Quasipolynomial Computation of Nested Fixpoints

Daniel Hausmann and Lutz Schröder

Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

Abstract. It is well known that the winning region of a parity game with n nodes and k priorities can be computed as a k-nested fixpoint of a suitable function; straightforward computation of this nested fixpoint requires $\mathcal{O}(n^{\frac{k}{2}})$ iterations of the function. Calude et al.'s recent quasipolynomial-time parity game solving algorithm essentially shows how to compute the same fixpoint in only quasipolynomially many iterations by reducing parity games to quasipolynomially sized safety games. Universal graphs have been used to modularize this transformation of parity games to equivalent safety games that are obtained by combining the original game with a universal graph. We show that this approach naturally generalizes to the computation of solutions of systems of *any* set-valued fixpoint equations; hence, the solution of fixpoint equation systems can be computed by quasipolynomially many iterations of the equations. While this result is of clear interest in itself, we focus in particular on applications to modal fixpoint logics beyond relational semantics. For instance, the model checking problems for the graded and the (two-valued) probabilistic μ-calculus – with numbers coded in binary – can be solved via nested fixpoints of functions that differ substantially from the function for parity games but still can be computed in quasipolynomial time; our result hence implies that model checking for these μ-calculi is in QP. Moreover, we improve the exponent in known exponential bounds on satisfiability checking.

Keywords: Fixpoint theory, model checking, satisfiability checking, parity games, μ-calculus, coalgebraic logic

1 Introduction

Fixpoints are pervasive in computer science, governing large portions of recursion theory, concurrency theory, logic, and game theory. One famous example are parity games, which are central, e.g., to networks and infinite processes [3], tree automata [47], and μ-calculus model checking [18]. Winning regions in parity games can be expressed as nested fixpoints of particular set functions (e.g. [17,5]). In recent breakthrough work on the solution of parity games in quasipolynomial time, Calude et al. [6] essentially show how to compute this particular fixpoint in quasipolynomial time, that is, in time $2^{\mathcal{O}((\log n)^c)}$ for some constant c. Subsequently, it has been shown [29,14] that universal graphs can be used to transform parity games to equivalent safety games obtained by pairing the original game with a universal graph; the size of these safety games is determined by the size of the employed universal graphs and it has been shown [14] that there are universal graphs of quasipolynomial size. This yields a uniform algorithm for solving parity games to which all currently known quasipolynomial algorithms for parity games have been shown to instantiate using appropriately defined universal graphs [14].

Briefly, our contribution in the present work is to show that the method of using universal graphs to solve parity games generalizes to the computation of nested fixpoints of arbitrary set functions. That is, given set functions $f_i : \mathcal{P}(U)^{k+1} \to \mathcal{P}(U)$, $0 \leq i \leq k$ on a finite set U, we give an algorithm that uses universal graphs to compute the solutions of systems of equations

$$X_i =_{\eta_i} f_i(X_0, \ldots, X_k) \qquad\qquad 0 \leq i \leq k$$

where $\eta_i = \mathsf{GFP}$ (greatest fixpoint) or $\eta_i = \mathsf{LFP}$ (least fixpoint). Since there are universal graphs of quasipolynomial size, the algorithm requires only quasipolynomially many iterations of the functions f_i and hence runs in quasipolynomial time, provided that all f_i are computable in quasipolynomial time. While it seems plausible that this time bound may also be obtained by translating equation systems to equivalent standard parity games by emulating Turing machines to encode the functions f_i as Boolean circuits (leading to many additional states but avoiding exponential blowup during the process), we emphasive that the main point of our result is not so much the ensuing time bound but rather the insight that universal graphs and hence many algorithms for parity games can be used on a much more general level and the precise (and relatively low) quasipolynomial bound on the number of function calls that are required to obtain solutions of fixpoint equation systems.

In more detail, the method of Calude et al. can be described as annotating nodes of a parity game with histories of quasipolynomial size and then solving this annotated game, but with a safety winning condition instead of the much more involved parity winning condition. It has been shown that these histories can be seen as nodes in universal graphs, in a more general reduction of parity games to safety games in which nodes from the parity game are annotated with nodes from a universal graph. This method has also been described as pairing *separating automata* with safety games [14]. It has been shown [14] that there are exponentially sized universal graphs (yielding e.g. the fixpoint iteration algorithm [5] or the small progress measures algorithm [28]) and quasipolynomially sized universal graphs (corresponding, e.g., to the succinct progress measure algorithm [29], or to the recent quasipolynomial variant of Zielonka's algorithm [39]).

Hasuo et al. [24], and more generally, Baldan et al. [2] show that nested fixpoints in highly general settings can be computed by a technique based on progress measures, implicitly using exponentially sized universal trees, obtaining an exponential bound on the number of iterations. Our technique is based on showing that one can make explicit use of universal trees, correspondingly obtaining a quasipolynomial upper bound on the number of iterations. In both cases, computation of the nested fixpoint is reduced to a single (least or greatest depending on exact formulation) fixpoint of a function that extends the given set function to keep track of the exponential and quasipolynomial histories, respectively, in analogy to the previous reduction of parity games to safety games. Our central result can then be phrased as saying that the method of transforming parity conditions to safety conditions using universal graphs generalizes from solving parity games to solving systems of equations that use arbitrary set functions. We use *fixpoint games* [46,2] to obtain the crucial results that the solutions of equation systems have history-free witnesses, in analogy to history-freeness of winning strategies in parity games. These fixpoint games have exponential size but we show how to extract *polynomial-size* witnesses for winning strategies of Eloise, and use these witnesses to show that any node won by Eloise is also won in the safety game obtained by a universal graph. For the backwards direction, we show that a winning strategy in the safety game induces a winning strategy in the fixpoint game. This proves that universal graphs can be used to compute nested fixpoints of arbitrary set functions and hence yields the quasipolynomial upper bound for computation of nested fixpoints.

As an immediate application of these results, we improve generic upper complexity bounds on model checking and satisfiability checking in the *coalgebraic μ-calculus* [11], which serves as a generic framework for fixpoint logics beyond relational semantics. Well-known instances of the coalgebraic μ-calculus include the alternating-time μ-calculus [1], the graded μ-calculus [32], the (two-valued) probabilistic μ-calculus [11,35], and the monotone μ-calculus [19] (the ambient fixpoint logic of concurrent dynamic logic CPDL [41] and Parikh's game logic [38]). This level of generality is achieved by abstracting systems types as set functors and systems as coalgebras for the given functor following the paradigm of universal coalgebra [42].

It was previously shown [25] that the model checking problem for coalgebraic μ-calculi reduces to the computation of a nested fixpoint. This fixpoint may be seen as a coalgebraic generalization of a parity game winning region but can be literally phrased in terms of small standard parity games (implying quasipolynomial run time) only in restricted cases. Our results show that the relevant nested fixpoint can be computed in quasipolynomial time in all cases of interest. Notably, we thus obtain as new specific upper bounds that even under binary coding of numbers, the model checking problems of both the graded μ-calculus and the probabilistic μ-calculus are in QP, even when the syntax is extended to allow for (monotone) polynomial inequalities.

Similarly, the satisfiability problem of the coalgebraic μ-calculus has been reduced to a computation of a nested fixpoint [26], and our present results imply a marked improvement in the exponent of the associated exponential time bound. Specifically, the nesting depth of the relevant fixpoint is exponentially smaller than the number of states of the respective carrier set. Our results imply that this fixpoint is computable in polynomial time, which means that its computation is dominated by other steps of the procedure (especially determinization of Büchi automata). The complexity of satisfiability checking in coalgebraic μ-calculi (satisfying mild conditions on the modalities) thus drops from $2^{\mathcal{O}(n^2k^2\log n)}$ to $2^{\mathcal{O}(nk\log n)}$ for formulae of size n and with alternation depth k.

Related Work The quasipolynomial bound on parity game solving has in the meantime been realized by a number of alternative algorithms. For instance, Jurdzinski and Lazic [29] use succinct progress measures to improve to quasilinear (instead of quasipolynomial) space; Fearnley et al. [21] similarly achieve quasilinear space. Lehtinen [34] and Boker and Lehtinen [4] present a quasipolynomial algorithm using register games. Parys [39] improves Zielonka's algorithm [47] to run in quasipolynomial time. In particular the last algorithm is of interest as an additional candidate for generalization to nested fixpoints, due to the known good performance of Zielonka's algorithm in practice. Daviaud et al. [16] generalize quasipolynomial-time parity game solving by providing a pseudo-quasipolynomial algorithm for mean-payoff parity games. On the other hand, Czerwinski et al. [14] give a quasipolynomial lower bound on universal trees, implying a barrier for prospective polynomial-time parity game solving algorithms. Chatterjee et al. [7] describe a quasipolynomial time set-based symbolic algorithm for parity game solving that is parametric in a *lift* function that determines how ranks of nodes depend on the ranks of their successors, and thereby unifies the complexity and correctness analysis of various parity game algorithms. Although part of the parity game structure is encapsulated in a set operator *CPre*, the development is tied to standard parity games, e.g. in the definition of the *best* function, which picks minimal or maximal ranks of successors depending on whether a node belongs to Abelard or Eloise.

Early work on the computation of unrestricted nested fixpoints has shown that greatest fixpoints require less effort in the fixpoint iteration algorithm, which can hence be optimized to compute nested fixpoints with just $\mathcal{O}(n^{\frac{k}{2}})$ calls of the functions at hand [36,45], improving the previously known (straightforward) bound $\mathcal{O}(n^k)$; here, n denotes the size of the carrier set and k the number of fixpoint operators. Recent progress in the field has established the above-mentioned approaches using progress measures [24] and fixpoint games [2] in general settings, both with a view to applications in coalgebraic model checking like in the present paper. In comparison to the present work, the above approaches are set in more general frameworks, differing in particular in employing arbitrary complete lattices instead of finite powersets. On the other hand, their respective bounds on the required number of function iterations all are exponential.

2 Notation

Let U and V be sets, and let $R \subseteq U \times U$ be a binary relation on U. For $u \in U$, we then put $R(u) := \{v \in U \mid (u,v) \in R\}$. We put $[k] = \{0, \dots, k\}$ for $k \in \mathbb{N}$. *Labelled graphs* $G = (W, R)$ consist of a set W together with a relation $R \subseteq W \times A \times W$ where A is some set of labels; typically, we use $A = [k]$ for some $k \in \mathbb{N}$. An *R-path* in a labelled graph is a finite or infinite sequence $v_0, a_0, v_1, a_1, v_2 \dots$ (ending in a node from W if finite) such that $(v_i, a_i, v_{i+1}) \in R$ for all i. For $v \in W$ and $a \in A$, we put $R_a(v) = \{w \in W \mid (v, a, w) \in R\}$ and sometimes write $|G|$ to refer to $|W|$. As usual, we write U^* and U^ω for the sets of finite sequences or infinite sequences, respectively, of elements of U. The *domain* $\mathsf{dom}(f)$ of a partial function $f : U \rightharpoonup V$ is the set of elements on which f is defined. We often regard (finite) sequences $\tau = u_0, u_1, \ldots \in U^* \cup U^\omega$ of elements of U as partial functions of type $\mathbb{N} \rightharpoonup U$ and then write $\tau(i)$ to denote the element u_i, for $i \in \mathsf{dom}(\tau)$. For $\tau \in U^* \cup U^\omega$, we define the set

$$\mathsf{Inf}(\tau) = \{u \in U \mid \forall i \geq 0.\, \exists j > i.\, \tau(j) = u\}$$

of elements that occur infinitely often in τ (so $\mathsf{Inf}(\tau) = \emptyset$ for $\tau \in U^*$). An infinite R-path $v_0, p_0, v_1, p_1, \ldots$ in a labelled graph $G = (W, R)$ with labels from $[k]$ is *even* if $\max(\mathsf{Inf}(p_0, p_1, \ldots))$ is even, and G is *even* if every infinite R-path in G is even. We write $\mathcal{P}(U)$ for the powerset of U, and U^m for the m-fold Cartesian product $U \times \cdots \times U$. A function $g : \mathcal{P}(U)^k \to \mathcal{P}(U)$ is *monotone* if $g(V_1, \ldots, V_k) \subseteq g(W_1, \ldots, W_k)$ whenever $V_i \subseteq W_i$ for $1 \leq i \leq k$. For monotone $f : \mathcal{P}(U) \to \mathcal{P}(U)$, we put

$$\begin{aligned} \mathsf{GFP}\, f &= \bigcup\{V \subseteq U \mid V \subseteq f(V)\} \\ \mathsf{LFP}\, f &= \bigcap\{V \subseteq U \mid f(V) \subseteq V\}, \end{aligned}$$

which, by the Knaster-Tarski fixpoint theorem, are the greatest and the least fixpoint of f, respectively. Furthermore, we define $f^0(V) = V$ and $f^{m+1}(V) = f(f^m(V))$ for $m \geq 0$, $V \subseteq U$; if U is finite, then $\mathsf{GFP}\, f = f^n(U)$ and $\mathsf{LFP}\, f = f^n(\emptyset)$ by Kleene's fixpoint theorem. As usual, the *(forward) image* of $A' \subseteq A$ under a function $f : A \to B$ is $f[A'] = \{b \in B \mid \exists a \in A'.\, f(a) = b\}$ and the *preimage* $f^{-1}[B']$ of $B' \subseteq B$ under f is defined by $f^{-1}[B'] = \{a \in A \mid \exists b \in B'.\, f(a) = b\}$. *Projections* $\pi_j : A_1 \times \ldots \times A_m \to A_j$ for $1 \leq j \leq m$ are given by $\pi_i(a_1, \ldots, a_m) = a_j$.

3 Systems of Fixpoint Equations

We now introduce our central notion, that is, systems of fixpoint equations over a powerset lattice. We fix a set U and $k+1$ monotone functions $f_i : \mathcal{P}(U)^{k+1} \to \mathcal{P}(U)$, $0 \leq i \leq k$.

Definition 1. A *system of equations* consists of $k+1$ equations of the form

$$X_i =_{\eta_i} f_i(X_0, \ldots, X_k)$$

where $\eta_i \in \{\mathsf{LFP}, \mathsf{GFP}\}$, briefly referred to as f. For a partial valuation $\sigma : [k] \rightharpoonup \mathcal{P}(U)$, we inductively define

$$[\![X_i]\!]^\sigma = \eta_i X_i . f_i^\sigma,$$

where the function f_i^σ is given by

$$\begin{aligned} f_i^\sigma(A) = f_i([\![X_0]\!]^{\sigma'}, \ldots, [\![X_{i-1}]\!]^{\sigma'}, A, \\ \mathsf{ev}(\sigma', i+1), \ldots, \mathsf{ev}(\sigma', k)) \end{aligned}$$

for $A \subseteq U$, where $\sigma' = \sigma[i \mapsto A]$, $(\sigma[i \mapsto A])(j) = \sigma(j)$ for $j \neq i$ and $(\sigma[i \mapsto A])(i) = A$ and where $\mathsf{ev}(\sigma, j) = \sigma(j)$ if $j \in \mathsf{dom}(\sigma)$ and $\mathsf{ev}(\sigma, j) = [\![X_j]\!]^\sigma$

otherwise (the latter clause handles *free variables*). Then, the *solution* of the system of equations is $[\![X_k]\!]^\epsilon$ where $\epsilon : [k] \rightharpoonup \mathcal{P}(U)$ denotes the empty valuation (i.e. $\mathsf{dom}(\epsilon) = \emptyset$). Similarly, we can obtain solutions for the other components as $[\![X_i]\!]^\epsilon$ for $0 \leq i < k$; we drop the valuation index if no confusion arises, and sometimes write $[\![X_i]\!]_f$ to make the system f of equations explicit. We denote by E^{f_0} the solution $[\![X_k]\!]$ for the system of equations of the particular shape

$$\begin{aligned} X_i &=_{\eta_i} X_{i-1} && i > 0 \\ X_0 &=_{\mathsf{GFP}} f_0(X_0, \dots, X_k), \end{aligned}$$

where $\eta_i = \mathsf{LFP}$ for odd i and $\eta_i = \mathsf{GFP}$ for even i; the k-th component of the solution of the similar system obtained by putting $\eta_i = \mathsf{GFP}$ for odd i and $\eta_i = \mathsf{LFP}$ for even i is denoted by A^{f_0}.

Example 2. For an example of a system of fixpoint equations, let $A = (U, \Sigma, \delta, \Omega)$ be a parity automaton (e.g. [22]), consisting of a set U of states, an alphabet Σ, a transition relation $\delta \subseteq U \times \Sigma \times U$ and a priority function $\Omega : \delta \to [k]$ assigning priorities 0 to k to *edges* instead of nodes (this is standard since it allows for slightly smaller automata in some cases); for $0 \leq i \leq k$, we put $\Omega_i = \{(u, a, v) \in \delta \mid \Omega(u, a, v) = i\}$. Recall that by definition, A accepts an infinite word $w \in \Sigma^\omega$ if A has some run on w on which the highest priority that is visited infinitely often is even. For $0 \leq i \leq k$, we define $R_i : U \to \mathcal{P}(U)$ by putting

$$R_i(u) = \{v \in U \mid \exists a \in \Sigma. \, (u, a, v) \in \Omega_i\}$$

so that $R_i(u)$ is the set of states to which u has some transition with priority i. Then we define $f_{\mathsf{PA}} : \mathcal{P}(U)^{k+1} \to \mathcal{P}(U)$ by putting

$$f_{\mathsf{PA}}(U_0, \dots, U_k) = \{v \in U \mid \exists 0 \leq i \leq k. \, R_i(v) \cap U_i \neq \emptyset\}$$

for $(U_0, \dots, U_k) \in \mathcal{P}(U)^{k+1}$, that is, $f_{\mathsf{PA}}(U_0, \dots, U_k)$ consists of states v that have some transition with priority i that leads to some state from U_i. Then the set $\mathsf{E}^{f_{\mathsf{PA}}}$ is exactly the non-emptiness region of A (that is, the set of states in A that accept some infinite word).

4 Parity Games

We recall some basic notions on parity games (see, e.g., [22]).

Definition 3 (Parity games). A *parity game* (V, E, Ω) consists of a set of *nodes* V, a set $E \subseteq V \times V$ of *moves* encoding the rules of the game, and a *priority function* $\Omega : V \to \mathbb{N}$, which assigns *priorities* $\Omega(v) \in \mathbb{N}$ to nodes $v \in V$. Moreover, each node belongs to exactly one of the two players Eloise or $\mathsf{Abelard}$, where we denote the set of Eloise's nodes by $V_\exists$ and that of $\mathsf{Abelard}$'s nodes by $V_\forall$. A *play* $\rho \in V^* \cup V^\omega$ is a (finite or infinite) sequence of nodes that follows the rules of the game, that is, such that for all $i \geq 0$ such that ρ contains at least $i+1$

nodes, we have $(\rho(i),\rho(i+1)) \in E$. We say that an infinite play $\rho = v_0, v_1, \dots$ is *even* if the largest priority that occurs infinitely often in it (i.e. $\max(\mathsf{Inf}(\Omega \circ \rho))$) is even, and *odd* otherwise; finite plays are required to end in nodes that have no outgoing move. Player Eloise *wins* all even plays and finite plays that end in an Abelard-node; player Abelard *wins* all other plays (note that this implies that Eloise wins if an Abelard-node is reached from which there are no moves). The *size* of a parity game (V, E, Ω) is $|V|$. A *(history-free)* Eloise-*strategy* $s : V_\exists \rightharpoonup V$ is a partial function that assigns single moves $s(x)$ to Eloise-nodes $x \in \mathsf{dom}(s)$. A play ρ *follows* an Eloise-strategy s if for all $i \in \mathsf{dom}(\rho)$ such that $\rho(i) \in V_\exists$, we have $\rho(i+1) = s(\rho(i))$; then we also say that ρ is an *s-play*. An Eloise-strategy *wins* a node $v \in V$ if Eloise wins all s-plays that start at v. We have a dual notion of Abelard-strategies; *solving* a parity game consists in computing the *winning regions* $\mathsf{win}_\exists$ and $\mathsf{win}_\forall$ of the two players, that is, the sets of states that they respectively win by some strategy. A parity game is *alternating* if $E[V_E] \subseteq V_\forall$ and $E[V_\forall] \subseteq V_\exists$, that is, if all of Eloise's moves lead to Abelard-nodes and vice versa.

It is known that solving parity games is in NP $\cap$ coNP (and, more specifically, in UP $\cap$ co-UP). Recently it has also been shown [6] that for parity games with n nodes and k priorities, $\mathsf{win}_\exists$ and $\mathsf{win}_\forall$ can be computed in quasipolynomial time $\mathcal{O}(n^{\log k+6})$. Another crucial property of parity games is that they are *history-free determined* [22], that is, that every node in a parity game is won by exactly one of the two players and then there is a history-free strategy for the respective player that wins the node. This is reflected by the central fact that the winning regions in parity games can be computed by *fixpoint iteration*, that is, each of these regions is the solution of a system of fixpoint equations as introduced in Section 3:

Lemma 4 ([17,5]). *Let (V, E, Ω) be a parity game with priorities 0 to k. Then we have*

$$\mathsf{win}_\exists = \mathsf{E}^{f_\exists} \quad \textit{and} \quad \mathsf{win}_\forall = \mathsf{A}^{f_\forall},$$

where $f_\exists : \mathcal{P}(V)^{k+1} \to \mathcal{P}(V)$ and $f_\forall : \mathcal{P}(V)^{k+1} \to \mathcal{P}(V)$ are defined, for $(V_0, \dots, V_k) \in \mathcal{P}(V)^{k+1}$, by

$$\begin{aligned} f_\exists(V_0,\dots,V_k) = &\{v \in V_\exists \mid \exists 0 \le i \le k.\, \Omega(v) = i, \\ &\qquad\qquad E(v) \cap V_i \neq \emptyset\} \cup \\ &\{v \in V_\forall \mid \exists 0 \le i \le k.\, \Omega(v) = i, \\ &\qquad\qquad E(v) \subseteq V_i, \} \end{aligned}$$

and

$$\begin{aligned} f_\forall(V_0,\dots,V_k) = &\{v \in V_\exists \mid \exists 0 \le i \le k.\, \Omega(v) = i, \\ &\qquad\qquad E(v) \subseteq V_i\} \cup \\ &\{v \in V_\forall \mid \exists 0 \le i \le k.\, \Omega(v) = i, \\ &\qquad\qquad E(v) \cap V_i \neq \emptyset\}. \end{aligned}$$

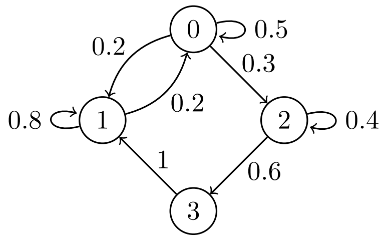
Intuitively, $f_\exists(V_0, \dots, V_k)$ is the set of nodes v for which there is some $0 \leq i \leq k$ such that v has priority i and from which Eloise can enforce that some node from V_i is reached in one step of the game. The nested fixpoint $\mathsf{E}^{f_\exists}$ (in which least (greatest) fixpoints correspond to odd (even) priorities) is constructed in such a way that Eloise only has to rely infinitely often on an argument V_i for odd i if she can also ensure that some argument V_j for $j > i$ is used infinitely often.

Example 5. We now consider probabilistic parity games, which make use of systems of fixpoint equations that deviate considerably from (and apparently do not reduce easily to) the ones for standard parity games. Probabilistic parity games are parity games in which both moves and nodes are annotated with probabilities (these games are not to be confused with the $2\frac{1}{2}$-player *stochastic parity games* that are considered in [8,23]). They arise naturally as model checking games for the (two-valued) probabilistic μ-calculus (see Example 15.2); we postpone a more formal and detailled treatment to Section 7 below, where we discuss the more general coalgebraic μ-calculus (covering e.g. probabilistic, graded and the alternating-time μ-calculi as instances) and its model checking problem (corresponding to solving e.g. probabilistic, graded and alternating-time parity games).

A *probabilistic parity game* (V, D, Ω, σ) consists of a set V of nodes, a set of probabilistic moves, given by a function D which assigns probability distributions $D(v)$ over V (with $\Sigma_{w \in V}(D(v))(w) = 1$) to nodes $v \in V$, a priority function $\Omega : V \to \mathbb{N}$ and a *probability assignment* $\sigma : V \to [0, 1]$. The intuition of $(D(v))(w) = p$ is that the move from v node to node w has probability p. A *play* $\rho = v_0, v_1, \dots$ in a probabilistic parity game is a sequence of nodes such that for all $i \geq 0$, we have $(D(v_i))(v_{i+1}) > 0$ and the winning conditions on plays are the same as in standard parity games. A *(history-free) strategy* is a partial function $s : V \rightharpoonup \mathcal{P}(V)$ such that for all $v \in \mathsf{dom}(s)$, $\Sigma_{w \in s(v)}(D(v))(w) > \sigma(v)$, that is, strategies pick *sets* of moves whose joint probability is larger than the probability assignment of the respective node. Crucially, strategies in probabilistic parity games involve branching, unlike strategies in standard parity games which pick single moves for each node on which they are defined. An s-play is a play $\rho = v_0, v_1, \dots$ such that for all $i \geq 0$, $v_i \in \mathsf{dom}(s)$ and $v_{i+1} \in s(v_i)$. Player Eloise wins a node v if there is a strategy s such that Eloise wins all s-plays that start at v. We point out that the (somewhat concealed) two-player nature of probabilistic parity games manifests in the fact that Eloise has to pick, in each turn, *some* suitable set of moves, whereupon Abelard can challenge *any* of these moves (that is, *all* s-plays need to be even in order for s to be a winning strategy for Eloise). Player Eloise hence wins a node v if and only if there is a set $W \subseteq V$ containing v and a graph $G = (W, R \subseteq W \times W)$ such that

– each R-edge has D-probability greater than 0,
– for each $w \in W$, the R-successors of w have a joint D-probability of more than $\sigma(w)$ and
– for each infinite R-path that starts at v, the highest priority that is visited infinitely often by the path is even.

Consider, for instance, the probabilistic parity game depicted below with $V = \{0, 1, 2, 3\}$, $\Omega(i) = i$ for $i \in V$ and, e.g. $(D(0))(1) = 0.2$, $(D(3))(1) = 1$ and $(D(3))(3) = 0$; let us fix the probability assigment σ by putting $\sigma(0) = 0.7$, $\sigma(1) = 0.3$ $\sigma(2) = 0.1$ and $\sigma(3) = 0$.



To win e.g. the node 0, Eloise has to have a strategy that selects a set of nodes that have a joint probability (of being reached from 0 in one step) greater than $\sigma(0) = 0.7$ and that are in turn all won by the strategy. In this example, player Eloise wins the nodes 0 and 2 with the strategy s defined by $s(0) = \{0, 2\}$ and $s(2) = \{2\}$: this function indeed is a valid strategy since it uses only moves with nonzero probabilities and also respects the probability assignment σ as we have $\Sigma_{j\in s(0)}(D(0))(j) = 0.5 + 0.3 > \sigma(0) = 0.7$ and $\Sigma_{j\in s(2)}(D(2))(j) = 0.4 > \sigma(2) = 0.1$. Also, every s-play that starts at node 0 is of the form 0^ω or 0^*2^ω and hence even and every s-play that starts at node 2 is of the form 2^ω and hence even. On the other hand, there is no strategy with which Eloise can win the nodes 1 or 3 since for any strategy t with $1 \in \mathsf{dom}(t)$, we have $\Sigma_{w\in t(1)}(D(1))(w) > \sigma(1) = 0.3$ and hence $1 \in t(1)$; but then there is an odd t-play of the form 1^ω so that t is not a winning strategy for Eloise. Also, for any candidate winning strategy u with $3 \in \mathsf{dom}(u)$, we have $u(3) = \{1\}$ and hence $1 \in \mathsf{dom}(u)$ which shows that u is not a winning strategy for Eloise. Hence we have $\mathsf{win}_\exists = \{0, 2\}$ and $\mathsf{win}_\forall = \{1, 3\}$.

The winning regions in probabilistic parity games are again just nested fixpoints, where the functions however deviate significantly from the functions for standard parity games. Player Eloise has to pick sets of moves now, so it does not suffice to consider existential or universal branching, like in the functions $f_\exists$ and $f_\forall$ from Lemma 4. We define $f_{\exists p} : \mathcal{P}(V)^{k+1} \to \mathcal{P}(V)$, for $(V_0, \dots, V_k) \in \mathcal{P}(V)^{k+1}$, by putting

$$f_{\exists p}(V_0, \dots, V_k) = \{v \in V \mid \exists 0 \le i \le k.\, \Omega(v) = i,\\ \Sigma_{w\in V_i}(D(v))(w) > \sigma(v)\}$$

Then we have $\mathsf{win}_\exists = \mathsf{E}^{f_{\exists p}}$ (formally, this is a consequence of Lemma 17, below). The winning region of Abelard is characterized in a dual manner. Note that nodes v with $\sigma(v) = p$ correspond to model checking modal operators $\langle p\rangle$ which require that their argument is satisfied with probability more than p in the next step (see Example 15.2). The full probabilistic μ-calculus also has dual operators $[p]$ which state that their argument holds with probability at least $1 - p$ in the next step; for brevity, we refrain from modelling these operators in this example.

While the above example apparently already goes beyond the setting of [7] in which standard parity games with existential and universal branching are hardwired throughout (e.g. in the set operator *CPre* and the function *best*), we note that our results cover systems of fixpoint equations for arbitrary set-functions, which need not be 'game-like' at all, that is, they need not be parametrized by any graph structure or priority and player assignment.

5 Fixpoint Games and History-free Witnesses

We instantiate the existing notion of fixpoint games [46,2], which characterize containment in nested fixpoints, to our setting (that is, to powerset lattices), and then use these games to establish our crucial notion of history-freeness for systems of fixpoint equations.

Definition 6 (Fixpoint games). Let $X_i =_{\eta_i} f_i(X_0, \dots, X_k)$, $0 \leq i \leq k$, be a system of fixpoint equations. The associated *fixpoint game* is a parity game (V, E, Ω) with set of nodes $V = (U \times [k]) \cup \mathcal{P}(U)^{k+1}$, where nodes from $U \times [k]$ belong to player Eloise and nodes from $\mathcal{P}(U)^{k+1}$ belong to player $\mathsf{Abelard}$. For nodes $(u, i) \in U \times [k]$, we put

$$E(u,i) = \{(U_0, \dots, U_k) \in \mathcal{P}(U)^{k+1} \mid u \in f_i(U_0, \dots, U_k)\},$$

and for nodes $(U_0, \dots, U_k) \in \mathcal{P}(U)^{k+1}$, we put

$$E(U_0, \dots, U_k) = \{(u, i) \mid 0 \leq i \leq k, u \in U_i\}.$$

The priority function $\Omega : V \to [2k+1]$ is defined by

$$\Omega(u,i) = \begin{cases} 2i & \text{if } \eta_i = \mathsf{GFP} \\ 2i+1 & \text{if } \eta_i = \mathsf{LFP} \end{cases}$$

and $\Omega(U_0, \dots, U_k) = 0$.

We import the associated characterization theorem [2, Theorem 4.8]:

Lemma 7 ([2]). *We have $u \in [\![X_i]\!]_f$ if and only if Eloise wins the node (u,i) in the fixpoint game for the given system f of equations.*

We now define our notion of history-freeness for systems of fixpoint equations.

Definition 8 (History-free witness). A *history-free witness* for containment of $u \in U$ in $[\![X_i]\!]_f$ is an even labelled graph (W, R) with labels from $[k]$ such that $W \subseteq U \times [k]$, $(u, i) \in W$, and for all $(v, p) \in W$, we have $v \in f_p(\pi_1[R_0(v,p)], \dots, \pi_1[R_k(v,p)])$, noting that for $0 \leq j \leq k$, $R_j(v,p) \subseteq W$ so that $\pi_1[R_j(v,p)] \subseteq U$.

History-free witnesses thus assign tuples $(R_1(v,p), \dots, R_k(v,p))$ of sets $R_j(v,p) \subseteq W$ to pairs $(v,p) \in W$ without relying on a history of previously visited nodes. We have $|W| \leq (k+1)|U|$ and $|R| \leq (k+1)|W|^2$, that is, the size of history-free witnesses is polynomial in U. Crucially, history-free witnesses always exist:

Lemma 9. *We have $u \in [\![X_i]\!]_f$ if and only if there is a history-free witness for containment of u in $[\![X_i]\!]_f$.*

Proof. In one direction, we have $u \in [\![X_i]\!]_f$ so that Eloise wins the node (u, i) in the according fixpoint game by Lemma 7. Let s be a corresponding *history-free* winning strategy. We inductively construct a witness for containment of u in $[\![X_i]\!]_f$, starting at (u, i). When at $(v, p) \in U \times [k]$ with $s(v, p) = (U_0, \dots, U_k)$, we put $R_i(v, p) = U_i \times \{i\}$ for $0 \le i \le k$ and hence have $i = i'$ for all $((v, p), i, (u, i')) \in R$. Since s is a winning strategy, the resulting graph (W, R) is a history-free witness for containment of u in $[\![X_i]\!]_f$ by construction; in particular, (W, R) is even. For the converse direction, the witness for containment of u in $[\![X_i]\!]_f$ directly yields a winning Eloise-strategy for the node (u, i) in the associated fixpoint game. This implies $u \in [\![X_i]\!]_f$ by Lemma 7.

6 Solving Equation Systems using Universal Graphs

We go on to prove our main result. To this end, we fix a finite set U and a system f of fixpoint equations $f_i : \mathcal{P}(U)^{k+1} \to \mathcal{P}(U)$, $0 \le i \le k$, and put $n := |U|$.

Definition 10 (Universal graphs [14,13]). Let $G = (W, R)$ and $G' = (W', R')$ be labelled graphs with labels from $[k]$. A *homomorphism of labelled graphs* from G to G' is a function $\Phi : W \to W'$ such that for all $(v, p, w) \in R$, we have $(\Phi(v), p, \Phi(w)) \in R'$. An $(n, k+1)$-*universal graph* S is an even graph with labels from $[k]$ such that for all even graphs G with labels from $[k]$ and with $|G| \le n$, there is a homomorphism from G to S.

We fix an

$$(n(k+1), (k+1))\text{-universal graph } S = (Z, L),$$

noting that there are $(n(k+1), (k+1))$-universal graphs (obtained from universal trees) of size quasipolynomial in n and k [14]. We now combine the system f with the universal graph S to turn the parity conditions associated to general systems of fixpoint equations into a safety condition, associated to a single greatest fixpoint.

Definition 11 (Chained-product fixpoint). We define a function $g : \mathcal{P}(U \times [k] \times Z) \to \mathcal{P}(U \times [k] \times Z)$ by

$$g(X) = \{(v, p, q) \in U \times [k] \times Z \mid v \in f_p(X_0^q, \dots, X_k^q)\}$$

where

$$X_i^q = \{u \in U \mid \exists s \in L_i(q).\, (u, i, s) \in X\}.$$

We refer to $Y_0 =_{\mathsf{GFP}} g(Y_0)$ as the *chained-product fixpoint (equation)* of f and S.

We now show our central result: apart from the annotation with states from the universal graph, the chained-product fixpoint g is the solution of the system f.

Theorem 12. *For $0 \le i \le k$, we have $u \in [\![X_i]\!]_f$ if and only if $(u,i) \in \tau[[\![Y_0]\!]_g]$, where $\tau(v,p,q) = (v,p)$.*

Proof. For the forward direction, let $u \in [\![X_i]\!]_f$. By Lemma 9, there is a history-free witness $G = (W,R)$ for containment of u in $[\![X_i]\!]_f$. Since S is a $(n(k+1), k+1)$-universal graph and since G is a witness and hence an even labelled graph of suitable size $|G| \le n(k+1)$, there is a graph homomorphism Φ from G to S. Pick some $q \in Z$. Starting at $(u,i,q,0)$, we inductively construct a witness for containment of (u,i,q) in $[\![Y_0]\!]_g$. When at $(v_1,p_1,q_1,0)$ with $(v_1,p_1) \in W$, we put

$$\begin{aligned} R'_0(v_1,p_1,q_1,0) = \{(v_2,p_2,q_2,0) \in U \times [k] \times Z \times [0] \mid \\ (v_2,p_2) \in R_{p_2}(v_1,p_1), \\ (\Phi(v_1,p_1),p_2,q_2) \in L\} \end{aligned}$$

and continue the inductive construction with all these $(v_2,p_2,q_2,0)$, knowing $(v_2,p_2) \in W$. The resulting structure $G' = (W',R')$ indeed is a witness for containment of (u,i,q) in $[\![Y_0]\!]_g$: G' is even by construction. Moreover, we need to show for $(v_1,p_1,q_1,0) \in W'$ that $(v_1,p_1,q_1,0) \in g(\pi_1[R'_0(v_1,p_1,q_1,0)])$, i.e. $v_1 \in f_{p_1}(\pi_1[R'_0(v_1,p_1,q_1)]^{q_1}_0, \dots, \pi_1[R'_0(v_1,p_1,q_1)]^{q_1}_k)$. Since G is a witness and $(v_1,p_1) \in W$ by construction of W', we have $v_1 \in f_{p_1}(\pi_1[R_0(v_1,p_1)], \dots, \pi_1[R_k(v_1,p_1)])$. By monotonicity of f_{p_1}, it thus suffices to show that $\pi_1[R_j(v_1,p_1)] \subseteq R'(v_1,p_1,q_1)^{q_1}_j$ for $0 \le j \le k$. So let $w \in \pi_1[R_j(v_1,p_1)]$. Since R is a witness that is constructed as in the proof of Lemma 9, we have $i = i'$ for all $((v',p'),i,(w',i')) \in R$. Thus $(w,j) \in R_j(v_1,p_1)$, that is, $((v_1,p_1),j,(w,j)) \in R$, hence $(\Phi(v_1,p_1),j,\Phi(w,j)) \in L$ because Φ is a graph homomorphism. It follows that $(w,j,\Phi(w,j),0) \in R'_0(v_1,p_1,q_1,0)$ and hence $w \in \pi_1[R'_0(v_1,p_1,q_1,0)]^{q_1}_j$, as required.

For the converse implication, let $(u_0,p_0,q_0) \in [\![Y_0]\!]_g$ for some $q_0 \in Z$. Let $G = (W,R)$ be a history-free witness for this fact. By Lemma 7, it suffices to provide a strategy for the fixpoint game for the system f with which Eloise wins the node (u_0,p_0). We inductively construct a *history-dependent* strategy s as follows: We put $s((u_0,p_0)) = (R_0(u_0,p_0,q_0,0)^{q_0}_0, \dots, R_0(u_0,p_0,q_0,0)^{q_0}_k)$. For the inductive step, let

$$\begin{aligned} \tau =& (u_0,p_0), (U^0_0, \dots, U^0_k), (u_1,p_1), \dots, (u_{n-1},p_{n-1}), \\ & (U^{n-1}_0, \dots, U^{n-1}_k), (u_n,p_n) \end{aligned}$$

be a partial play of the fixpoint game that follows the strategy that has been constructed so far. Then we have an R-path $(u_0,p_0,q_0,0), (u_1,p_1,q_1,0), \dots, (u_n,p_n,q_n,0)$, where, for $0 \le i < n$, we have $(q_i,p_{i+1},q_{i+1}) \in L$ since $u_{i+1} \in R_0(u_i,p_i,q_i,0)^{q_i}_{p_{i+1}}$ by the inductive construction. Put $s(\tau) = (R_0(u_n,p_n,q_n,0)^{q_n}_0, \dots, R_0(u_n,p_n,q_n,0)^{q_n}_k)$. Since G is a witness, the strategy uses only moves that are available to Eloise (i.e. ones with $u_n \in f_{p_n}(s(\tau))$). Also, s is a winning strategy as can be seen by looking at the L-paths that are induced by complete plays τ that follow s, as described (for partial plays) above. Since U is a universal graph and hence even, every such L-path is even and the sequence of priorities in τ is just the sequence of priorities of one of these L-paths.

Corollary 13. *Solutions of systems of fixpoint equations can be computed with*

$$2n^2(k+1)^2\binom{\log(n(k+1))+k+2}{k+1}$$

(that is, quasipolynomially many) computations of the functions f_i.

Proof. It has been shown in [14], Theorem 2.2. that there are (l,h)-universal trees of size $2l\binom{\log l+h+1}{h}$; [13] shows how to obtain a suitable universal graph of the same size. Using $l = n(k+1)$ and $h = k+1$ and a universal graph obtained in this way, the solution of g stabilizes after $n(k+1)\cdot q$ many iterations of g where $q = 2n(k+1)\binom{\log n(k+1)+k+2}{k+1}$.

We furthermore have that if $k+1 < \log n(k+1)$, then the number of function calls required for the solution of equations systems is bounded by a polynomial:

Theorem 14. *Let $k+1 < \log n(k+1)$. Then $[\![Y_0]\!]_g$ can be computed with $(n(k+1))^8$ computations of functions f_i.*

Proof. We note that if $k+1 < \log n(k+1)$, then the leaves of tree-like $(n(k+1), k+1)$-universal graphs of quasipolynomial size can, as in Theorem 2.8 in [6], be represented by a set of size $\mathcal{O}((n(k+1))^3)$ in such a way that leaves can be correctly recreated from their representation. Hence the solution $[\![Y_0]\!]_g$ can be computed with $\mathcal{O}((n(k+1))^4)$ iterations of g and a single computation of g requires at most $\mathcal{O}((n(k+1))^4)$ calls of some function f_i.

While the obtained bounds on function calls are parametric in the number k of fixpoint equations, we conjecture that our constructions can be improved so that k can instead be taken to be the number of alternating nestings of least and greatest fixpoint equations.

7 Applications to Coalgebraic μ-Calculi

We next show how to apply our results to model checking and satisfiability checking for generalized μ-calculi in the setting of coalgebraic logic, covering, for instance, graded [32], probabilistic [12,35], and alternating-time [1] μ-calculi. It has been shown in previous work [25] that model checking for coalgebraic μ-calculi reduces to computing winning regions in a generalized variant of parity games where the game arenas are coalgebras instead of Kripke frames. We proceed to recall basic definitions and examples in universal coalgebra [42] and the coalgebraic μ-calculus [10] and then continue to show that our main result yields new quasipolynomial-time upper bounds for the model checking problem and improves the known exponential-time upper bound for the satisfiability problem [26] of the coalgebraic μ-calculus. These generic results instantiate to new upper bounds in all concrete cases except the standard relational μ-calculus.

The abstraction principle underlying universal coalgebra is to encapsulate system types as functors, for our present purposes on the category of sets. Such a functor $T : \mathsf{Set} \to \mathsf{Set}$, which we fix in the following, maps every set X to a set TX, and every map $f : X \to Y$ to a map $Tf : TX \to TY$, preserving identities and composition. We think of TX as a type of structured collections over X; a basic example is the covariant powerset functor $\mathcal{P}$, which assigns to each set its powerset and acts on maps by taking forward image. Systems of the intended type are then cast as *T-coalgebras* (C, ξ) (or just ξ) consisting of a set C of *states* and a *transition map* $\xi : C \to TC$, thought of as assigning to each state $x \in C$ a structured collection $\xi(x) \in TC$ of successors. E.g. a $\mathcal{P}$-coalgebra $\xi : C \to \mathcal{P}C$ assigns to each state a set of successors, i.e. is a transition system.

Following the paradigm of *coalgebraic logic* [12], we fix a set Λ of modal operators; we interpret each $\heartsuit \in \Lambda$ as *predicate lifting* $[\![\heartsuit]\!]$ for T, i.e. a natural transformation

$$[\![\heartsuit]\!]_X : 2^X \to 2^{TX}.$$

Here, the index X ranges over all sets; 2^X denotes the set of maps $X \to 2$ into the two-element set $2 = \{\bot, \top\}$, isomorphic to the powerset of X (i.e. 2^- is the *contravariant powerset functor*; we generally keep the conversion between 2^X and $\mathcal{P}(X)$ implicit); and naturality means that $[\![\heartsuit]\!]_X(f^{-1}[A]) = (Tf)^{-1}[[\![\heartsuit]\!]_Y(A)]$ for $f : X \to Y$ and $A \in 2^Y$. Thus, the predicate lifing $[\![\heartsuit]\!]$ indeed lifts predicates on a base set X to predicates on the set TX. Standard examples for $T = \mathcal{P}$ are the predicate liftings for the $\Box$ and $\Diamond$ modalities, given by

$$\begin{aligned} [\![\Box]\!]_X(A) &= \{B \in \mathcal{P}X \mid B \subseteq A\} \qquad \text{and} \\ [\![\Diamond]\!]_X(A) &= \{B \in \mathcal{P}X \mid B \cap A \neq \emptyset\} \end{aligned}$$

for $A \in PX$. Since we mean to form fixpoint logics, we need to require that every $[\![\heartsuit]\!]$ is *monotone*, that is, $A \subseteq B \subseteq X$ implies $[\![\heartsuit]\!]_X(A) \subseteq [\![\heartsuit]\!]_X(B)$. To support negation, we assume moreover that Λ is closed under *duals*, i.e. for each $\heartsuit \in \Lambda$ we have $\overline{\heartsuit} \in \Lambda$ such that $[\![\overline{\heartsuit}]\!]_X(A) = TX \setminus [\![\heartsuit]\!]_X(X \setminus A)$, chosen so that $\overline{\overline{\heartsuit}} = \heartsuit$ (e.g. $\overline{\Box} = \Diamond$, $\overline{\Diamond} = \Box$).

Given a set Var of *fixpoint variables*, the set of *formulae* $\phi, \psi, \dots$ of the coalgebraic μ-calculus is then defined by the grammar

$$\psi, \phi := \top \mid \bot \mid \psi \vee \phi \mid \psi \wedge \phi \mid \heartsuit\psi \mid X \mid \eta X.\psi \qquad (\heartsuit \in \Lambda, X \in \mathsf{Var}, \eta \in \{\mu, \nu\}).$$

Given a T-coalgebra $\xi : C \to TC$ and a valuation $\sigma : \mathsf{Var} \to \mathcal{P}C$, the *extension*

$$[\![\phi]\!]_\sigma \subseteq C$$

of a formula ϕ is defined recursively by $[\![X]\!]_\sigma = \sigma(X)$; the expected clauses for the propositional operators ($[\![\top]\!]_\sigma = C$; $[\![\bot]\!]_\sigma = \emptyset$; $[\![\phi \wedge \psi]\!]_\sigma = [\![\phi]\!]_\sigma \cap [\![\psi]\!]_\sigma$;

$[\![\phi \vee \psi]\!]_\sigma = [\![\phi]\!]_\sigma \cup [\![\psi]\!]_\sigma$); and

$$\begin{aligned}[\![\heartsuit\psi]\!]_\sigma &= \xi^{-1}[[\![\heartsuit]\!]([\![\psi]\!]_\sigma)]\\ [\![\mu X.\psi]\!]_\sigma &= \mathsf{LFP}[\![\psi]\!]^X_\sigma\\ [\![\nu X.\psi]\!]_\sigma &= \mathsf{GFP}[\![\psi]\!]^X_\sigma\end{aligned}$$

where the (monotone) map $[\![\psi]\!]^X_\sigma : \mathcal{P}C \to \mathcal{P}C$ is defined by $[\![\psi]\!]^X_\sigma(A) = [\![\psi]\!]_{\sigma[X\mapsto A]}$ for $A \subseteq C$, with $(\sigma[X \mapsto A])(X) = A$ and $(\sigma[X \mapsto A])(Y) = \sigma(Y)$ for $X \neq Y$.

The *alternation depth* $\mathsf{ad}(\eta X.\psi)$ of a fixpoint $\eta X.\psi$ is the depth of alternating nesting of such fixpoints in ψ that depend on X; we assign *odd* numbers to least fixpoints and *even* numbers to greatest fixpoints. E.g. for $\psi = \nu X.\phi$ and $\phi = \mu Y.(p \wedge \heartsuit X) \vee \heartsuit Y$, we have $\mathsf{ad}(\psi) = 2$, $\mathsf{ad}(\phi) = 1$. For a detailed definition of alternation depth, see e.g. [37].

Example 15. As indicated above, the standard relational μ-calculus [30] is one example of a coalgebraic μ-calculus, with propositional atoms treated as nullary modalities. Further important examples are as follows [43,10,44].

1. The *graded μ-calculus* [32] has modalities $\langle b\rangle$, $[b]$, indexed over $b \in \mathbb{N}$, read 'in more than b successors' and 'in all but at most b successors', respectively. These can be interpreted over relational structures but it is more natural and technically more convenient to use *multigraphs* [15], i.e. transition systems with edge weights (*multiplicities*) in $\mathbb{N} \cup \{\infty\}$, which are coalgebras for the multiset functor $\mathcal{B}$ given by $\mathcal{B}X = (X \to (\mathbb{N} \cup \{\infty\}))$. Over $\mathcal{B}$, we interpret $\langle b\rangle$ and $[b]$ by the mutually dual predicate liftings

$$\begin{aligned}[\![\langle b\rangle]\!]_X(A) &= \{\beta \in \mathcal{B}X \mid \textstyle\sum_{x\in A} \beta(x) > b\}\\ [\![[b]]\!]_X(A) &= \{\beta \in \mathcal{B}X \mid \textstyle\sum_{x\in X\setminus A} \beta(x) \leq b\}.\end{aligned}$$

E.g. the formula $\nu X.\,(\phi \wedge \Diamond_1 X)$ says that the current state is the root of an infinite tree with branching degree at least 2 (counting multiplicities) on which ϕ holds everywhere.

2. The (two-valued) *probabilistic μ-calculus* [10,35] is interpreted over Markov chains, which are coalgebras for the *discrete distribution functor $\mathcal{D}$* where $\mathcal{D}X = \{\beta : X \to [0,1] \mid \sum_{x\in X} \beta(x) = 1\}$ is the set of discrete probability distributions on X, represented, e.g., as probability mass functions $\beta : X \to [0,1]$. We abuse β to denote also the induced probability distribution, writing $\beta(A) = \sum_{x\in A} \beta(x)$ for $A \subseteq X$. The logic has modalities $[p]$, $\langle p\rangle$ indexed over $p \in [0,1] \cap \mathbb{Q}$, interpreted over $\mathcal{D}$ by

$$\begin{aligned}[\![\langle p\rangle]\!]_X(A) &= \{\beta \in \mathcal{D}X \mid \beta(A) > p\}\\ [\![\langle p\rangle]\!]_X(A) &= \{\beta \in \mathcal{D}X \mid \beta(X \setminus A) \leq p\}.\end{aligned}$$

This example (as well as the previous one) can be extended to admit (monotone) polynomial inequalities among probabilities (or multiplicities, respectively) instead of only comparison with constants, allowing, e.g., for expressing probabilistic independence [20,33,26]. In more detail, we can introduce n-ary modalities

$L_{p,b}$, $M_{p,b}$ indexed over polynomials $p \in \mathbb{Q}_{\geq 0}[x_1, \dots, x_n]$ and rational numbers $b \geq 0$, with $L_{p,b}$ interpreted by the predicate lifting

$$\begin{aligned} [\![L_{p,b}]\!]_X(A_1, \dots, A_n) = \{\beta \in \mathcal{D}X \mid \\ p(\beta(A_1), \dots, \beta(A_n)) > b\} \end{aligned}$$

and $M_{p,b}$ by the corresponding dual predicate lifting. E.g. the formula

$$\nu X.\, \mu Y.\, L_{x_1 x_2, 0.8}(p \wedge X, q \vee Y)$$

says roughly that if we independently sample two successors of the current state, then with probability at least 0.8, the first successor state will satisfy p, and then X again (continuing indefinitely), and the second successor state will remain on a path where it satisfies Y again until it eventually reaches q.

3. *Monotone μ-calculus:* The *monotone neighbourhood functor* $\mathcal{M}$ maps a set X to the set

$$\mathcal{M}X = \{\mathfrak{A} \in 2^{(2^X)} \mid \mathfrak{A} \text{ upwards closed}\}$$

of set systems over X that are upwards closed under subset inclusion (i.e. $A \in \mathfrak{A}$ and $A \subseteq B$ imply $B \in \mathfrak{A}$). Coalgebras for $\mathcal{M}$ are *monotone neighbourhood frames* in the sense of Scott-Montague semantics [9]. We take $\Lambda = \{\Box, \Diamond\}$ and interpret $\Box$ over $\mathcal{M}$ by the predicate lifting

$$\begin{aligned} [\![\Box]\!]_X(A) &= \{\mathfrak{A} \in \mathcal{M}X \mid A \in \mathfrak{A}\} \\ &= \{\mathfrak{A} \in \mathcal{M}X \mid \exists B \in \mathfrak{A}.\, B \subseteq A\}, \end{aligned}$$

and $\Diamond$ by the corresponding dual lifting, $[\![\Diamond]\!]_X(A) = \{\mathfrak{A} \in \mathcal{M}X \mid (X \setminus A) \notin \mathfrak{A}\} = \{\mathfrak{A} \in \mathcal{M}X \mid \forall B \in \mathfrak{A}.\, B \cap A \neq \emptyset\}$. The arising coalgebraic μ-calculus is known as the *monotone μ-calculus* [19]. When we add propositional atoms and actions, and replace $\mathcal{M}$ with its subfunctor $\mathcal{M}_s$ defined by $\mathcal{M}_s X = \{\mathfrak{A} \in \mathcal{M}X \mid \emptyset \notin \mathfrak{A} \ni X\}$, whose coalgebras are *serial* monotone neighbourhood frames, we arrive at the ambient fixpoint logic of *concurrent dynamic logic* [41] and Parikh's *game logic* [38]. In game logic, actions are understood as atomic games of Angel vs. Demon, and we read $\Box_a \phi$ as 'Angel has strategy to enforce ϕ in game a'. Game logic is then mainly concerned with composite games, formed by the control operators of dynamic logic and additional ones; the semantics can be encoded into fixpoint definitions. For instance, the formula $\nu X.\, p \wedge \Box_a X$ says that Angel can enforce p in the composite game where a is played repeatedly, with Demon deciding when to stop.

4. *Alternating-time μ-calculus:* Fix a set $N = \{1, \dots, n\}$ of *agents*. Using alternative notation from *coalition logic* [40], we present the *alternating-time μ-calculus (AMC)* [1] by modalities $[D]$, $\langle D \rangle$ indexed over *coalitions* $D \subseteq N$, read 'D can enforce' and 'D cannot prevent', respectively. We define a functor $\mathcal{G}$ by

$$\begin{aligned} \mathcal{G}X = \{(k_1, \dots, k_n, f) \mid k_1, \dots, k_n \in \mathbb{N} \setminus \{0\}, \\ f : \left(\textstyle\prod_{i \in N} [k_i]\right) \to X\} \end{aligned}$$

where we write $[k] = \{1, \dots, k\}$ in this example. We understand $(k_1, \dots, k_n, f) \in \mathcal{G}X$ as a one-step concurrent game with k_i available moves for agent $i \in N$, and outcomes in X determined by the *outcome function* f from a joint choice of moves by all the agents. For $D \subseteq N$, we write $S_D = \prod_{i \in D}[k_i]$. Given joint choices $s_D \in S_D$, $s_{\overline{D}} \in S_{\overline{D}}$ of moves for D and $\overline{D} = N \setminus D$ respectively, we write $(s_D, s_{\overline{D}}) \in s_N$ for the joint move of all agents induced in the evident way. In this notation, we interpret the modalities $[D]$ over $\mathcal{G}$ by the predicate lifting

$$\begin{aligned}\llbracket [D] \rrbracket_X(A) = \{(k_1, \dots, k_n, f) \in \mathcal{G}X \mid \\ \exists s_D \in S_D.\, \forall s_{\overline{D}} \in S_{\overline{D}}.\, f(s_D, s_{\overline{D}}) \in A\},\end{aligned}$$

and the modalities $\langle D \rangle$ by dualization. This captures exactly the semantics of the AMC: $\mathcal{G}$-coalgebras are precisely *concurrent game structures* [1], i.e. assign a one-step concurrent game to each state, and $[D]\phi$ says that the agents in D have a joint move such that however the agents in $\overline{D}$ move, the next state will satisfy ϕ. E.g. $\nu Y.\, \mu X.\, (p \wedge [D]Y) \vee [D]X$ says that coalition D can enforce that p is satisfied infinitely often.

We now fix a target formula χ that does not contain free fixpoint variables, assuming w.l.o.g. that χ is *clean*, i.e. that every fixpoint variable is bound by at most one fixpoint operator in χ. For a variable $x \in \mathsf{Var}$ that is bound in χ, we then write $\theta(x)$ to denote *the* formula $\eta X.\psi$ that is a subformula of χ. Let $\mathsf{Cl}(\chi)$ be the *closure* (that is, the set of subformulae) of χ. We have $|\mathsf{Cl}(\chi)| \leq |\chi|$, where $|\chi|$ denotes the number of operators or variables in χ.

We proceed to recall how model checking in the coalgebraic μ-calculus is reduced to computing a nested fixpoint of a particular function [25]:

Definition 16 (Coalgebraic model checking function). Let $\xi : C \to TC$ be a coalgebra, and $U = \mathsf{Cl}(\chi) \times C$. The *(coalgebraic) model checking function* $\alpha_{\mathsf{mc}} : \mathcal{P}(U)^{k+1} \to \mathcal{P}(U)$ is given by putting, for $\mathbf{U} = (U_1, \dots, U_{k+1}) \in \mathcal{P}(U)^{k+1}$,

$$\begin{aligned}\alpha_{\mathsf{mc}}(\mathbf{U}) = &\{(\top, x) \mid (\top, x) \in U\} \cup \\ &\{(\heartsuit\psi, x) \in U \mid \xi(x) \in \llbracket \heartsuit \rrbracket \{y \mid (\psi, y) \in U_1\}\} \cup \\ &\{(\psi \vee \phi, x) \in U \mid \{(\psi, x), (\phi, x)\} \cap U_1 \neq \emptyset\} \cup \\ &\{(\psi \wedge \phi, x) \in U \mid \{(\psi, x), (\phi, x)\} \subseteq U_1\} \cup \\ &\{(\eta X.\, \psi, x) \in U \mid (\psi, x) \in U_1\} \cup \\ &\{(X, x) \mid (\theta(X), x) \in U_{\mathsf{ad}(\theta(X))+1}\}.\end{aligned}$$

Lemma 17 (Coalgebraic model checking [25]). *Let χ be a formula of alternation depth k, $\xi : C \to TC$ a coalgebra, and $x \in C$ a state. Then we have*

$$(\chi, x) \in \mathsf{A}^{\alpha_{\mathsf{mc}}} \text{ if and only if } x \in \llbracket \chi \rrbracket.$$

The *one-step satisfaction problem* consists in deciding whether $t \in \llbracket \heartsuit \rrbracket(W)$, for given $t \in TC$, $\heartsuit \in \Lambda$ and $W \subseteq C$. The time $t(\alpha_{\mathsf{mc}})$ it takes to compute the model checking function α_{mc} hence depends on the time it takes to solve the one-step satisfaction problem for the modal operators at hand. By Corollary 13, we obtain

Corollary 18. *Model checking for coalgebraic μ-calculus formulae of alternation depth k against coalgebras $\xi : C \to TC$ can be done in time $t_2 \cdot t_1$ where $t_2 = \max(t(\alpha_{\mathsf{mc}}), t_1)$, $t_1 = 2n^3(k+1)^2\binom{\log(n(k+1))+k+2}{k+1}$, and $n = |\mathsf{Cl}(\chi)| \cdot |C|$.*

Example 19 (Quasipolynomial-time model checking for graded and probabilistic μ-calculi). In [25, Examples 3.2 and 3.3], it was shown that in the graded and probabilistic cases, the one-step satisfaction problem can be solved in time $\mathcal{O}(\mathsf{size}(\chi) \cdot |C|)$ and $\mathcal{O}(\mathsf{size}(C)^2 \cdot |C|^3)$, respectively; here, $\mathsf{size}(\chi)$ denotes the representation size of the formula χ and $\mathsf{size}(C)$ denotes the representation size of the coalgebra C. We hence obtain the following quasipolynomial upper time bounds for the model checking problems of the respective μ-calculi, both with numbers coded in binary (where $t_1 = 2n^3(k+1)^2\binom{\log(n(k+1))+k+2}{k+1}$ and $n = |\mathsf{Cl}(\chi)| \cdot |C|$):

– for the graded μ-calculus: $\mathcal{O}(t_1 \cdot t_2)$, where $t_2 = \max(\mathsf{size}(\chi) \cdot |C|, t_1)$;
– for the probabilistic μ-calculus: $\mathcal{O}(t_1 \cdot t_2)$, where $t_2 = \max(\mathsf{size}(C)^2 \cdot |C|^3, t_1)$.

Similar bounds, with slightly larger t_2, are obtained for the respective extensions with polynomial inequalities. To the best of our knowledge, these bounds are new. We similarly obtain quasipolynomial bounds for model checking the monotone μ-calculus and the alternating-time μ-calculus. In these cases, the *time* bounds are already in [25], via an encoding into standard parity games; but we emphasize again that the point of our main result (Corollary 13) is not so much the time bound but rather the quasipolynomial bound on the number of iterations – in this case, we obtain that the fixpoint can be computed with quasipolynomially many calls to the one-step satisfaction problem (which at least for the alternating-time case seems also algorithmically preferable to an encoding in parity games with many additional states).

We now consider satisfiability checking for the coalgebraic μ-calculus, which also reduces to the computation of a nested fixpoints of a certain function [26]. We recall the essential notions that are required to define this function; see [26] for details of the construction. We fix a target formula χ of size n and alternation depth k, to be checked for satisfiability. One then has a deterministic parity automaton (see also Example 2) that accepts precisely the *good branches* in tableaux representing prospective models of χ, i.e. the ones not containing infinite deferrals of least fixpoints (which represent eventualities). We work with parity automata in which priorities are assigned to the *transitions* (rather than the states); our automaton thus has the form $(D_\chi, \Sigma, \delta, \beta)$ where D_χ is the set of states; Σ is the alphabet (designed to allow identifying manipulations of formulae happening in the transitions); δ is the transition function; and β assigns priorities to transitions. Since the automaton is deterministic, we can take β to be a function $D_\chi \times \Sigma \to \mathbb{N}$. Recall that such a parity automaton accepts an infinite word w if and only if it has a run for w in which the highest priority that occurs infinitely often is even. We have $|D_\chi| \in \mathcal{O}(2^{\mathcal{O}(nk\log n)})$, and nodes $v \in D_\chi$ are labelled with sets $l(v)$ of formulae. We denote the set of nodes whose labels contain some propositional formula by $\mathsf{prestates}$ and the set of nodes whose labels contain only modal formulae by states; for $v \in \mathsf{prestates}$, ψ_v is a fixed propositional formula from the label of v. The transition function δ tracks sets of formulae according to the logical manipulations described by a given letter from Σ. Besides letters identifying propositional transformations, Σ contains sets of modal formulae describing modal steps; we write $\mathsf{selections} \subseteq \Sigma$ for the set of these letters.

Definition 20 (Coalgebraic satisfiability checking function [26]). For sets $U \subseteq D_\chi$ and $\mathbf{U} = (U_1, \ldots, U_{2nk}) \in \mathcal{P}(U)^{2nk}$, we put

$$\begin{aligned}\alpha_{\mathsf{sat}}(\mathbf{U}) = &\{v \in \mathsf{prestates} \mid \\ &\qquad \exists b \in \{0,1\}.\, \delta(v, (\psi_v, b)) \in U_{\beta(v,(\psi_v,b))}\} \cup \\ &\{v \in \mathsf{states} \mid T(\textstyle\bigcup_{1 \le i \le 2nk} U_i(v)) \cap [\![l(v)]\!]_1 \neq \emptyset\}\end{aligned}$$

where $\beta(v, (\psi_v, b))$ abbreviates $\beta(v, (\psi_v, b), \delta(v, (\psi_v, b)))$ and where

$$\begin{aligned}U_i(v) = \{l(u) \mid u \in X_i, \exists \kappa \in \mathsf{selections}.\\ \delta(v, \kappa) = u, \beta(v, \kappa, u) = i\}.\end{aligned}$$

The *one-step satisfiability problem* is to decide whether

$$T(\textstyle\bigcup_{1 \le i \le 2nk} U_i(v)) \cap [\![l(v)]\!]_1 \neq \emptyset$$

for given U, v. Hence checking whether some $v \in \mathsf{prestates}$ is contained in $\alpha_{\mathsf{sat}}(\mathbf{U})$ for given $\mathbf{U}$ is an instance of the one-step satisfiability problem.

Lemma 21 (Fixpoint characterization of satisfiability [26]). *In the above notation,*

$$v_0 \in \mathsf{A}^{\alpha_{\mathsf{sat}}} \text{ if and only if } \chi \text{ is satisfiable.}$$

Corollary 22. *If the one-step satisfiability problem of a coalgebraic logic can be solved in time $2^{\mathcal{O}(nk \log n)}$, then the satisfiability problem of the μ-calculus over this logic can be solved in time $2^{\mathcal{O}(nk \log n)}$ as well.*

Proof. By the previous Lemma, it suffices to show that $\mathsf{A}^{\alpha_{\mathsf{sat}}}$ can be computed in time $2^{\mathcal{O}(nk \log n)}$. Since we have $2nk < \log |D_\chi|$, $\mathsf{A}^{\alpha_{\mathsf{sat}}}$ can – by Theorem 14 – be computed in time $\mathcal{O}(t(\alpha_{\mathsf{sat}}) \cdot (|D_\chi|(2nk+1))^8)$, where $t(\alpha_{\mathsf{sat}})$ denotes the maximum of $(|D_\chi|(2nk+1))^8$ and the time it takes to compute α_{sat}; by assumption, α_{sat} can be computed in time $2^{\mathcal{O}(nk \log n)}$ so that we have $t(\alpha_{\mathsf{sat}}) \in 2^{\mathcal{O}(nk \log n)}$.

Example 23. It has been shown (e.g. in [26]) that the one-step satisfiability problems of all logics from Example 15 can be solved in time $2^{\mathcal{O}(nk \log n)}$. Hence we obtain an upper bound $2^{\mathcal{O}(nk \log n)}$ for the satisfiability problems of all these logics, in particular including the monotone μ-calculus, the alternating-time μ-calculus, the graded μ-calculus and the (two-valued) probabilistic μ-calculus, even when the latter two are extended with (monotone) polynomial inequalities. This improves on the best previous bounds in all cases.

8 Conclusion

We have shown how to use universal graphs to compute solutions of systems of fixpoint equations $X_i = \eta_i.\, f_i(X_0, \dots, X_k)$ (with the η_i marking least or greatest fixpoints) that use functions $f_i : \mathcal{P}(U)^{k+1} \to \mathcal{P}(U)$ (over a finite set U) and involve $k+1$-fold nesting of fixpoints. Our algorithm needs quasipolynomially many iterations of the functions f_i, and runs in time $\mathcal{O}(q \cdot \max(t(f), q))$, where q is a quasipolynomial in $|U|$ and k and where $t(f)$ is a bound on the time it takes to compute f_i for all i. One consequence of this is that upper time bounds for model checking and satisfiability checking for the coalgebraic μ-calculus improve; in particular, model checking in the coalgebraic μ-calculus can be performed in quasipolynomial time under very mild assumptions on the modalities. In terms of concrete instances, we obtain, e.g., quasipolynomial-time model checking for the graded μ-calculus [32] and the probabilistic μ-calculus [11,35] as new results (corresponding results for, e.g., the alternating-time μ-calculus [1] and the monotone μ-calculus [19] follow as well but have already been obtained in our previous work [25]), as well as improved upper bounds for satisfiability checking in the graded μ-calculus, the probabilistic μ-calculus, the monotone μ-calculus, and the alternating-time μ-calculus. We foresee further applications, e.g. in the computation of fair bisimulations and fair equivalence [27,31] beyond relational systems, e.g. for probabilistic systems.

As in the case of parity games, a natural open question that remains is whether solutions of fixpoint equations can be computed in polynomial time (which would of course imply that parity games can be solved in polynomial time). A more immediate perspective for further investigation is to generalize the recent quasipolynomial variant [39] of Zielonka's algorithm [47] for solving parity games to solving systems of fixpoint equations, with a view to improving efficiency in practice.

References

1. R. Alur, T. Henzinger, and O. Kupferman. Alternating-time temporal logic. *J. ACM*, 49:672–713, 2002.
2. P. Baldan, B. König, C. Mika-Michalski, and T. Padoan. Fixpoint games on continuous lattices. *PACMPL*, 3(POPL):26:1–26:29, 2019.
3. H. Bodlaender, M. Dinneen, and B. Khoussainov. On game-theoretic models of networks. In *Algorithms and Computation, ISAAC 2001*, vol. 2223 of *LNCS*, pp. 550–561. Springer, 2001.
4. U. Boker and K. Lehtinen. On the way to alternating weak automata. In *Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2018*, vol. 122 of *LIPIcs*, pp. 21:1–21:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.
5. F. Bruse, M. Falk, and M. Lange. The fixpoint-iteration algorithm for parity games. In *Games, Automata, Logics and Formal Verification, GandALF 2014*, vol. 161 of *EPTCS*, pp. 116–130, 2014.
6. C. Calude, S. Jain, B. Khoussainov, W. Li, and F. Stephan. Deciding parity games in quasipolynomial time. In *Theory of Computing, STOC 2017*, pp. 252–263. ACM, 2017.

7. K. Chatterjee, W. Dvorák, M. Henzinger, and A. Svozil. Quasipolynomial set-based symbolic algorithms for parity games. In *Logic for Programming, Artificial Intelligence and Reasoning, LPAR 2018*, vol. 57 of *EPiC*, pp. 233–253. EasyChair, 2018.
8. K. Chatterjee, M. Jurdziński, and T. A. Henzinger. Simple stochastic parity games. In M. Baaz and J. A. Makowsky, eds., *Computer Science Logic*, pp. 100–113, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
9. B. F. Chellas. *Modal Logic*. Cambridge University Press, 1980.
10. C. Cîrstea, C. Kupke, and D. Pattinson. EXPTIME tableaux for the coalgebraic μ-calculus. In *Computer Science Logic, CSL 2009*, vol. 5771 of *LNCS*, pp. 179–193. Springer, 2009.
11. C. Cîrstea, C. Kupke, and D. Pattinson. EXPTIME tableaux for the coalgebraic μ-calculus. *Log. Meth. Comput. Sci.*, 7, 2011.
12. C. Cîrstea, A. Kurz, D. Pattinson, L. Schröder, and Y. Venema. Modal logics are coalgebraic. *Comput. J.*, 54:31–41, 2011.
13. T. Colcombet and N. Fijalkow. Universal graphs and good for games automata: New tools for infinite duration games. In *Foundations of Software Science and Computation Structures, FOSSACS 2019*, vol. 11425 of *LNCS*, pp. 1–26. Springer, 2019.
14. W. Czerwinski, L. Daviaud, N. Fijalkow, M. Jurdzinski, R. Lazic, and P. Parys. Universal trees grow inside separating automata: Quasi-polynomial lower bounds for parity games. In *Symposium on Discrete Algorithms, SODA 2019*, pp. 2333–2349. SIAM, 2019.
15. G. D'Agostino and A. Visser. Finality regained: A coalgebraic study of Scott-sets and multisets. *Arch. Math. Logic*, 41:267–298, 2002.
16. L. Daviaud, M. Jurdzinski, and R. Lazic. A pseudo-quasi-polynomial algorithm for mean-payoff parity games. In *Logic in Computer Science, LICS 2018*, pp. 325–334. ACM, 2018.
17. A. Dawar and E. Grädel. The descriptive complexity of parity games. In *Computer Science Logic, CSL 2008*, vol. 5213 of *LNCS*, pp. 354–368. Springer, 2008.
18. E. A. Emerson, C. Jutla, and A. P. Sistla. On model checking for the μ-calculus and its fragments. *Theor. Comput. Sci.*, 258:491–522, 2001.
19. S. Enqvist, F. Seifan, and Y. Venema. Monadic second-order logic and bisimulation invariance for coalgebras. In *Logic in Computer Science, LICS 2015*. IEEE, 2015.
20. R. Fagin and J. Halpern. Reasoning about knowledge and probability. *J. ACM*, 41(2):340–367, 1994.
21. J. Fearnley, S. Jain, B. de Keijzer, S. Schewe, F. Stephan, and D. Wojtczak. An ordered approach to solving parity games in quasi-polynomial time and quasi-linear space. *STTT*, 21(3):325–349, 2019.
22. E. Grädel, W. Thomas, and T. Wilke, eds. *Automata, Logics, and Infinite Games: A Guide to Current Research*, vol. 2500 of *LNCS*. Springer, 2002.
23. E. M. Hahn, S. Schewe, A. Turrini, and L. Zhang. A simple algorithm for solving qualitative probabilistic parity games. In S. Chaudhuri and A. Farzan, eds., *Computer Aided Verification*, pp. 291–311, Cham, 2016. Springer International Publishing.
24. I. Hasuo, S. Shimizu, and C. Cîrstea. Lattice-theoretic progress measures and coalgebraic model checking. In *Principles of Programming Languages, POPL 2016*, pp. 718–732. ACM, 2016.
25. D. Hausmann and L. Schröder. Game-based local model checking for the coalgebraic μ-calculus. In *Concurrency Theory, CONCUR 2019*, LIPIcs. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

26. D. Hausmann and L. Schröder. Optimal satisfiability checking for arithmetic μ-calculi. In *Foundations of Software Science and Computation Structures, FOSSACS 2019*, vol. 11425 of *LNCS*, pp. 277–294. Springer, 2019.
27. T. Henzinger and S. Rajamani. Fair bisimulation. In *Tools and Algorithms for Construction and Analysis of Systems, TACAS 2000*, vol. 1785 of *LNCS*, pp. 299–314. Springer, 2000.
28. M. Jurdziński. Small progress measures for solving parity games. In H. Reichel and S. Tison, eds., *Symposium on Theoretical Aspects of Computer Science, STACS 2000*, pp. 290–301, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.
29. M. Jurdzinski and R. Lazic. Succinct progress measures for solving parity games. In *Logic in Computer Science, LICS 2017*, pp. 1–9. IEEE Computer Society, 2017.
30. D. Kozen. Results on the propositional μ-calculus. *Theor. Comput. Sci.*, 27:333–354, 1983.
31. O. Kupferman, N. Piterman, and M. Vardi. Fair equivalence relations. In *Verification: Theory and Practice*, vol. 2772 of *LNCS*, pp. 702–732. Springer, 2003.
32. O. Kupferman, U. Sattler, and M. Vardi. The complexity of the graded μ-calculus. In *Automated Deduction, CADE 02*, vol. 2392 of *LNCS*, pp. 423–437. Springer, 2002.
33. C. Kupke, D. Pattinson, and L. Schröder. Reasoning with global assumptions in arithmetic modal logics. In *Fundamentals of Computation Theory, FCT 2015*, vol. 9210 of *LNCS*, pp. 367–380. Springer, 2015.
34. K. Lehtinen. A modal μ perspective on solving parity games in quasi-polynomial time. In *Logic in Computer Science, LICS 2018*, pp. 639–648. ACM, 2018.
35. W. Liu, L. Song, J. Wang, and L. Zhang. A simple probabilistic extension of modal mu-calculus. In *International Joint Conference on Artificial Intelligence, IJCAI 2015*, pp. 882–888. AAAI Press, 2015.
36. D. E. Long, A. Browne, E. M. Clarke, S. Jha, and W. R. Marrero. An improved algorithm for the evaluation of fixpoint expressions. In D. L. Dill, ed., *Computer Aided Verification*, pp. 338–350, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.
37. D. Niwinski. On fixed-point clones (extended abstract). In *Automata, Languages and Programming, ICALP 1986*, vol. 226 of *LNCS*, pp. 464–473. Springer, 1986.
38. R. Parikh. The logic of games and its applications. *Ann. Discr. Math.*, 24:111–140, 1985.
39. P. Parys. Parity games: Zielonka's algorithm in quasi-polynomial time. In *Mathematical Foundations of Computer Science, MFCS 2019*, LPIcs. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.
40. M. Pauly. A modal logic for coalitional power in games. *J. Logic Comput.*, 12:149–166, 2002.
41. D. Peleg. Concurrent dynamic logic. *J. ACM*, 34:450–479, 1987.
42. J. Rutten. Universal coalgebra: A theory of systems. *Theor. Comput. Sci.*, 249:3–80, 2000.
43. L. Schröder and D. Pattinson. Strong completeness of coalgebraic modal logics. In *Theoretical Aspects of Computer Science, STACS 09*, pp. 673–684. Schloss Dagstuhl – Leibniz-Zentrum für Informatik; Dagstuhl, Germany, 2009.
44. L. Schröder and Y. Venema. Completeness of flat coalgebraic fixpoint logics. *ACM Trans. Comput. Log.*, 19(1):4:1–4:34, 2018.
45. H. Seidl. Fast and Simple Nested Fixpoints. *Universität Trier, Mathematik/Informatik, Forschungsbericht*, 96-05, 1996.
46. Y. Venema. Lectures on the modal μ-calculus. Lecture notes, Institute for Logic, Language and Computation, Universiteit van Amsterdam, 2008.

47. W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theor. Comput. Sci.*, 200(1-2):135–183, 1998.