Backtracking games and inflationary fixed points

Anuj Dawar

University of Cambridge Computer Laboratory, Cambridge CB3 0FD, UK, anuj.dawar@cl.cam.ac.uk

Erich Grädel

Mathematische Grundlagen der Informatik, RWTH Aachen University, graedel@informatik.rwth-aachen.de

Stephan Kreutzer

Logic in Computer Science, Humboldt-University, Berlin, kreutzer@informatik.hu-berlin.de

Abstract

We define a new class of games, called *backtracking games*. Backtracking games are essentially parity games with an additional rule allowing players, under certain conditions, to return to an earlier position in the play and revise a choice or to force a countback of the number of moves. This new feature makes backtracking games more powerful than parity games. As a consequence, winning strategies become more complex objects and computationally harder. The corresponding increase in expressiveness allows us to use backtracking games as model checking games for inflationary fixed-point logics such as IFP or MIC. We identify a natural subclass of backtracking games, the *simple games*, and show that these are the "right" model checking games for IFP by a) giving a translation of formulae φ and structures $\mathfrak{A}$ into simple games such that $\mathfrak{A} \models \varphi$ if, and only if, Player 0 wins the corresponding game and b) showing that the winner of simple backtracking games can again be defined in IFP.

[1] First author supported in part by EPSRC grant GR/S06721.

1 Introduction

The view of logic as a dialectic game, a set of rules by which a proponent attempts to convince an opponent of the truth of a proposition, has deep roots going back to Aristotle. One of the modern manifestations of this view is the presentation of the semantics of logical operators as moves in a two-player game. A paradigmatic example is the Hintikka semantics of first-order logic, which is just one instance of what are now commonly called *model-checking games*. These are two-player games played on an arena which is formed as the product of a structure $\mathfrak{A}$ and a formula φ where one player attempts to prove that φ is satisfied in $\mathfrak{A}$ while the other player attempts to refute this.

Model-checking games have proved an especially fruitful area of study in connection with logics for the specification of concurrent systems. The modal μ-calculus L_μ is widely used to express properties of such systems and, in terms of expressive power it subsumes a variety of common modal and temporal logics. The most effective algorithms for model checking properties specified in L_μ are based on *parity games*. Formally, a parity game is played on an arena $\mathcal{G} := (V, E, V_0, V_1, \Omega)$, where (V, E) is a directed graph, $V_0, V_1 \subseteq V$ form a partition of V, and $\Omega : V \to \{0, \ldots, k-1\}$ assigns to each node a priority. The two players move a token around the graph, with Player 0 moving when the token is on a node in V_0 and Player 1 when it is on V_1. The edges E determine the possible moves. To determine the winner, we look at the sequence of priorities $\Omega(v_i)$ occurring in an infinite play $v_0 v_1 \ldots$. Player 0 wins if the smallest priority occurring infinitely often is even and Player 1 wins if it is odd.

Parity games are the model-checking games not just for L_μ but also of LFP—the extension of first-order logic with an operator for forming relational least fixed points. That is, for any formula φ of LFP and any structure $\mathfrak{A}$ one can easily construct a game $\mathcal{G}(\mathfrak{A}, \varphi)$ where Player 0 has a winning strategy if, and only if, the formula φ is satisfied in $\mathfrak{A}$. The game arena is essentially obtained as the product of $\mathfrak{A}^w$ and φ, where w is the width of the formula—the maximal arity of a relation defined by a subformula of φ. Furthermore, for any fixed number k, the class of parity games with k priorities in which Player 0 has a winning strategy is itself definable in L_μ and therefore by an LFP formula of width 2. This tight correspondence between games and the fixed-point logic leads us to describe parity games as the "right" model-checking games for LFP.

LFP is not the only logic that extends first-order logic with a means of forming fixed points. 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. Here we are mainly concerned with IFP, the logic of *inflationary fixed points* (see Section 3.1 for a definition). In the context of 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 that the two logics are equally expressive even without the restriction to finite structures [8]. However, it has also recently been shown that MIC, the extension of propositional modal logic by inflationary fixed points, is vastly more expressive than the modal μ-calculus L_μ [2] and that LFP and IFP have very different structural properties even when they have the same expressive power [8]. 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.

The correspondence between parity games and logics with least and greatest fixed point operators rests on the structural property of *well-foundedness*. A proponent in a game who is trying to prove that a certain element x belongs to a least fixed point X, needs to present a well-founded justification for its inclusion. That is, the inclusion of x in X may be based on the inclusion of other elements in X whose inclusion in turn needs to be justified but the entire process must be well-founded. On the other hand, 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 an element x enters the construction of the fixed point X 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 x can be justified at some stage, it can be justified at all later stages. In contrast, in constructing an inflationary fixed point, if x 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, 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 2 below. The backtracking games we define are far more complex than parity games. We prove that winning strategies are necessarily more complicated, requiring unbounded memory, in contrast to the memoryless 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 priorites. In contrast, parity games are known to be decidable in NP $\cap$ Co-NP and in PTIME when the number of priorities is fixed. In Section 3 we show that 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. In Section 4 we show that in IFP we can define the class of simple backtracking games that are won by Player 0. Thus, we obtain a tight correspondence between the game and the logic, as exists between LFP and parity games.

2 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.

Definition 2.1 *The arena $\mathcal{G} := (V, E, V_0, V_1, B, \Omega)$ of a backtracking game is a directed graph (V, E), with a partition $V = V_0 \cup V_1$ of V into positions of Player 0 and positions of Player 1, a subset $B \subseteq V$ of backtrack positions and a map $\Omega : V \to \{0, \dots, k-1\}$ that assigns to each node a priority.*

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* 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 $v_0 v_1 \ldots$ of the positions that have been visited. For instance, if a player backtracks from v_n in $v_0 \ldots v_i \ldots v_j \ldots v_n$, it matters whether she backtracks to i or j, even if $v_i = v_j$ because the associated numbers $d(p)$ are different.

We now proceed to a more formal description of how backtracking games are played. We distinguish therefore between the notion of a *(partial) play*, which is a word $\pi \in (V \cup \mathbb{N})^{\leq\omega}$ and the sequence $\mathrm{path}(\pi)$ of nodes visited by π. Further, we associate with every partial play π a function $d_\pi : \{0, \ldots, k-1\} \to \mathbb{N} \cup \{\infty\}$ associating with every priority p the distance $d_\pi(p)$. Here $d(p) = \infty$ means that p is not active: either there never has been a backtracking move of priority p, or the priority p has since been released by a smaller priority. Every occurrence of a node with priority p decrements $d_\pi(p)$, with the convention that $\infty - 1 = \infty$. A play π cannot be extended if $d_\pi(p) = 0$ for some p.

Definition 2.2 (Playing backtracking games) *Let $\mathcal{G} = (V, E, V_0, V_1, B, \Omega)$ be a backtracking game with priorities $\{0, \ldots, k-1\}$, and $v_0 \in V$. The set of* partial plays π *from position v_0, together with the associated sequence* $\mathrm{path}(\pi)$ *of the visited positions and the distance function $d_\pi : \{0, \ldots, k-1\} \to \mathbb{N} \cup \{\infty\}$, are inductively defined as follows.*

start: *v_0 is a partial play, with $\mathrm{path}(v_0) = v_0$, and $d_{v_0}(p) = \infty$ for all p.*

ordinary move: *If π is a partial play with $d_\pi(p) > 0$ for all p, $\mathrm{path}(\pi) = v_0 \ldots v_n$ and $v_n \in V_\sigma$, then Player σ can extend π to πv for each $v \in v_n E$; Further, $\mathrm{path}(\pi v) = \mathrm{path}(\pi) v$ and $d_{\pi v}(p) := d_\pi(p)$ for $p < \Omega(v)$, $d_{\pi v}(p) := d_\pi(p) - 1$ for $p = \Omega(v)$, and $d_{\pi v}(p) := \infty$ for $p > \Omega(v)$.*

backtracking move: *Suppose that π is a partial play with $d_\pi(p) > 0$ for all p and that $\mathrm{path}(\pi) = v_0 \ldots v_n$ with $v_n \in V_\sigma \cap B$, $\Omega(v_n) = q$, and $d_\pi(q) = \infty$. Then Player σ can extend π to πi for any number $i < n$ such that $\Omega(v_i) = q$ and $\Omega(v_k) \geq q$ for all k with $i < k < n$. Further $\mathrm{path}(\pi i) = \mathrm{path}(\pi) v_i$ and $d_{\pi i}(p) := d_\pi(p)$ for $p < q$, $d_{\pi i}(p) := |\{k : i \leq k < n : \Omega(v_k) = q\}|$ for $p = q$, and $d_{\pi i}(p) := \infty$ for $p > q$.*

Definition 2.3 (Winning condition) *A partial play π with $\mathrm{path}(\pi) = v_0 \ldots v_n$ is won by Player σ, if $v_n \in V_{1-\sigma}$ and no move is possible. This is the case if either $d_\pi(p) = 0$ for some p, or if $v_n E$ is empty and no backtracking move is possible from π. An infinite play π is won by Player 0 if the smallest priority occurring infinitely often on $\mathrm{path}(\pi)$ is even; otherwise π is won by Player 1.*

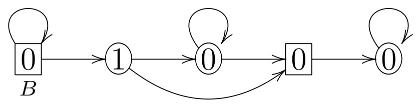
A game is *determined* if from each position one of the two players has a winning strategy. Determinacy of backtracking games follows from general facts on infinite games. Indeed, by Martin's Theorem [9] all Borel games are determined, and it is easy to see that backtracking games are Borel games.

Proposition 2.4 *Backtracking games are determined.*

Backtracking games generalise parity games. Indeed a parity game is a backtracking game without backtrack positions. Since parity games are determined via positional (i.e. memoryless) winning strategies, the question arises whether this also holds for backtracking games. We present a simple example to show that this is not the case. In fact, no fixed amount of finite memory suffices. For background on positional and finite-memory strategies we refer to [7].

Theorem 2.5 *Backtracking games in general do not admit finite-memory winning strategies.*

Proof. Consider the following game (where circles are positions of Player 0 and boxes are positions of Player 1) and the numbers indicate priorities.



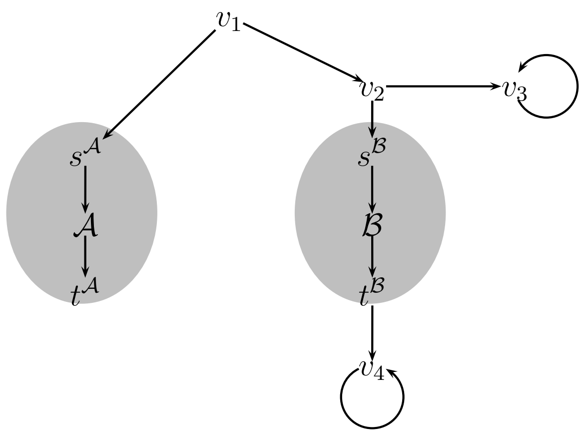
We claim that Player 0 wins from the leftmost position, but needs infinite memory to do so. Clearly, if Player 1 never leaves the leftmost position, or if she leaves it before doing a backtracking move, then Player 0 wins seeing priority 0 infinitely often. If Player 1 at some point backtracks at the leftmost position and then moves on, the strategy of Player 0 depends on the value of $d(0)$ to make sure that the fourth node is hit at the point when $d(0) = 0$. But as Player 1 can make $d(0)$ arbitrarily large, no finite-memory strategy suffices for Player 0. □

This result establishes that winning strategies for backtracking games are more complex than the strategies needed for parity games. It is also the case that the computational complexity of deciding which player has a winning strategy is also 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), we shall see below that the corresponding problem for backtracking games is PSPACE-hard. Further, for any fixed number of priorities, parity games can be decided in PTIME, but we show that backtracking games with just two priorities are already NP-hard. This is shown by reduction from the language equivalence problem for finite automata over a unary alphabet, which is known to be Co-NP-hard [4]. As the problem of deciding the winner of a backtracking game is closed under complementation, it is also NP-hard.

Theorem 2.6 *Deciding the winner of backtracking games is* Co-NP *and* NP-*hard, for games with only two priorities.*

Proof. Consider the problem of deciding, for two given directed graphs $\mathcal{A}$ and $\mathcal{B}$ with distinguished pairs of nodes $(s^{\mathcal{A}}, t^{\mathcal{A}})$ and $(s^{\mathcal{B}}, t^{\mathcal{B}})$ (which you may think of as automata over a unary alphabet with initial states s and final states t), whether the possible lengths of paths between $s^{\mathcal{A}}$ and $t^{\mathcal{A}}$ in $\mathcal{A}$ and $s^{\mathcal{B}}$ and $t^{\mathcal{B}}$ in $\mathcal{B}$ are the same. This problem is called the *unary trace* or *language equivalence problem* and is known to be Co-NP-hard [4, problem AL1]. Here, we use the variant where it is only verified that all possible lengths of paths between $s^{\mathcal{A}}$ and $t^{\mathcal{A}}$ are also possible between $s^{\mathcal{B}}$ and $t^{\mathcal{B}}$. Clearly, this is Co-NP-hard too. It is this *language inclusion* problem we are going to reduce to backtracking games with two priorities.

For any given pair $(\mathcal{A}, s^{\mathcal{A}}, t^{\mathcal{A}})$, $(\mathcal{B}, s^{\mathcal{B}}, t^{\mathcal{B}})$ of graphs we construct a backtracking game $\mathcal{G}_{\mathcal{A},\mathcal{B}} := (V, E, V_0, V_1, B, \Omega)$ such that Player 0 wins the game if, and only if, for any path in $\mathcal{A}$ between $s^{\mathcal{A}}$ and $t^{\mathcal{A}}$ of length n there also is a path of that length between $s^{\mathcal{B}}$ and $t^{\mathcal{B}}$ in $\mathcal{B}$. The arena is formally defined as follows. The set of positions is $V := V_A \cup V_B \cup \{v_1, v_2, v_3, v_4\}$, where V_A and V_B are the state sets of the automata $\mathcal{A}$ and $\mathcal{B}$ respectively. The positions $v_1, t^{\mathcal{B}}$ and all nodes in V_A belong to Player 1 and all other positions belong to Player 0. Further, the positions v_2 and v_4 have priority 1 and all other positions have priority 0. Finally, the set of backtrack positions B only contains $t^{\mathcal{A}}$. The game starts at v_1.



We claim that Player 0 wins the game if, and only if, for every path from $s^{\mathcal{A}}$ to $t^{\mathcal{A}}$ there exists a path of the same length from $s^{\mathcal{B}}$ to $t^{\mathcal{B}}$. Consider a play starting at v_1. As $v_1 \in V_1$, Player 1 moves first. If she goes to v_2, she loses immediately as then Player 0 can move to v_3 and loop forever. As $\Omega(v_3) = 0$, Player 0 wins this play. Thus Player 1 has to move from v_1 to $s^{\mathcal{A}}$. All positions in the graph $\mathcal{A}$ belong to 1 but have priority 0. So if Player 1 stays in $\mathcal{A}$ forever, she will lose. The only chance for her to win is to choose a path from $s^{\mathcal{A}}$ to

$t^{\mathcal{A}}$ and then backtrack to v_1. Let n be the length of this path. Thus, after the backtracking move, the play continues at v_1 with n positions of priority 0 left to be played. Now, Player 1 can not again move to $s^{\mathcal{A}}$ as she will lose in this case. Thus, she has to choose v_2 as the next position. Now, it is Player 0's choice. She no longer can go to v_3 but has to continue with $s^{\mathcal{B}}$. Now, the only chance for Player 0 to win this play is to find a path of length n from $s^{\mathcal{B}}$ to $t^{\mathcal{B}}$ and follow it. If there is such a path, then after n positions of priority 0 the play will stop at $t^{\mathcal{B}}$ with Player 1 to move next. Thus 1 loses. On the other hand, if there is no such path, then Player 0 loses. Thus, Player 1 wins the game if, and only if, there is a path of length n from $s^{\mathcal{A}}$ to $t^{\mathcal{A}}$ but there is no such path between $s^{\mathcal{B}}$ and $t^{\mathcal{B}}$. $\square$

3 Model checking games for inflationary fixed point logic

In this section we want to show that backtracking games can be used as model checking games for inflationary fixed point logics. We will present the games in terms of IFP, the extension of first-order logic by inflationary and deflationary fixed points, but the construction applies, with the obvious modifications, also to the modal iteration calculus MIC [2].

3.1 Inflationary fixed point logic

A formula $\varphi(R, \boldsymbol{x})$ with a free k-ary second-order variable and a free k-tuple of first-order variables $\boldsymbol{x}$ defines, on every structure $\mathfrak{A}$, a relational operator $F_\varphi : \mathcal{P}(A^k) \to \mathcal{P}(A^k)$ taking $R \subseteq A^k$ to the set $\{\boldsymbol{a} : (\mathfrak{A}, R) \models \varphi(\boldsymbol{a})\}$. Fixed point extensions of first-order logic are obtained by adding to FO explicit constructs to form fixed points of definable operators. The type of fixed points that are used determines the expressive power and also the algorithmic complexity of the resulting logics. The most important of these extensions are least fixed point logic (LFP) and inflationary fixed point logic (IFP).

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 $R^0 := \varnothing$, $R^{\alpha+1} := R^\alpha \cup F(R^\alpha)$, 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.

Definition 3.1 Inflationary fixed-point logic (IFP) *is obtained from* FO *by allowing formulae of the form* $[\mathbf{ifp}\, R\boldsymbol{x} \,.\, \varphi(R, \boldsymbol{x})](\boldsymbol{x})$ *and* $[\mathbf{dfp}\, R\boldsymbol{x} \,.\, \varphi(R, \boldsymbol{x})](\boldsymbol{x})$, *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

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

For some applications (one of which will appear in Section 4) one is interested to have 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 $\preccurlyeq$ on $\mathcal{K}$ such that $u \sim v$ iff $u \preccurlyeq v$ and $v \preccurlyeq u$. We can again do this via a fixed point construction, by defining a sequence $\preccurlyeq_\alpha$ of pre-orders (where α ranges over ordinals) such that $\preccurlyeq_{\alpha+1}$ refines $\preccurlyeq_\alpha$ and $\preccurlyeq_\lambda$, for limit ordinals λ, is the intersection of the pre-orders $\preccurlyeq_\alpha$ with $\alpha < \lambda$. Let

$$u \preccurlyeq_1 v :\Longleftrightarrow \bigwedge_{i \le m} P_i u \to \Big(P_i v \vee \bigvee_{j<i} (\neg P_j u \wedge P_i v) \Big)$$

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

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

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 $u \preccurlyeq_{\alpha+1} v$ if, and only if, $u \preccurlyeq_\alpha v$ and there is an edge from v (and hence none from u) into the smallest $\sim_\alpha$-class (wrt. $\preccurlyeq_\alpha$) that separates u from v (if it exists). Since the sequence of the pre-orders $\preccurlyeq_\alpha$ is decreasing, it must indeed reach a fixed point $\preccurlyeq$, 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 $\preccurlyeq$ is a deflationary fixed point of a non-monotone induction. Indeed, the refinement operator on pre-orders is not monotone and does, in general, not 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 relativizes $\mathfrak{A}$ 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 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 [8]. There seems to be no simple direct definition based on least and greatest fixed points only.

Knowledge and Public Announcement. Iterated relativisation has a natural meaning also in epistemic logics, i.e. logics of knowledge. For background we refer to [3]. Basic epistemic logic (for a group A of agents and a set of atomic propositions $\{P_b : b \in B\}$) is just propositional modal logic, interpreted on possible-world models, i.e., Kripke structures $\mathcal{K} = (V, (E_a : a \in A), (P_b : b \in B))$, where each possibility relation E_a is an equivalence relation. The intended meaning of $[a]\varphi$ is "agent a knows φ", which is true in a world $v \in V$ if φ holds in all worlds w that agent a considers possible in world v.

A key concept in epistemic logics is *common knowledge*. A proposition φ is common knowledge at a world v (in short: $\mathcal{K}, v \models C\varphi$) if everybody knows φ, and everybody knows that everybody knows φ, and everybody knows that everybody knows that everybody knows .... Clearly, common knowledge is a greatest fixed point. In the modal μ-calculus, $C\varphi$ is defined by $\nu X.\varphi \wedge$

$\bigwedge_{a\in A}[a]X$.

Suppose now that somebody (who is trusted by all agents) publicly announces φ. One would think that by this action, φ has become common knowledge, since everybody has learned that φ is true and everybody has learned that everybody has learned, and so on. Indeed, the announcement changes the state of knowledge of the agents, and thus induces an update of the model: all worlds which currently do not satisfy φ are eliminated, in other words, $\mathcal{K}$ is relativised to φ. Epistemic logics with public announcement (as considered for instance in [1,10]) admit formulae $[\varphi!]\psi$ expressing that ψ holds after announcement of φ, i.e., after the model has been relativised to φ. Of course this can easily be captured via syntactic relativisation so it does not go beyond basic epistemic logic (if common knowledge is present, it has to be expanded as a greatest fixed point before relativisation).

However, it is important to note that in the updated model $\mathcal{K}|_\varphi$, φ is *not* necessarily common knowledge. Consider announcements involving ignorance like $\neg[a][b]\psi$ ("a considers it possible that b does not know ψ"). Removal of those worlds where this is false may have the effect that at others, agent a now knows that b knows, so the announced statement becomes false there by its very announcement. But if somebody keeps announcing φ after each relativisation step, we have a process of iterated relativisation that will eventually restrict the model to the deflationary fixed point $(\mathbf{dfp}\, X \leftarrow \varphi|_X)$. We can again ask if this fixed point is definable by monotone inductions, but this time in a more specific scenario.

Johan van Benthem has aked whether iterated relativisations of formulae from basic epistemic logic (with or without common knowledge) are definable in the modal μ-calculus? In [6] we have shown that this is not the case.

3.3 Model checking games for LFP

Let us recall the definitions of model checking games for least fixed-point logic LFP (the games for the modal μ-calculus are analogous). Consider a sentence $\psi \in$ LFP which we assume is in negation normal form and *well-named*, i.e. every fixed-point variable is bound only once.

The game $\mathcal{G}(\mathfrak{A}, \psi)$ is a parity game whose positions are subformulae of ψ instantiated by elements of $\mathfrak{A}$, i.e. expressions $\varphi(\boldsymbol{a})$ such that $\varphi(\boldsymbol{x})$ is a subformula of ψ, and $\boldsymbol{a}$ a tuple of elements of $\mathfrak{A}$. Player 0 (Verifier) moves at positions associated with disjunctions and formulae $\exists y\varphi(\boldsymbol{a}, y)$. From a position $(\varphi\vee\vartheta)(\boldsymbol{a})$ she moves to either $\varphi(\boldsymbol{a})$ or $\vartheta(\boldsymbol{a})$ and from a position $\exists y\varphi(\boldsymbol{a}, y)$ she can move to any position $\varphi(\boldsymbol{a}, b)$ such that $b \in \mathfrak{A}$. In addition, Verifier is supposed to move at atomic false positions, i.e., at positions $R\boldsymbol{a}$ where $\boldsymbol{a} \notin R^{\mathfrak{A}}$

and $\neg R\boldsymbol{a}$ where $\boldsymbol{a} \in R^{\mathfrak{A}}$. However, these positions do not have successors, so Verifier loses at atomic false positions. Dually, Player 1 (Falsifier) moves at conjunctions and formulae $\forall y \varphi(\boldsymbol{a}, y)$, and loses at atomic true positions. The rules described so far determine the model checking game for FO-formulae ψ and it is easily seen that Verifier has a winning strategy in this game $\mathcal{G}(\mathfrak{A}, \psi)$ starting at a position $\varphi(\boldsymbol{a})$ if, and only if, $\mathfrak{A} \models \varphi(\boldsymbol{a})$.

For formulae in LFP, we also have positions $[\mathbf{fp}\, T\boldsymbol{x}.\varphi](\boldsymbol{a})$ (where **fp** stands for either **lfp** or **gfp**) and $T\boldsymbol{a}$, for fixed-point variables T. At these positions there is a unique move (by Falsifier, say) to $\varphi(\boldsymbol{a})$, i.e. to the formula defining the fixed point. The priority labelling assigns even priorities to **gfp**-atoms $T\boldsymbol{a}$ and odd priorities to **lfp**-atoms $T\boldsymbol{a}$. Further, if T, T' are fixed-point variables of different kind with T' depending on T (which means that T occurs free in the formula defining T'), then T-positions get lower priority than T'-positions. The remaining positions, not associated with fixed-point variables, do not have a priority (or have the maximal one). As a result, the number of priorities in the model checking game equals the alternation depth of the fixed-point formula plus one. For more details and explanations, and for the proof that the construction is correct, see e.g. [5,11].

Theorem 3.2 *For all formulae $\psi \in$ LFP and all structures $\mathfrak{A}$, $\mathfrak{A} \models \psi$ if, and only if, Verifier has a winning strategy for the parity game $\mathcal{G}(\mathfrak{A}, \psi)$ from position ψ.*

If LFP-formulae have unbounded width the game graph of $\mathcal{G}(\mathfrak{A}, \psi)$ may become rather large; indeed the model checking problem for LFP is EXPTIME-complete, even for rather simple formulae, with just one fixed point. For LFP-formulae where both the alternation depth and the width are bounded, the model checking problem can be solved in polynomial time (for instance via solving the model checking game). The important unresolved case concerns LFP-formulae with bounded width, but unbounded alternation depth. This includes the μ-calculus, since every formula of L_μ can be translated into an equivalent LFP-formula of width two. In fact 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 sets 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.

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 $\varphi^*(\boldsymbol{x}) := [\mathbf{ifp}\, T\boldsymbol{x} \,.\, \varphi(T, \boldsymbol{x})](\boldsymbol{x})$ 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 $T\boldsymbol{x} \vee \varphi$ and $\overline{T}\boldsymbol{x} \wedge \overline{\varphi}$. 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 $[\mathbf{ifp}\, R\boldsymbol{y} \,.\, \vartheta(R, \overline{R}, \boldsymbol{y})](\boldsymbol{y})$ in φ will correspond to a subformula $[\mathbf{dfp}\, R'\boldsymbol{y} \,.\, \overline{\vartheta}(\overline{R'}, R', \boldsymbol{y})](\boldsymbol{y})$ of $\overline{\varphi}$ where R' is a new relation variable, distinct from R.

From a position $\varphi^*(\boldsymbol{a})$ the play proceeds to $T\boldsymbol{a} \vee \varphi(T, \boldsymbol{a})$. When a play reaches a position $T\boldsymbol{c}$ or $\overline{T}\boldsymbol{c}$ the play proceeds back to the formula defining the fixed point by a regeneration move. More precisely, the regeneration of an **ifp**-atom $T\boldsymbol{c}$ is $T\boldsymbol{c} \vee \varphi(T, \boldsymbol{c})$, the regeneration of $\overline{T}\boldsymbol{c}$ is $\overline{T}\boldsymbol{c} \wedge \overline{\varphi}(T, \boldsymbol{c})$. Verifier can move from $T\boldsymbol{c}$ to its regeneration, Falsifier from $\overline{T}\boldsymbol{c}$. For **dfp**-subformulae $\vartheta^*(\boldsymbol{x}) := [\mathbf{dfp}\, R\boldsymbol{x} \,.\, \vartheta(R, \boldsymbol{x})](\boldsymbol{x})$, dual definitions apply. Verifier moves from $\overline{R}\boldsymbol{c}$ to its regeneration $\overline{R}\boldsymbol{c} \vee \overline{\vartheta}(R, \boldsymbol{c})$, and Falsifier can make regeneration moves from $R\boldsymbol{c}$ to $R\boldsymbol{c} \wedge \vartheta(R, \boldsymbol{c})$. 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 $S\boldsymbol{c}$ and $\overline{S}\boldsymbol{c}$ 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, $\psi := [\mathbf{ifp}\, T\boldsymbol{x} \,.\, \varphi](\boldsymbol{a})$ where $\varphi(T, \boldsymbol{x})$ is a first-order formula. When the play reaches a position $T\boldsymbol{c}$ Verifier can make a regeneration move to $T\boldsymbol{c} \vee \varphi(T, \boldsymbol{c})$ or backtrack. Dually, Falsifier can regenerate from positions $\overline{T}\boldsymbol{c}$ 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 $T\boldsymbol{c}$, and Verifier has won if it is of form $\overline{T}\boldsymbol{c}$.

The differences between IFP model checking and LFP model checking are in fact best illustrated with this simple case. For this reason, we give a full proof of the correctness of the model checking game for this case only.

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 $\mathfrak{A} \not\models \psi$.

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

The first-order game $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\boldsymbol{a}))$ can be seen as an unfolding of the game $\mathcal{G}(\mathfrak{A}, \psi(\boldsymbol{a}))$. Every position in $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\boldsymbol{a}))$ corresponds to a unique position in $\mathcal{G}(\mathfrak{A}, \psi(\boldsymbol{a}))$, and conversely, for a pair (p, β) where p is a position of $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\boldsymbol{a}))$ and $\beta \leq \alpha$ is an ordinal, there is a unique associated position p_β of the unfolded game $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\boldsymbol{a}))$. When a play in $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\boldsymbol{a}))$ reaches a position $T\boldsymbol{c}$, it is regenerated to either $T\boldsymbol{c}$ or $\varphi(T, \boldsymbol{c})$ and such regeneration move decrements the associated ordinal. The corresponding play in $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\boldsymbol{a}))$ proceeds to position $\varphi^\beta(\boldsymbol{c})$ or $\varphi[T/\varphi^\beta, \overline{T}/\overline{\varphi}^\beta](\boldsymbol{c})$. 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 $\mathcal{G}(\mathfrak{A}, \varphi^\alpha(\boldsymbol{a}))$ will produce a non-positional strategy f^* in the original game $\mathcal{G}(\mathfrak{A}, \psi)$: start with $\beta = \alpha$ and let, 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 $\mathfrak{A} \models \varphi^\beta(\boldsymbol{a})$. Then Verifier has a winning strategy in the first-order game $\mathcal{G}(\mathfrak{A}, \varphi^\beta(\boldsymbol{a}))$ (from position $\varphi^\beta(\boldsymbol{a})$) 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 $\mathfrak{A} \not\models \varphi^\beta(\boldsymbol{a})$ then Falsifier has a winning strategy for $\mathcal{G}(\mathfrak{A}, \varphi^\beta(\boldsymbol{a}))$, 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 $\psi(\boldsymbol{a})$ 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 3.3 *Suppose that a play on $\mathcal{G}(\mathfrak{A}, \psi)$ has been backtracked to the initial position $\psi(\boldsymbol{a})$ after β T-positions have been visited. Verifier has a winning strategy for the remaining game if, and only if, $\mathfrak{A} \models \varphi^\beta(\boldsymbol{a})$.*

From this we obtain the desired result.

Theorem 3.4 *If $\mathfrak{A} \models \psi(\boldsymbol{a})$, then Verifier wins the game $\mathcal{G}(\mathfrak{A}, \psi(\boldsymbol{a}))$ from position $\psi(\boldsymbol{a})$. If $\mathfrak{A} \not\models \psi(\boldsymbol{a})$, then Falsifier wins the game $\mathcal{G}(\mathfrak{A}, \psi(\boldsymbol{a}))$ from position $\psi(\boldsymbol{a})$.*

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

From $\psi(\boldsymbol{a})$ the game proceeds to $(T\boldsymbol{a} \vee \varphi(\boldsymbol{a}))$. At this position, Verifier repeatedly chooses the node $T\boldsymbol{a}$ until this node has been visited α-times. After that, she backtracks and moves to $\varphi(\boldsymbol{a})$. By Lemma 3.3 and since $\mathfrak{A} \models \varphi^\alpha(\boldsymbol{a})$, Verifier has a strategy to win the remaining play.

Now suppose that $\mathfrak{A} \not\models \psi(\boldsymbol{a})$. If, after α T-positions, one of the players backtracks, then Falsifier has a winning strategy for the remaining game, since $\mathfrak{A} \not\models \varphi^\alpha(\boldsymbol{a})$. Hence, the only possibility for Verifier to win the game in a finite number of moves is to avoid positions $\overline{T}\boldsymbol{b}$ where Falsifier can backtrack.

Consider the formulae φ_f^α, with $\varphi_f^0 = \mathit{false}$ and $\varphi_f^{\alpha+1}(\boldsymbol{x}) = \varphi[T/\varphi_f^\alpha, \overline{T}/\mathit{false}](\boldsymbol{x})$. They define the stages of $[\mathbf{ifp}\, T\boldsymbol{x} \,.\, \varphi[T, \mathit{false}](\boldsymbol{x})]$, 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 $T\boldsymbol{c}$ and without positions $\overline{T}\boldsymbol{c}$, then she would in fact have a winning strategy for the model checking game $\mathcal{G}(\mathfrak{A}, \varphi_f^\alpha(\boldsymbol{a}))$. Since ψ_f^α implies φ^α, it would follow that $\mathfrak{A} \models \varphi^\alpha(\boldsymbol{a})$. But this is impossible.
□

The extension of the proof of Theorem 3.4 to arbitrary IFP-formulae poses no major difficulties although a detailed exposition would be quite lengthy. 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 side, 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.

Theorem 3.4 allows us also to draw a consequence regarding the complexity of backtracking games. In [2] it is shown that the model-checking problem for MIC is PSPACE-complete. As the coding of the model-checking problem for IFP, and hence MIC, into backtracking games described above clearly constitutes a polynomial-time reduction, we have the following.

Corollary 3.5 *Deciding the winner of a backtracking game is* PSPACE*-hard.*

It is natural to ask whether this lower bound is optimal. We do not know whether backtracking games are decidable in PSPACE. If it could be shown that there is a polynomial bound on the maximum distance that a player might need to backtrack, we would obtain such an upper bound. However, this is not the case. The construction in the proof of Theorem 2.6 shows that an exponential amount of backtracking is necessary. Indeed, it is known that there is a family of pairs of nondeterministic automata in a one-letter alphabet such that the shortest string distinguishing the languages accepted by a pair is exponential in the size of the automata. This follows from the construction showing the NP-hardness of the language equivalence problem in [12].

4 Definability of Backtracking Games

In the previous section we demonstrated that backtracking games can be used as model-checking games for IFP. The aim of this section is to show that they are, in some sense, the "right" model-checking games for inflationary fixed-point logics. For this, we identify a natural sub-class of backtracking games, which we call *simple*, such 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 $k \in \mathbb{N}$ there is a formula $\varphi \in$ IFP defining the winning region for Player 0 in any simple game with at most k priorities. In this sense, simple backtracking games precisely capture IFP model-checking.

Consider again the proof given in Section 3.4 for winning strategies in a 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 $[\mathbf{ifp}\, R\boldsymbol{x} \,.\, R\boldsymbol{x} \vee \varphi]$. She then looped α times through the $R\boldsymbol{x}$ sub-formula and backtracked. By choosing the α she essentially picked a stage of the fixed-point induction on φ and claimed that $\boldsymbol{x} \in \varphi^{\alpha}$. 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 $R\boldsymbol{x} \vee \varphi$ and the sub-formula $R\boldsymbol{x}$, 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.

Definition 4.1 *A strategy in a backtracking game $\mathcal{G}$ is* local *if, for any backtracking node v, all backtracking moves from 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.*

As explained above, we can easily modify the construction of the game graph $\mathcal{G}(\mathfrak{A}, \varphi)$ for a formula φ and structure $\mathfrak{A}$ such that every node in B has a self loop. We call such game graphs *inflationary.*

Definition 4.2 *A backtracking game $\mathcal{G} := (V, E, V_0, V_1, B, \Omega)$ is* inflationary*, if every node in B has a self-loop. An inflationary game $\mathcal{G}$ is called* simple *if both players have local linear winning strategies on their winning regions.*

Proposition 4.3 *For any* IFP*-formula ψ and every finite structure $\mathfrak{A}$, the model-checking game $\mathcal{G}(\mathfrak{A}, \varphi)$, as defined in Section 3.4, is simple.*

We will construct IFP-formulae defining the winning regions of simple backtracking games. Since backtracking games are extensions of parity games we start with the formula defining winning regions in parity games (see [13]). Let $\mathcal{G}$ be a parity game with $k + 1$ priorities and consider the formula

$$\varphi(x) := [\mathbf{gfp}\, R_0 x\, .\, \mathbf{lfp}\, R_1 x\, .\, \ldots \mathbf{fp}\, R_k x\, .\, \vartheta(x, R_0, \ldots, R_k)](x),$$

where

$$\begin{aligned}\vartheta(x, R_0, \ldots, R_k) := {} & \textstyle\bigwedge_{i=0}^{k}(V_0 x \wedge \Omega(x) = i \to \exists y\, (Exy \wedge R_i y)) \wedge \\ & \textstyle\bigwedge_{i=0}^{k}(V_1 x \wedge \Omega(x) = i \to \forall y\, (Exy \to R_i y)).\end{aligned}$$

For every node $v \in V$, we have that $\mathcal{G} \models \varphi(v)$ if, and only if, Player 0 has a winning strategy for the game $\mathcal{G}$ from v. A simple way to see this is to analyse the model checking game for $\varphi(v)$ on $\mathcal{G}$. If we remove the edges which would force a player to lose immediately, we obtain $\mathcal{G}$ itself (from position v).

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.

Lemma 4.4 *There is a formula $\varphi_{ord}(x, y) \in \text{IFP}$ defining on every bisimulation minimal arena a linear order.*

This is well-known in finite model theory. An explicit construction has been given in Section 3.2. 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.

In Theorem 2.5 we exhibited a backtracking game that requires infinite memory strategies. All strategies in this game are necessarily local. Thus Theorem 2.5 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 occured, 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 $(v, d_\pi(0), \dots, d_\pi(k))$, where d_π denotes the distance function as defined in Definition 2.2. This is formalised in the notion of a *configuration*.

Definition 4.5 *Let $\mathcal{G}$ be a backtracking game with $k+1$ priorities. A* configuration *is a pair $(v, \boldsymbol{d})$ consisting of a node v and a tuple $\boldsymbol{d} \in (\mathbb{N} \cup \{\infty\})^{k+1}$. Let π be a (partial) play ending in node v. The* configuration of π *is defined as the tuple $(v, d_\pi(0), \dots, d_\pi(k))$.*

We are now ready to present a formula defining the winning region for Player 0 in a simple backtracking game with priorities $0, \dots, k$. For this recall that in a simple backtracking game the distance of all backtracking moves is at most n, where $n := |\mathcal{G}|$ is the number of nodes in the game graph $\mathcal{G}$. Furthermore, by Lemma 4.4, 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 $(v, \boldsymbol{d})$ where $\boldsymbol{d} \in \{0, \dots, n, \infty\}^{k+1}$ and we can use nodes in the game graph to represent the values of the d_i. Note that strictly speaking we need to encode each d_i by a pair of elements, as the d_i can take values between 0 and n and may also take the value ∞. However, to simplify notation, we only use one variable for each d_i and allow it to take all possible values.

The structure of the formula is similar to the structure of $\varphi(x)$ for parity games, in the sense that for games with $k{+}1$ priorities we have $k{+}1$ nested fixed points of the form $\mathbf{gfp}\, R_0 x\boldsymbol{d}\,.\, \mathbf{lfp}\, R_1 x\boldsymbol{d}\,.\,\dots \mathbf{fp}\, R_k x\boldsymbol{d}$ and a ψ 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 $(x, d_0, \dots, d_k)$ 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 $0 \le j \le k$, to a winning play.

The inner formula ψ is split in two parts $\psi_0 \vee \psi_1$ taking care of positions where Player 0 moves and positions where Player 1 moves. We first present the formula $\psi_0(x, R_0, \dots, R_k)$ defining positions in V_0 from which Player 0 can win. As explained above, we encode elements from the set $\{0, \dots, n, \infty\}$ by a single variable instead of pairs of variables. Further, we will use symbols $\mathtt{i}$, $\mathtt{j}$, ... in typewriter font to denote constants between 0 and k. Finally, in the case distinctions below we write $d_i = m$ for $\exists m \in \{0, \dots, n\} \wedge d_i = m$.

$$
\begin{aligned}
\psi_0(x, \boldsymbol{d}) := \;& V_0 x \wedge \textstyle\bigvee_{\mathtt{i}} \Omega(x) = \mathtt{i} \wedge \textstyle\bigwedge_{l=i+1}^{k} d_l = \infty \;\wedge \\
& \exists y \exists \boldsymbol{d}' Exy \wedge \textstyle\bigvee_{\mathtt{j}} \Omega(y) = \mathtt{j} \wedge R_j y \boldsymbol{d}' \wedge \\
& d_i = \infty \wedge \boldsymbol{d}' = (d_0, \dots, d_j, \infty, \dots, \infty) \;\vee \\
& \qquad\qquad Bx \wedge \exists m \neq \infty\, R_i(x, d_0, \dots, d_{i-1}, m, \infty, \dots, \infty) \;\vee \\
& d_i = m \wedge \;\; \mathtt{j} < \mathtt{i} \wedge \boldsymbol{d}' = (d_0, \dots, d_j, \infty, \dots, \infty) \;\vee \\
& \qquad\qquad \mathtt{j} = \mathtt{i} \wedge \boldsymbol{d}' = (d_0, \dots, d_{i-1}, m-1, \infty, \dots, \infty) \;\vee \\
& \qquad\qquad \mathtt{j} > \mathtt{i} \wedge \boldsymbol{d}' = \boldsymbol{d}
\end{aligned}
$$

The first line of the formula states that x has to be in V_0, the priority of x is i, for some i, and the tuple $(d_0, \dots, d_k)$ has ∞ at all positions greater than i. This corresponds to the fact that a node of priority i releases all backtracking moves on higher priorities. Now, Player 0 can win from configuration $(x, \boldsymbol{d})$ if she can move to a successor y of x from which she wins the play. Winning from y means that the configuration $(x, \boldsymbol{d}')$ reached from $(x, \boldsymbol{d})$ after moving to y is in $R_{\Omega(y)}$. The second line of the formula states the existence of such a successor y and the rest of the formula defines what it means for $(y, \boldsymbol{d}')$ to be the configuration reached from x when moving to y.

The remaining part of the formula consists of two somewhat independent subformulae. The first, where $d_i = \infty$, consists of a case distinction taking the various options for Player 0 to win into account: She can make an ordinary move to a successor y of x from which she can win. In this case there must be a successor y and a tuple $\boldsymbol{d}'$ such that $(y, \boldsymbol{d}') \in R_{\Omega(y)}$, i.e. Player 0 wins from y, and $(y, \boldsymbol{d}')$ is the configuration reached when Player 0 moves from x with configuration $(x, \boldsymbol{d})$ to y. Alternatively, she can decide to backtrack, provided that $x \in B$. Then there must be a number $m \leq n = |\mathcal{G}|$ such that Player 0 wins the m-step game from x.

The second part, where $d_i = m$ for some $m \leq n$, defines the numbers m such that Player 0 wins the m-step game on priority i from node 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 this case the play continues with the configuration $(y, d_0, \ldots, d_i, \infty, \ldots, \infty)$, i.e. all active backtracking moves (whose distances are stored in $d_0, \ldots, d_i$) remain unchanged and the play continues on priority j without any active backtracking moves on priorities greater than i.

The next formula ψ_1 takes care of nodes $x \in V_1$.

$$
\begin{aligned}
\psi_1(x, \boldsymbol{d}) := \ & V_1 x \wedge \textstyle\bigvee_{\mathtt{i}} \Omega(x) = \mathtt{i} \wedge \bigwedge_{l=i+1}^{k} d_l = \infty \ \wedge \\
& (Bx \rightarrow \forall m < \infty \, R_i(x, d_0, \ldots, d_{i-1}, m, \infty, \ldots, \infty)) \ \wedge \\
& \forall y (Exy \rightarrow \textstyle\bigvee_j \Omega(y) = \mathtt{j} \wedge \exists \boldsymbol{d}' \, R_j y \boldsymbol{d}' \wedge \\
& d_i = \infty \wedge \boldsymbol{d}' = (d_0, \ldots, d_j, \infty, \ldots, \infty) \ \vee \\
& d_i = m \wedge \ \mathtt{j} < \mathtt{i} \wedge \ \boldsymbol{d}' = (d_0, \ldots, d_j, \infty, \ldots, \infty) \ \vee \\
& \qquad\qquad\ \mathtt{j} = \mathtt{i} \wedge \ \boldsymbol{d}' = (d_0, \ldots, d_{i-1}, d_i - 1, \infty, \ldots, \infty) \ \vee \\
& \qquad\qquad\ \mathtt{j} > \mathtt{i} \wedge \ \boldsymbol{d}' = \boldsymbol{d} \ \vee \\
& \qquad\qquad\ m = 0)
\end{aligned}
$$

A node $x \in V_1$ with configuration $(x, \boldsymbol{d})$ is good for Player 0 if Player 1 has no choice but to move to a node from which Player 0 wins. The formula is defined similarly to ψ_0 only that in the second line we ensure that if $x \in B$ then Player 0 must win the m-step game from x for all m, as otherwise Player 1 could backtrack and win. Further Player 0 now also wins the m-step game from x for $m = 0$.

With ψ_0 and ψ_1 defined we can now present the formula $\varphi_0(x)$ which is true for a node x in a simple backtracking game with $k+1$ priorities if, and only if, Player 0 has a linear winning strategy from x with local backtracking.

$$
\varphi_0(x) := [\mathbf{gfp}\, R_0 x \boldsymbol{d} \,.\, \mathbf{lfp}\, R_2 x \boldsymbol{d} \,.\, \ldots \mathbf{fp}\, R_k x \boldsymbol{d} \,.\, (\psi_0 \vee \psi_1)](x, \infty, \ldots, \infty)
$$

The next step is to show that the formula indeed defines the winning region for Player 0. This is done by showing that whenever for a node x the tuple $(x, \infty, \dots, \infty)$ satisfies φ_0 then Player 0 has a winning strategy for the game starting at x. For $(x, \boldsymbol{d}) \in R_0^\infty$ define $\mathrm{ord}(x, \boldsymbol{d})$ as the lexicographically smallest tuple $\boldsymbol{\alpha} := (\alpha_0, \dots, \alpha_k)$ such that $\alpha_i := \infty$ for all even i and $(x, \boldsymbol{d}) \in \psi(R_0^{\alpha_0}, \dots, R_k^{\alpha_k})$, where $\psi := \psi_0 \vee \psi_1$. We write $\mathrm{ord}_i(x, \boldsymbol{d}) := (\alpha_0, \dots, \alpha_i)$ for the tuple of stages up to position $i \leq k$.

Lemma 4.6 *For all nodes x of priority i and all $\boldsymbol{d} := (d_0, \dots, d_i, \infty, \dots, \infty)$ such that $(x, \boldsymbol{d}) \in R_0^\infty$:*

(i) If $d_i = \infty$, $x \in V_0$ and $x \notin B$ then there is a successor y of x such that

$$\begin{gathered} (y, \boldsymbol{d}') \in R_0^\infty \text{ and } \mathrm{ord}_i(y, \boldsymbol{d}') \leq \mathrm{ord}_i(x, \boldsymbol{d}) \text{ and} \\ \text{if } i \text{ is odd then } \mathrm{ord}_i(y, \boldsymbol{d}') < \mathrm{ord}_i(x, \boldsymbol{d}) \end{gathered} \tag{1}$$

where $j := \Omega(y)$ and $\boldsymbol{d}' := (d_0, \dots, d_j, \infty, \dots, \infty)$.

(ii) If $d_i = \infty$, $x \in V_0$ and $x \in B$ then there is a successor y of x such that condition (1) of Part (i) holds true for y or there is an $m \in \{0, \dots, n\}$ such that $(x, \boldsymbol{d}') \in R_0^\infty$ and $\mathrm{ord}_i(x, \boldsymbol{d}') < \mathrm{ord}_i(\boldsymbol{x}, \boldsymbol{d})$, where $\boldsymbol{d}' := (d_0, \dots, d_{i-1}, m, \infty, \dots, \infty)$.

(iii) If $d_i = \infty$ and $x \in V_1$ then (1) of Part (i) is true for all successors y of x.

(iv) If $d_i = m < \infty$ and $x \in V_0$ then there is a successor y of x such that for $j := \Omega(y)j$ condition (1) of Part (i) is true for y and $\boldsymbol{d}'$, defined by

$$\boldsymbol{d}' := \begin{cases} (d_0, \dots, d_j, \infty, \dots, \infty) & \text{if } j < i \\ (d_0, \dots, d_{i-1}, d_i', \infty, \dots, \infty) & \text{if } i \leq j \end{cases}$$

$$\text{where } d_i' := \begin{cases} d_i & \text{if } j > i \\ d_i - 1 & \text{otherwise.} \end{cases}$$

(v) Finally, if $d_i = m$, where $0 < m < \infty$ and $x \in V_1$ then the same applies to all successors y of x.

The proof of the lemma follows immediately from the construction of the formulae ψ_0 and ψ_1. The lemma allows us to define a strategy for Player 0 for all games starting from nodes x such that $(x, \infty, \dots, \infty) \in R_0^\infty$.

Strategy for Player 0: Let $\pi := v_0 \dots v_s$ be a partial play with $v_s \in V_0$ and let $(v_s, \boldsymbol{d})$, with $\boldsymbol{d} := (d_0, \dots, d_k)$, be the configuration for v_s in π. Depending on d_i and v_s, Player 0 chooses one of the successors y of x satisfying the criteria of the matching Part (i), (ii), or (iv) of Lemma 4.6.

We show next that this is indeed a winning strategy for Player 0.

Lemma 4.7 *Following the above strategy, Player* 0 *wins every play from a node* x *such that* $(x, \infty, \dots, \infty) \in R_0^\infty$.

Proof. Suppose $(x, \infty, \dots, \infty) \in R_0^\infty$ and let $\pi := v_0 \dots$ with $v_0 = x$ be a play where Player 0 plays according to the strategy outlined above. Consider the sequence of configurations $\boldsymbol{d}_0, \boldsymbol{d}_1, \dots$ induced by the play.

We prove by induction on i that $(v_i, \boldsymbol{d}_i) \in R_0^\infty$ for all v_i. This is clear for $i = 0$. We show now that if the claim holds true for v_i it is also true for v_{i+1}. For all nodes $v_i \in V_0$ this follows immediately from the definition of the strategy for Player 0. For all nodes $v_i \in V_1$ where Player 1 does not backtrack this follows from Lemma 4.6. Finally, let v_i be a node such that Player 1 backtracks m steps from v_i to v_i. Thus the game continues with the node v_{i+1} and the configuration $\boldsymbol{d}_{i+1} := (d_{i,0}, \dots, d_{i,\Omega(v_i)-1}, m, \infty, \dots, \infty)$. By induction hypothesis, $(v_i, \boldsymbol{d}_i) \in R_0^\infty$. This implies that for all $m \in \{0, \dots, n\}$, $(v_i, d_{i,0}, \dots, d_{i,\Omega(v_i)-1}, m, \infty, \dots, \infty) \in R_0^\infty$, as otherwise $(v_i, \boldsymbol{d}_i)$ would not have satisfied ψ_1. Thus, we have $(v_{i+1}, \boldsymbol{d}_{i+1}) \in R_0^\infty$. This proves the claim.

Suppose first that the play is infinite. Let i be the smallest priority occurring infinitely often in the play and let v_s be the first node after which no node of priority less than i occurs. Lemma 4.6 implies that for all $l \geq s$, $\mathrm{ord}_i(v_l, \boldsymbol{d}_l) \geq \mathrm{ord}_i(v_{l+1}, \boldsymbol{d}_{l+1})$ and furthermore, $\mathrm{ord}_m(v_l, \boldsymbol{d}_l) > \mathrm{ord}_m(v_{l+1}, \boldsymbol{d}_{l+1})$ whenever $\Omega(v_l)$ is odd. Thus, if i is odd, the ordinals up to priority i strictly decrease whenever the play reaches a node of priority i. As the ordering on the ordinals is well-founded but the game is infinite, this implies that i must be even and thus Player 0 wins the play.

Now suppose that the play is finite, i.e. $\pi := v_0 \dots v_s$. Then either v_s is a leaf or the play is terminated according to the backtracking condition. If v_s is a leaf, then using Lemma 4.6 and the construction of the strategy for Player 0, a simple induction on s shows that v_s must be a position for Player 1 and thus Player 0 wins the play.

If the play is terminated according to the backtracking condition, then there must be a node $v \in B$ with priority i such that one of the players backtracks on v and after v no node of priority less than i occurs. Let v_m be the maximal node with this property, i.e. the play is terminated by the backtracking move on v_m and priority i. Let, for all l, $\boldsymbol{d}_l$ be the configuration at node v_l and let $\boldsymbol{d}_m := (d_0, \dots, d_i, \infty, \dots, \infty)$. If $v_m \in V_0$, then, by construction of the strategy, there is a $0 \leq l \leq n$ such that $(v_m, d_0, \dots, d_{i-1}, l, \infty, \dots, \infty) \in R_0^\infty$ and Player 0 backtracks from v_m l steps. Otherwise, $v_m \in V_1$ and Player 1 backtracks l-steps for some l. As $(v_m, d_0, \dots, d_i, \infty, \dots, \infty) \in R_0^\infty$ and $v_m \in V_1 \cap B$, the formula ψ_1 ensures that for all $0 \leq j \leq n$, $(v_m, d_0, \dots, d_{i-1}, j, \infty, \dots, \infty) \in R_0^\infty$. In particular, $(v, d_0, \dots d_{i-1}, l, \infty, \dots, \infty) \in R_0^\infty$ for the value l chosen by Player 1. In either case, the game continues with configuration $\boldsymbol{d}_{m+1} =$

$(v_{m+1}, d_0, \ldots, d_{i-1}, l, \infty, \ldots, \infty)$ for $v_{m+1} = v_m$ and $(v_{m+1}, \boldsymbol{d}_{m+1}) \in R_0^\infty$. As, by assumption on m, no node of priority less than $i = \Omega(v_m)$ occurs after v_m, Lemma 4.6 implies that for all $m \leq j \leq n$, $(v_j, \boldsymbol{d}_j) \in R_0^\infty$ with $\boldsymbol{d}_j := (d_0, \ldots, d_{i-1}, l_j, \ldots)$. Further, $l_j \geq l_{j'}$ for $j < j'$ and if $\Omega(v_j) = i$ then $l_j > l_{j'}$.

The play terminates at v_n, with configuration $\boldsymbol{d}_n = (d_0, \ldots, d_{i-1}, l_n, \infty, \ldots, \infty)$, so we get that $l_n = 0$ and $\Omega(v_n) = i$. Further, $(v_n, \boldsymbol{d}_n) \in R_0^\infty$. But this can only be the case if $v_n \in V_1$ as $(v_n, \boldsymbol{d}_n)$ with $v_n \in V_0$ would not satisfy ψ_0. Thus, Player 1 is to move at v_n and therefore loses the play. $\Box$

It is a simple observation that the formula φ_1 defining the winning positions for Player 1 analogous to φ_0 is equivalent to the dual formula of φ_0. Thus, all nodes x either satisfy φ_0 or φ_1 and therefore φ_0 defines the winning region for Player 0 and analogously φ_1 defines the winning region for Player 1. This establishes the definability theorem for backtracking games.

Theorem 4.8 *Winning regions of simple backtracking games are definable in* IFP.

Note that the definition of simple games involves semantic conditions, i.e. the players having linear strategies. It is open whether there is a purely syntactic criterion on game graphs allowing for the same kind of results.

Clearly, this result extends to polynomial backtracking games.

Corollary 4.9 *Winning regions of local polynomial backtracking games are definable in* IFP.

References

[1] J. van Benthem. One is a lonely number. Technical Report ILLC Research Report PP-2003-07, Institute for Logic, Language and Computation, University of Amsterdam, 2003.

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

[3] R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning About Knowledge.* MIT Press, 1995.

[4] M. R. Garey and D. S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness.* W. H. Freeman and company, New York, 1979. ISBN 0-7167-1044-7.

[5] E. Grädel. Finite model theory and descriptive complexity. In *Finite Model Theory and Its Applications.* Springer-Verlag, 2003. To appear. See http://www-mgi.informatik.rwth-aachen.de/Publications/pub/graedel/Gr-FMTbook.ps.

[6] E. Grädel and S. Kreutzer. Will deflation lead to depletion? On non-monotone fixed-point inductions. In *IEEE Symp. of Logic in Computer Science (LICS)*, 2003.

[7] E. Grädel, W. Thomas, and T. Wilke (eds). *Automata, Logics, and Infinite Games. A Guide to Current Research.* Lecture Notes in Computer Science Nr. 2500, Springer, 2002.

[8] S. Kreutzer. Expressive equivalence of least and inflationary fixed-point logic. In *17th Symp. on Logic in Computer Science (LICS)*, pages 403 – 413, 2002.

[9] D. Martin. Borel determinacy, *Annals of Mathematics* 102 (1975), pp. 336–371.

[10] J. Miller and L. Moss. The undecidability of iterated modal relativization. Manuscript, Indiana University, 2003.

[11] C. Stirling. Bisimulation, model checking and other games. Notes for the Mathfit instructional meeting on games and computation. Edinburgh, 1997.

[12] L.J. Stockmeyer and A.R. Meyer. Word problems requiring exponential time. In *Proc. of the 5th ACM Symp. on Theory of Computing*, pages 1 – 9, 1973.

[13] I. Walukiewicz. Monadic second order logic on tree-like structures. In *STACS'96*, volume 1046 of *Lecture Notes in Computer Science (LNCS)*, pages 401 – 414. Springer Verlag, 1996.