Its Applications.* Springer, Berlin, 2007.

R. Greenlaw, J. Hoover, and W. Ruzzo. *Limits to Parallel Computation. P-Completeness Theory.* Oxford University Press, Oxford, 1995.

N. Immerman. Relational queries computable in polynomial time. *Information and Control*, 68:86–104, 1986.

A. Itai and J. Makowsky. Unification as a complexity measure for logic programming. *Journal of Logic Programming*, 4:105–117, 1987.

M. Jurdziński. Small progress measures for solving parity games. In *Proceedings of 17th Annual Symposium on Theoretical Aspects of Computer Science, STACS*

2000, Lecture Notes in Computer Science Nr. 1770, pages 290–301. Springer, 2000.

M. Jurdziński. Deciding the winner in parity games is in UP ∩ Co-UP. *Information Processing Letters*, 68:119–124, 1998.

M. Jurdziński, M. Paterson, and U. Zwick. A deterministic subexponential algorithm for solving parity games. In *Proceedings of ACM-SIAM Proceedings on Discrete Algorithms, SODA 2006*, pages 117–123, 2006.

S. Kreutzer. Expressive equivalence of least and inflationary fixed point logic. *Annals of Pure and Applied Logic*, 130:61–78, 2004.

D. Martin. Borel determinacy. *Annals of Mathematics*, 102:336–371, 1975.

A. Mostowski. Games with forbidden positions. Technical Report 78, University of Gdańsk, 1991.

M. Otto. Bisimulation-invariant Ptime and higher-dimensional mu-calculus. *Theoretical Computer Science*, 224:237–265, 1999.

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

E. Zermelo. über eine Anwendung der Mengenlehre auf die Theorie des Schachpiels. In *Proc. 5th Internat. Congr. Mathematicians*, volume 2, pages 501–504, Cambridge, 1913.

W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200:135–183, 1998.