Automated Approach for Solving Infinite-state Polynomial Reachability Games

Krishnendu Chatterjee[1] , **Ehsan Kafshdar Goharshady**[1] , **Mehrdad Karrabi**[1] ,
Maximilian Seeliger[2] and **Đorđe Žikelić**[3]
[1] Institute of Science and Technology Austria (ISTA)
[2] ETH Zurich
[3] Singapore Management University
{krishnendu.chatterjee, ehsan.goharshady, mehrdad.karrabi}@ist.ac.at, mseeliger@ethz.ch,
dzikelic@smu.edu.sg

Abstract

Reachability games are two-player games played on a graph, where the objective of REACH player is to reach the target set whereas the objective of SAFE player is to stay away from the target set. Reachability games have important applications in artificial intelligence and reactive synthesis, and many of these applications give rise to infinite-state reachability games. In this paper, we study turn-based reachability games on infinite-state graphs defined over valuations of a finite set of real variables. We consider the problem of determining the existence of and computing a winning strategy for REACH player. Our contributions are twofold. First, we propose ranking certificates for reachability games, a sound and complete proof rule for proving that REACH player has a winning strategy from the specified initial state. Second, we consider polynomial reachability games, where transitions and objectives are described by polynomial constraints over real variables, and propose a fully automated algorithm for computing a winning strategy for REACH player together with a formal correctness witness in the form of a ranking certificate. The algorithm is sound, semi-complete, and runs in sub-exponential time. Our experiments demonstrate the ability of our method to solve challenging examples from the literature that were out of the reach of existing methods. Specifically, for the classical Cinderella-Stepmother game, we are able to compute an optimal winning strategy for an arbitrary precision parameter for the first time.

1 Introduction

A standard way of modeling adversarial semantics is considering games played on graphs defined as a turn-based game on a graph where the nodes are partitioned between the two players. Such graph games have been widely studied in the literature and have many applications in formal verification, control synthesis, and artificial intelligence. For instance, alternating Turing machine [Chandra *et al.*, 1981], decision making in multi-agent systems [Xu *et al.*, 2022; Zhang *et al.*, 2021], planning [Ahuja *et al.*, 2022; Skrynnik *et al.*, 2024], opponent modeling [Yu *et al.*, 2022; Zhang *et al.*, 2021; Li *et al.*, 2024] and controller synthesis [Asarin *et al.*, 1994; Alur *et al.*, 1997] are all modeled via games on graphs. Given a winning condition and a game graph, solving the game corresponds to deciding which player (if any) has a winning strategy, together with finding such a winning strategy.

One of the most fundamental and well-studied problems in graph games considers *reachability* and *safety* objectives, where the goal of the first (REACH) player is to force a specific target set of states to be reached, while the second (SAFE) player's goal is to prevent this from happening. In safety-critical systems, a winning strategy for SAFE induces a safe controller, and a winning strategy for REACH corresponds to a buggy behavior in the system. Similarly, in planning, a winning strategy for REACH induces a controller that reaches the desired target despite the adversarial behavior of the environment. There are other objectives that have been considered in graph games. For instance, LTL and ω-regular logical specifications [de Alfaro *et al.*, 1998; Chatterjee and Henzinger, 2014] are qualitative objectives, and mean payoff, discounted sum, and total payoff are quantitative objectives [Steeples *et al.*, 2021; Vorobeychik and Singh, 2012; Sokota *et al.*, 2021] which are not the focus of this work.

Graph games can be defined on a finite or infinite set of states. It is well known that reachability games, in both cases, have memoryless determinacy, i.e., from each state either REACH or SAFE wins with a strategy that depends only on the current state of the game, not the history of the play [Martin, 1975; Mazala, 2001]. Infinite state games can be represented via many tools, e.g., pushdown automata [Bouajjani *et al.*, 1997], timed automata [Alur and Dill, 1994], or transition systems [Heim and Dimitrova, 2024]. While determining the winner from each state in finite state games can be done in linear time [Chatterjee, 2008; de Alfaro *et al.*, 1998], Rice's theorem [Rice, 1953] shows that the same task in general infinite state games is undecidable even on games where one of the players has only one possible action in each state. The undecidability result implies that one cannot hope for a sound and complete algorithm for solving reachability games. However, many works have examined restricted classes of games and proposed decision procedures on them, e.g. [Farzan and Kincaid, 2018] focuses on games definable in linear arithmetic while [Faella and Parlato, 2023] studies games where SAFE has a bounded number of choices in each turn. In our

case, we consider games defined via polynomial expressions over game variables and propose a sound and semi-complete algorithm for solving them.

Contributions. In this work, we consider turn-based infinite-state reachability games defined over the set of all valuations of finitely many real-valued variables, also sometimes called arithmetic reachability games [Farzan and Kincaid, 2018]. We consider the problem of determining the existence of and computing a winning strategy for REACH player. Our contributions are as follows:

- *Theory.* We introduce *ranking certificates for reachability games*, a sound and complete proof rule for proving the existence of a winning strategy of REACH player. Intuitively, our ranking certificate is a function which assigns a real value to each state of the game that is required to be non-negative and to strictly decrease along a game play until the target set is reached, thus acting as a witness that the game is always making progress towards reaching the target set. Our ranking certificates draw insight from the classical notion of ranking functions for proving termination of programs [Floyd, 1967] and generalize it to the setting of reachability games (Section 3).
- *Automation.* We present a fully automated algorithm for computing a winning strategy for REACH player together with a formal correctness witness in the form of a ranking certificate in *polynomial* reachability games, where transitions and the target set are described by polynomial constraints over real variables. We show that our algorithm is sound, semi-complete, and runs in sub-exponential time (Section 4).
- *Experiments.* We implement our algorithm and perform case studies to show its practical applicability. We consider the classical Cinderella-Stepmother game [Bodlaender *et al.*, 2012], for which our algorithm is able to compute a winning strategy for REACH player for an arbitrary precision parameter for the first time (Section 5).

Novelty and limitations. Our contributions significantly advance the study of infinite-state reachability games.

- *Theoretical improvements.* The previous sound and semi-complete approach with complexity guarantees is limited to linear arithmetic and has double-exponential complexity [Farzan and Kincaid, 2018]. In contrast, our approach handles polynomial arithmetic and has sub-exponential complexity, i.e., we solve a wider range of games with lower complexity.
- *Solving classical games.* For the Cinderella-Stepmother game (Example 1), while it is known that for bucket capacity less than 2 REACH has a winning strategy, the method of [Beyene *et al.*, 2014] could solve the game only up to capacity 1.4. This was later improved by the semi-complete method of [Farzan and Kincaid, 2018] for capacity up to 1.7. More recently, [Samuel *et al.*, 2021] proposed a method that handles capacity up to $2-10^{-20}$ but provides no completeness guarantees. Our approach is the first to solve this game for bucket capacity $2-\epsilon$, where $\epsilon > 0$ is a variable, i.e., we can solve the game for arbitrary precision parameter.

Our approach has two main limitations: (i) our algorithm applies only to polynomial games, and extension to more general classes of games is an interesting direction, and (ii) it focuses primarily on reachability games, and extension to other objectives is another interesting direction of research.

Related work. Infinite-state reachability games have received increased interest in recent years. One line of work reduces this problem to logical satisfiability problems and then uses off-the-shelf solvers to check satisfiability. For instance, [Beyene *et al.*, 2014] reduce the problem to constrained Horn clause (CHC) solving and [Faella and Parlato, 2023] use a similar translation for a class of games in which the number of choices of SAFE player is bounded. [Farzan and Kincaid, 2018] consider linear reachability games and propose a method for solving them by gradually increasing the finite-time horizon of the game, until a winning strategy for one of the players is computed. Another line of work is based on using fixed-point computation-based methods to compute winning strategies [Heim and Dimitrova, 2024; Schmuck *et al.*, 2024; Unno *et al.*, 2023]. These methods do not provide completeness guarantees and the fixed-point computation may not terminate. [Samuel *et al.*, 2021] develop a fixed-point computation-based method and show its scalability by outperforming other existing methods on challenging benchmarks. [Baier *et al.*, 2021] use Craig interpolation to achieve sufficient sub-goals and necessary sub-goals for REACH player to divide the game into simpler sub-games, which are easier to solve. In comparison to all these works, our automated method is the first to provide the following desirable features: (i) it is applicable to polynomial reachability games (unlike [Farzan and Kincaid, 2018]), (ii) it provides semi-completeness guarantees (unlike [Heim and Dimitrova, 2024]), and (iii) it runs in sub-exponential time.

2 Preliminaries

In this section, we define the model of infinite-state reachability games that we consider in this work. We use $\mathbb{R}$ and $\mathbb{N}$ for the sets of real numbers and of non-negative integers.

Definition 1 (Reachability game)**.** *A two-player (infinite-state) reachability game between players* REACH *and* SAFE *is a tuple* $\mathcal{G} = (\mathcal{V}, X, \mathbf{x}_{init}, \mathcal{L}_{Reach}, \mathcal{L}_{Safe}, l_{init}, \mapsto)$ *where:*

- $\mathcal{V}$ *is a finite set of real-valued* variables*;*
- $X \subseteq \mathbb{R}^{|\mathcal{V}|}$ *is the set of allowed valuations of variables;*
- $\mathbf{x}_{init} \in X$ *is the* initial variable valuation*;*
- $\mathcal{L}_{Reach}$ *and* $\mathcal{L}_{Safe}$ *are two disjoint sets of labels. We use* $\mathcal{L} = \mathcal{L}_{Reach} \cup \mathcal{L}_{Safe}$ *for the set of all* labels *in the game;*
- $l_{init} \in \mathcal{L}$ *is the* initial label *of the game;*
- $\mapsto$ *is a finite set of* transitions*. Each* $\tau \in \mapsto$ *is a tuple* $\tau = (l, l', G_\tau, U_\tau)$ *where* l *and* l' *are the source and the target labels,* $G_\tau \subseteq X$ *is the guard of the transition that specifies for which variable valuations the transition can be taken, and* U_τ *is a relation in* $G_\tau \times X$ *that specifies the set of all possible (not necessarily unique) variable valuation updates upon taking the transition.*

A *state* in the game is a tuple $(l, \mathbf{x})$ consisting of a label $l \in \mathcal{L}$ and a variable valuation $\mathbf{x} \in X$. The state $(l_{init}, \mathbf{x}_{init})$ is said to be the *initial state* of the game. For two states $s_1 = (l_1, \mathbf{x}_1)$ and $s_2 = (l_2, \mathbf{x}_2)$, we say that s_2 is a *successor* of s_1 and write $s_1 \mapsto s_2$ if there exists a

transition τ with source label l_1 and target label l_2 such that $\mathbf{x}_1 \in G_\tau$ and $(\mathbf{x}_1, \mathbf{x}_2) \in U_\tau$. For each state $s \in \mathcal{S}$, we use $succ(s) := \{s' | s \mapsto s'\}$ to denote the set of successors of s. We write $\mathcal{S}_{Reach} = \mathcal{L}_{Reach} \times X$ and $\mathcal{S}_{Safe} = \mathcal{L}_{Safe} \times X$ and say that these states *belong to* players REACH and SAFE, and write $\mathcal{S} = \mathcal{S}_{Reach} \cup \mathcal{S}_{Safe}$ for the set of all states.

We assume that for each label $l \in \mathcal{L}$ we have $\bigcup_{(l,l',G_\tau,U_\tau)\in\mapsto} G_\tau = X$, meaning that it is always possible to take at least one transition and that there are no deadlock states in the game. This assumption is imposed without loss of generality, as we may add two auxiliary labels belonging to the opposite player with self-loop transitions, to which the game transitions whenever a deadlock state is reached. Moreover, we assume that $G_\tau \cap G_{\tau'} = \emptyset$ for any two transitions τ, τ' outgoing from the same label l, which is also imposed without loss of generality as one can merge the update relations of two transitions at states in which the guards of both transitions are satisfied. These two assumptions together ensure that, for every state $s = (l, \mathbf{x})$, there is a unique transition (l, l', G_τ, U_τ) with $\mathbf{x} \in G_\tau$.

Semantics. A reachability game is played between two players REACH and SAFE, which indefinitely move a token along the states of the game. The token is initially placed at the initial state $(l_{init}, \mathbf{x}_{init})$. Then, in each turn, if the token is in a state $s = (l, \mathbf{x})$, the player to whom the state s belongs gets to move the token. That player takes the unique transition $\tau = (l, l', G_\tau, U_\tau)$ with $\mathbf{x} \in G_\tau$ and chooses a variable valuation $\mathbf{x}'$ such that $(\mathbf{x}, \mathbf{x}') \in U_\tau$. The token is then moved to the successor state $(l', \mathbf{x}')$. This is repeated indefinitely and gives rise to an infinite sequence $\rho = (s_0, s_1, \dots)$ where $s_0 = s_{init}$ and $s_i \mapsto s_{i+1}$ for all $i \in \mathbb{N}$, which we call a *play*. A *finite play* is a finite prefix of a play.

Strategies. A strategy of a player $P \in \{\text{REACH}, \text{SAFE}\}$ is a function $\sigma_P \colon \mathcal{S}^* \times \mathcal{S}_P \to \mathcal{S}$ which maps every finite play ending in a state that belongs to the player P to a successor state of the last state. A strategy σ_P is said to be *memoryless*, if for every two finite plays h and h' that end in the same state belonging to player P, we have $\sigma_P(h) = \sigma_P(h')$. Every two strategies σ_{REACH} and σ_{SAFE} of the two players induce a unique play which we denote by $\rho(\sigma_{\text{REACH}}, \sigma_{\text{SAFE}})$.

Winning condition: Reachability. In this work, we study infinite-state reachability games. A reachability game is equipped with a set of *target (or objective) states* $\mathcal{O} \subseteq \mathcal{S}$. A play ρ is said to be *winning for* REACH if it contains a state in the target set, otherwise, it is *winning for* SAFE.

A strategy σ_P of a player $P \in \{\text{REACH}, \text{SAFE}\}$ is said to be *winning* if, for every strategy σ of the other player, the induced play is winning for the player P. It is a classical result in turn-based games that, in reachability games, there exists a winning strategy for $P \in \{\text{REACH}, \text{SAFE}\}$ if and only if there exists a memoryless winning strategy for P [Mazala, 2001]. Hence, we, without loss of generality, restrict our attention to memoryless strategies.

Problem statement. Given a reachability game $\mathcal{G}$ with target set $\mathcal{O}$, the goal is to decide whether REACH has a winning strategy and, if yes, compute such winning strategy.

Example 1. *Fig. 1 shows the Cinderella-Stepmother game, which is a classical example of an infinite-state reachability game [Bodlaender* et al.*, 2012]. The game is defined over five real-valued variables ($b_0, \dots, b_4$) which denote capacities of five buckets that are placed along a circle. The buckets are initially empty, hence the initial variable valuation is $(0, \dots, 0)$. Each bucket has maximum capacity U, hence the value of the variables can range over $[0, U+1]^5$. There are two labels* C *and* SM *that belong to the Cinderella and the Stepmother players, respectively, with* SM *being the initial label. As can be seen from Fig. 1, the two players play alternatively, starting with the Stepmother. In each turn, Stepmother distributes 1 liter of water among the five buckets (the transition from* SM *to* C*). Then, Cinderella chooses two neighbouring buckets along the circle and empties them (the transition from* C *to* SM*). Stepmother's objective is to make one of the buckets overflow; hence, she is the* REACH *player with the target set of states $\mathcal{O} = \{(\text{C}, \mathbf{b}) | \bigvee_{0\le i\le 4} \mathbf{b}_i > U\}$. On the other hand, Cinderella's objective is to prevent any bucket from overflowing, hence she is the* SAFE *player.*

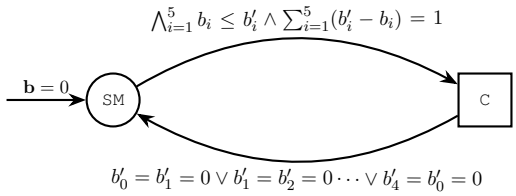

Figure 1: Cinderella-Stepmother game defined over $X = [0, U+1]^5$. REACH owns the label SM with the objective of reaching a state where $b_i > U$ for some i.

3 Ranking Certificate for Reachability Games

We now introduce our ranking certificates for reachability games. The results presented in this section are applicable to the *general class of reachability games*, and not just polynomial reachability games. In what follows, suppose that $\mathcal{G} = (\mathcal{V}, X, \mathbf{x}_{init}, \mathcal{L}_{Reach}, \mathcal{L}_{Safe}, l_{init}, \mapsto)$ is a reachability game with a set of target states $\mathcal{O} \subseteq \mathcal{S}$.

Motivation: Ranking functions for programs. Our certificate is motivated by the notion of ranking functions in the program verification literature [Floyd, 1967; Colón and Sipma, 2001; Podelski and Rybalchenko, 2004], presenting a classical approach to proving termination in programs. A ranking function maps program states to real values and is required to (1) be non-negative at all reachable states and (2) make progress towards termination by decreasing by at least 1 at every step until a terminal state is reached. If such a function exists, then every run of that program terminates as a ranking function cannot remain non-negative while indefinitely decreasing by at least 1 at every step. Hence, ranking functions provide a sound (and complete) proof rule for proving termination of programs [Floyd, 1967].

Our certificate for reachability games. We now introduce our ranking certificates for reachability games, which provide a formal proof of the existence of a winning strategy of the REACH player. Similarly to ranking functions in programs, our ranking certificate for reachability games is a function $f : \mathcal{S} \to \mathbb{R}$ which maps each game state to a real value. However,

unlike ranking functions that need to be non-negative and to make progress towards termination at all reachable program states, we encounter two challenges in generalizing this idea to reachability games:

1. *Not all states are winning for* REACH. In reachability games, there are states from which REACH does not have a winning strategy. Hence, imposing a progress condition on such states may make our ranking certificates overly conservative. To overcome this challenge, we also use our ranking certificates as indicators of whether a state is winning for REACH by using $f(s) \geq 0$ as an indicator that REACH has a winning strategy from a state s. In particular, in (C1) in Definition 2, we only require f to be non-negative at the initial game state. Then, in (C2) and (C3) in Definition 2, which are progress conditions, we only impose the progress condition when $f(s) \geq 0$.
2. *States may belong to different players.* In reachability games, states may belong to either REACH or SAFE player, who gets to choose the successor state to which the game token should be moved. This leads to different requirements on the progress condition. In particular, in (C2) in Definition 2, we impose the progress condition of f for *all* successor states that can be chosen by SAFE. In contrast, in (C3) in Definition 2, we only impose the progress condition of f for *some* successor state that can be chosen by REACH. That way REACH can always ensure to make progress towards the target set, while SAFE cannot prevent progress towards the target set.

The following definition formalizes the intuition discussed above and formalizes our novel notion of reachability certificates for reachability games.

Definition 2 (Ranking certificates for reachability games)**.** *Given a reachability game* $\mathcal{G} = (\mathcal{V}, X, \mathbf{x}_{init}, \mathcal{L}_{Reach}, \mathcal{L}_{Safe}, l_{init}, \mapsto)$ *with a set of target states* $\mathcal{O} \subseteq \mathcal{S}$*, a* ranking certificate *is a function* $f\colon \mathcal{S} \to \mathbb{R}$ *that satisfies the following conditions:*

(C1) $f(s_{init}) \geq 0$*, i.e.* f *is non-negative at the initial state.*

(C2) If $s \in \mathcal{S}_{Safe} \setminus \mathcal{O}$ *and* $f(s) \geq 0$*, then*

$$\forall s' \in succ(s).\, f(s) - 1 \geq f(s') \geq 0.$$

(C3) If $s \in \mathcal{S}_{Reach} \setminus \mathcal{O}$ *and* $f(s) \geq 0$*, then*

$$\exists s' \in succ(s).\, f(s) - 1 \geq f(s') \geq 0.$$

In condition (C3) above, we refer to every $s' \in succ(s)$ *for which the condition is satisfied as a* ranking successor *of* s *with respect to* f*.*

Soundness. The following theorem, proved in the extended version [Chatterjee *et al.*, 2026], establishes soundness of our certificates. Particularly, it shows that if there exists a ranking certificate in a reachability game, then there exists a winning strategy for REACH player. It also shows how a winning strategy can be constructed from the ranking certificate.

Theorem 3 (Soundness)**.** *Suppose that there exists a ranking certificate* f *in a reachability game* $\mathcal{G}$ *with a set of target states* $\mathcal{O} \subseteq \mathcal{S}$*. Then, there exists a memoryless winning strategy for* REACH *player.*

An example of such a strategy is a strategy σ^f_{Reach} *that maps each state* $s \in \mathcal{S}_{Reach}$ *to a ranking successor of* s *if* $f(s) \geq 0$*, and to an arbitrary successor of* s *if* $f(s) < 0$*.*

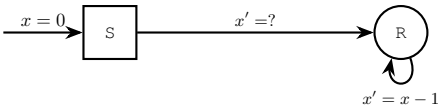

Figure 2: A reachability game demonstrating incompleteness of our ranking certificates.

Completeness. Although theorem 3 shows the soundness of our ranking certificates, it turns out that the proof rule is not complete. The following is an example of a reachability game in which the REACH player has a memoryless winning strategy, but there does not exist a ranking certificate to witness this.

Example 2. *Consider the reachability game in Figure 2. In this game, starting from the initial label* S *that belongs to* SAFE *and the variable valuation* $x = 0$*,* SAFE *adversarially assigns an arbitrary real value to* x*. The game then moves to the label* R *that belongs to* REACH*, and in every step* REACH *decrements the value of* x *by* 1*. The set of target states in this reachability game is* $\mathcal{O} = \{(R, x) | x < 0\}$*.*

It is clear that REACH *has a memoryless winning strategy by always decrementing* x *by* 1*. That is because, regardless of the value of* x *that* SAFE *chooses, the value of* x *will eventually become negative. However, this reachability game turns out not to admit a ranking certificate. To see this, suppose, for the sake of contradiction, that there exists a ranking certificate* f*. Then* f *must satisfy the following, by conditions (C1)-(C3) in Definition 2:*

$$(C1):\ f(S, 0) \geq 0$$
$$(C2):\ \forall x \in \mathbb{R}.\, f(S, 0) - 1 \geq f(R, x) \geq 0$$
$$(C3):\ \forall x \in \mathbb{R}.\, x \geq 0 \wedge f(R, x) \geq 0 \implies f(R, x) - 1 \geq f(R, x - 1) \geq 0$$

The second condition above implies that $f(R, x) \geq 0$ *for all* $x \in \mathbb{R}$*. Thus, it follows from the third condition above and by a simple induction that* $f(R, n) \geq n$ *for* $n \in \mathbb{N}$*. Then by (C2), it is directly implied that* $f(S, 0) \geq n$ *for all* $n \in \mathbb{N}$ *which means* $f(S, 0)$ *cannot have any finite value and the reachability game does not admit a ranking certificate.*

The incompleteness issue in Example 2 arises since SAFE can choose between infinitely many choices for the value of x. This allows SAFE to enforce an arbitrarily large lower bound on the value of f at the initial state of the game, and no finite value would be sufficient. It also hints that, if we were to restrict SAFE by only providing it with finite choices in each step of the game, it may be possible to overcome the incompleteness issue. The following theorem shows that this is indeed the case, and that ranking certificates for reachability games are both sound and complete if SAFE is restricted to having only finitely many choices at each step. The proof is provided in the extended version [Chatterjee *et al.*, 2026].

Theorem 4 (Completeness for finite SAFE choices)**.** *Let* $\mathcal{G}$ *be a reachability game with a set of target states* $\mathcal{O} \subseteq \mathcal{S}$*. Suppose that* SAFE *has only finitely many choices at each step, i.e. for every state* $s \in \mathcal{S}_{Safe}$ *the set of successor states* $succ(s)$ *is finite. If* REACH *has a winning strategy, then there exists a ranking certificate for this reachability game.*

Novelty of certificates. Unlike classical attractor computation (progress measure) [Mazala, 2001] methods that compute the minimal alternating distance, our notion of ranking certificates only requires a strict decrease in its value. Relaxing this optimality constraint allows for much simpler witnesses, i.e. in the form of polynomial functions.

4 Algorithm for Solving Reachability Games

We now present our automated algorithm for proving the existence of and computing a winning strategy of REACH player. Given that this problem is undecidable even in 1-player games due to Rice's theorem, we cannot hope for an algorithm that is both sound and complete. Instead, in what follows we design a *sound and semi-complete* algorithm. By soundness, we mean that the output of our algorithm and the produced winning strategy are guaranteed to be correct. By semi-completeness, we mean that our algorithm is guaranteed to compute a winning strategy for REACH player whenever it exists, for a subclass of reachability games that we formally characterize below.

Assumption: Polynomial reachability games. To enable full automation, we restrict our algorithm to polynomial reachability games. A reachability game $\mathcal{G}$ with target states $\mathcal{O} \subseteq \mathcal{S}$ is said to be *polynomial*, if (i) For every transition $\tau \in \mapsto$, the guard $G_\tau \subseteq \mathbb{R}^{\mathcal{V}}$ is a set of boolean combination of polynomial inequalities over variables $\mathcal{V}$; (ii) For every transition $\tau \in \mapsto$, the update $U_\tau \subseteq G_\tau \times X$ is a set of boolean combination of polynomial inequalities over variables $\mathcal{V} \cup \mathcal{V}'$, where $\mathcal{V}'$ denote the variables $\mathcal{V}$ upon update; (iii) For each label $l \in \mathcal{L}$, the set of target variable valuations $\{\mathbf{x} \mid (l, \mathbf{x}) \in \mathcal{O}\}$ can be represented as a set of a boolean combination of polynomial inequalities over variables $\mathcal{V}$; (iv) The set of allowed variable valuations $X \subseteq \mathbb{R}^{|\mathcal{V}|}$ is a satisfiability set of a boolean combination of polynomial inequalities over variables $\mathcal{V}$.

Algorithm overview. Our algorithm proves the existence of a memoryless winning strategy for REACH by synthesizing a ranking certificate $f : \mathcal{S} \to \mathbb{R}$ and a memoryless winning strategy $\sigma : \mathcal{S}_{Reach} \to \mathcal{S}$ for REACH. These two objects are searched for and computed simultaneously by following a template-based synthesis approach. The algorithm proceeds in four steps. In Step 1, the algorithm fixes symbolic polynomial template expressions that define the ranking certificate f and the strategy σ, one for each label in the game. In Step 2, the algorithm collects a system of constraints over the symbolic template variables that together encode that f defines a valid ranking certificate and that σ is a winning strategy induced by f as in Theorem 3. In Step 3, these constraints are simplified in order to reduce them to a sentence in the purely existentially quantified theory of reals. Finally, in Step 4, the constraints are solved by an off-the-shelf SMT solver, and any solution gives rise to a concrete instance of a valid ranking certificate and a winning strategy. In what follows, we present the details behind each of the four steps.

Step 1: Setting up polynomial templates. The algorithm takes as input a polynomial degree parameter $D \in \mathbb{N}$, and it fixes symbolic polynomial templates of maximal polynomial degree D for expressions that define the ranking certificate f and the strategy σ. Denote by $M^D_{\mathcal{V}} = \{m_0, \ldots, m_k\}$ the set of all monomials of degree at most D over the set of variables $\mathcal{V}$. The template for the strategy σ at each game label $l \in \mathcal{L}$ and variable $v \in \mathcal{V}$ is defined via

$$\sigma^v_l(\mathbf{x}) = \sum_{i=0}^{k} t_i^{l,v} \cdot m_i \tag{1}$$

where each $t_i^{l,v}$ is a real-valued symbolic template variable, and $\sigma^v_l(\mathbf{x})$ evaluates to the value of variable v upon a one-step execution of the strategy σ at state $(l, \mathbf{x})$. The template for the ranking certificate f at each game label $l \in \mathcal{L}$ is defined via

$$f_l(\mathbf{x}) = \sum_{i=0}^{k} s_i^l \cdot m_i \tag{2}$$

where each s_i^l is a real-valued symbolic template variable, and $f_l(\mathbf{x})$ evaluates to the value of the ranking certificate f at state $(l, \mathbf{x})$. The values of symbolic template variables $t_i^{l,v}$ and s_i^l are at this point unspecified, and the goal of template-based synthesis is to compute concrete values of template variables that together give rise to a valid ranking certificate f and a winning strategy σ.

Example 3. *Consider the Cinderella-Stepmother game presented in Example 1, and let $D = 1$ be the polynomial degree parameter. The variable set in this example is $\mathcal{V} = \{b_0, b_1, b_2, b_3, b_4\}$ and set of all monomials of degree at most 1 over $\mathcal{V}$ is $M^1_{\mathcal{V}} = \{1, b_0, b_1, b_2, b_3, b_4\}$. So, in Step 1, the following template is set for $\sigma^{b_0}_R$ which evaluates to the value of variable b_0 upon executing the strategy at state $(R, \mathbf{b})$:*

$$\sigma^{b_0}_R(\mathbf{b}) = t_0^{R,b_0} + t_1^{R,b_0} \cdot b_0 + t_2^{R,b_0} \cdot b_1 + t_3^{R,b_0} \cdot b_2 + t_4^{R,b_0} \cdot b_3 + t_5^{R,b_0} \cdot b_4$$

Similarly, the following template is fixed for f_R which evaluates to the value of f at state $(R, \mathbf{b})$:

$$f_R(\mathbf{b}) = s_0^R + s_1^R \cdot b_0 + s_2^R \cdot b_1 + s_3^R \cdot b_2 + s_4^R \cdot b_3 + s_5^R \cdot b_4$$

Step 2: Collecting defining constraints. The algorithm now collects a system of constraints over the symbolic template variables introduced in the previous step, which together encode that f is a valid ranking certificate and that σ is a strategy induced by f as in Theorem 3. In each constraint, every appearance of f_l and σ^v_l is substituted by the symbolic polynomial template expression introduced in Step 1:

(C1) For condition (C1) in Definition 2, the constraint $f_{l_{init}}(\boldsymbol{x}_{init}) \geq 0$ is collected.

(C2) For condition (C2) in Definition 2, for each label $l \in \mathcal{L}_{Safe}$ and each transition $\tau = (l, l', G_\tau, U_\tau) \in \mapsto$, the algorithm collects the following constraint

$$\forall \mathbf{x}, \mathbf{x}' \in X.$$
$$\Big((\mathbf{x} \vDash \neg \mathcal{O}_l \wedge G_\tau) \wedge (f_l(\mathbf{x}) \geq 0) \wedge (\mathbf{x}, \mathbf{x}') \vDash U_\tau \Big)$$
$$\implies f_l(\mathbf{x}) - 1 \geq f_{l'}(\mathbf{x}') \geq 0,$$

where $\mathcal{O}_l = \{\mathbf{x} \in \mathbb{R}^{\mathcal{V}} \mid (l, \mathbf{x}) \in \mathcal{O}\}$, G_τ and U_τ are all represented in terms of a boolean combination of polynomial constraints over the variables in $\mathcal{V} \cup \mathcal{V}'$, as specified in the algorithm assumption stated above.

(C3) For condition (C3) in Definition 2 and to enforce that for each state s belonging to REACH the state $\sigma(s)$ is a ranking successor of s as defined in Theorem 3, the algorithm collects the following constraint for each label $l \in \mathcal{L}_{Reach}$ and each transition $(l, l', G_\tau, U_\tau) \in \mapsto$

$$\forall \mathbf{x} \in X. \left((\mathbf{x} \vDash \neg \mathcal{O}_l \wedge G_\tau) \wedge (f_l(\mathbf{x}) \geq 0)] \right) \implies (\mathbf{x}, \sigma_l(\mathbf{x})) \vDash U_\tau \wedge f_l(\mathbf{x}) - 1 \geq f_{l'}(\sigma_l(\mathbf{x})) \geq 0,$$

where $\mathcal{O}_l = \{\mathbf{x} \in \mathbb{R}^{\mathcal{V}} \mid (l, \mathbf{x}) \in \mathcal{O}\}$ and G_τ are represented in terms of a boolean combination of polynomial constraints over the variables in $\mathcal{V}$. Intuitively, the right-hand-side of the implication encodes that $\sigma_l(\mathbf{x})$ is a ranking successor of $(l, \mathbf{x})$ as defined in Theorem 3.

Example 4. *Consider again the Cinderella-Stepmother game, where the maximum capacity of each bucket is set to $U = 1.8$. For (C1), the algorithm collects $f_{SM}(0) \geq 0$. For (C2), the following constraint is collected for the transition from label C to label SM, as Cinderella is the SAFE player:*

$$\forall \mathbf{b}, \mathbf{b}' \in [0, 2.8]^5. \Big(\bigwedge_{0 \leq i \leq 4} b_i \leq 1.8 \wedge f_C(\mathbf{b}) \geq 0 \wedge \bigvee_{0 \leq i \leq 4} [\bigwedge_{\substack{0 \leq j \leq 4 \\ j \neq i \\ j \neq (i+1)\%5}} b'_j = b_j \wedge b'_i = b'_{(i+1)\%5} = 0] \Big) \implies f_C(\mathbf{b}) - 1 \geq f_{SM}(\mathbf{b}') \geq 0.$$

Finally, for (C3), a similar constraint is collected for the transition from SM to C, as Stepmother is the REACH player.

Step 3: Removing $\forall$ quantifiers. The collected constraints for conditions (C2) and (C3) are of the form

$$\forall \mathbf{x} \in \mathbb{R}^{\mathcal{V}}. \big(X(\mathbf{x}) \wedge P(\mathbf{x})\big) \implies Q(\mathbf{x}), \tag{3}$$

with $P(\mathbf{x})$, $Q(\mathbf{x})$ and X being boolean combinations of polynomial inequalities over variables $\mathbf{x}$. The presence of universal quantification makes solving these constraints computationally expensive. To make our algorithm more efficient, we apply the translation of [Asadi *et al.*, 2021], which uses Putinar's theorem (for general polynomials) or Farkas' lemma (for degree 1 polynomials) to translate constraints of the form as in eq. (3) into a system of purely existentially quantified polynomial constraints. As shown in [Asadi *et al.*, 2021], the translation is sound in the sense that any satisfying assignment of the resulting system of constraints yields a satisfying assignment of the original system of constraints. Moreover, the translation is complete and guarantees equisatisfiability whenever the satisfiability set of the left-hand-side is compact. For the interest of space, we omit the details of this translation and refer the reader to [Asadi *et al.*, 2021].

Step 4: Solving the constraints. Step 3 results in a system of purely existentially quantified polynomial constraints over real-valued variables. In the last step, our algorithm calls an off-the-shelf SMT solver to solve this system of constraints. If a solution is found, the computed valuation of symbolic template variables gives rise to a concrete instance of the ranking certificate f and the memoryless winning strategy σ. Otherwise, the algorithm returns UNKNOWN.

Theorem 5. *Consider a polynomial reachability game $\mathcal{G} = (\mathcal{V}, X, \mathbf{x}_{init}, \mathcal{L}_{Reach}, \mathcal{L}_{Safe}, l_{init}, \mapsto)$ with target states $\mathcal{O}$:*

1. (Soundness) *If the algorithm returns a solution (f, σ), then f is a valid ranking certificate for this reachability game and σ is a memoryless winning strategy of REACH.*
2. (Semi-completeness) *Whenever the set of possible valuations X is compact, if there exist a polynomial ranking certificate f and a polynomial memoryless winning strategy σ for $\mathcal{G}$ then, for a sufficiently high value of the maximal polynomial degree parameter D, the algorithm is guaranteed to compute them.*
3. (Complexity) *The runtime of the algorithm is sub-exponential in the size of the input game $\mathcal{G}$ and $\mathcal{O}$, for each fixed value of the maximal polynomial degree D.*

The full proof of the theorem is provided in the extended version [Chatterjee *et al.*, 2026].

5 Case Studies and Experimental Results

Implementation. We implemented a prototype of our method, which takes as input a polynomial reachability game and tries to compute a ranking certificate and a winning strategy for REACH. For the last two steps of our algorithm, we used PolyQEnt [Chatterjee *et al.*, 2025] which automates translation of quantified formulas as in eq. (3) into a system of existentially quantified polynomial constraints. Our prototype then uses Z3 [de Moura and Bjørner, 2008] and MathSAT5 [Cimatti *et al.*, 2013] as backend SMT solvers. All our experiments were done on a 11th Gen Intel i5 machine with 16 GB memory and a timeout of 5 minutes.

Heuristic. We implement the following simple heuristic, which helps improve the efficiency of our method. The heuristic applies the $pre : \mathcal{P}(S) \to \mathcal{P}(S)$ operator once to expand the target set $\mathcal{O}$. For a set of states $A \subseteq \mathcal{S}$, the set $pre(A)$ is the set of states that are either (i) REACH states that have at least one successor in A, or (ii) SAFE states whose all successors are in A. This is the classical one-step attractor operator, which is common in the analysis of games on graphs. We apply the pre operator once to replace $\mathcal{O}$ by $pre(\mathcal{O})$. Note that REACH player has a 1-step winning strategy from $pre(\mathcal{O})$, hence this heuristic is sound.

We employ this heuristic because we observed that, in many reachability games in the literature, the last step of the game is qualitatively different from earlier steps. For instance, in the last step of the Cinderella-Stepmother game in Example 1, the stepmother will pour 1 liter of water into the bucket that will overflow. We observed that considering $pre(\mathcal{O})$ rather than $\mathcal{O}$ improves the efficiency of our tool.

Baselines. We consider two state-of-the-art infinite-state reachability game solvers as baselines: (i) GenSys [Samuel *et al.*, 2021], which uses a classical fixed-point algorithm generalized to the infinite-state setting, and (ii) rpgsolve [Heim and Dimitrova, 2024], which introduces acceleration into fixed-point computation to avoid divergence. We choose these since they are recent methods that outperformed previous methods in their experiments.

Case study: Cinderella-Stepmother game. We experimentally evaluate our method on two variants of the Cinderella-Stepmother game. The first is the classical variant of the

game [Bodlaender *et al.*, 2012], which was presented in Example 1. The second is a non-linear variant of the game in which the stepmother is allowed to distribute 1 liter of water across five buckets in any way that ensures $\|\mathbf{b} - \mathbf{b}'\|_2 \leq 1$, rather than $\|\mathbf{b} - \mathbf{b}'\|_1 \leq 1$ as in Example 1. This introduces non-linearity in the transition relation from SM to C, which can only be expressed in polynomial arithmetic.

In both variants of the game, the stepmother has a winning strategy for any bucket capacity $U < 2$. For example, for the first (classical) variant and for any fixed bucket capacity $U < 2$, the following define a valid ranking certificate and a REACH winning strategy:

$$f(l, \mathbf{b}) = \begin{cases} \frac{U - \max(b_1, b_3)}{2 - U} \times 2 & l = \texttt{SM} \\ \frac{U - \max(b_1, b_3)}{2 - U} \times 2 + 1 & l = \texttt{C} \end{cases}$$

$$\sigma_{\texttt{SM}}(\mathbf{b}) = \begin{cases} b'_1 = b_1 + 1 & b_1 > U - 1 \\ b'_3 = b_3 + 1 & b_3 > U - 1 \\ b'_1 = b'_3 = \frac{b_1 + b_3 + 1}{2} & o.w. \end{cases}$$

Experimental setup. As discussed in the Introduction, prior methods were able to solve the classical (first) variant of the Cinderella-Stepmother game for different fixed values of the maximum bucket capacity $U < 2$. However, no prior method could compute a winning strategy for REACH player which works for any bucket capacity $2-\epsilon$, where $\epsilon > 0$ is a symbolic variable which is provided as strategy input.

To evaluate our method and the baselines, for both game variants we first consider different fixed values of bucket capacities $U \in \{1.5, 1.7, 1.9, 2 - 10^{-10}\}$. We then consider the bucket capacity $2 - \epsilon$, where $\epsilon > 0$ is a symbolic variable. To implement this, we declare ϵ as a variable, add the constraints $\epsilon > 0$ and $U = 2 - \epsilon$ to the set of allowed valuations of variables X, and add $\epsilon' = \epsilon$ to each transition. This is supported by our tool and GenSys, but not by rpgsolve.[1]

Experimental results. The results are shown in Table 1. Our method is able to solve the classical (first) variant of the Cinderella-Stepmother game for all considered bucket capacities in less than 1s, including the most general case $2 - \epsilon$ where $\epsilon > 0$ is a symbolic variable. In contrast, both GenSys and rpgsolve fail on the largest fixed bucked capacity $2 - 10^{-10}$ and on the most general case $2 - \epsilon$. The results of our method are a significant achievement, given that automated solving of the Cinderella-Stepmother gamefor the general case of bucket capacity $2 - \epsilon$ was open and this is the first time it is solved by an automated method.

Our method is also able to solve the polynomial (second) variant of the Cinderella-Stepmother game for all considered fixed bucket capacities in less than 1s, but it fails on the most general case $2 - \epsilon$. However, it significantly outperforms the baselines on this more challenging benchmark as both baselines fail to compute a winning strategy for any considered bucket capacity. This shows that our method advances the state of the art in infinite-state reachability game solving on benchmarks that were beyond the reach of existing methods.

[1]We also tried providing rpgsolve with a variant in which Cinderella chooses the value of $\epsilon > 0$ in the first step, but this is also not supported as the syntax of rpgsolve as it requires SAFE player to only have finitely many available actions at each step.

	U	Ours	GenSys	rpgsolve
Classical	1.5	0.4	**0.3**	0.4
	1.7	0.4	0.5	**0.3**
	1.9	**0.3**	2.8	0.7
	$2 - 10^{-10}$	**0.4**	TO	TO
	$2 - \epsilon$	**0.9**	TO	NS
Non-linear	1.5	**0.4**	TO	TO
	1.7	**0.4**	TO	TO
	1.9	**0.5**	TO	TO
	$2 - 10^{-10}$	**0.6**	TO	TO
	$2 - \epsilon$	TO	TO	NS

Table 1: Our experimental results on two variants of the Cinderella-Stepmother game. Each cell displays the time (in seconds) it takes for a tool to solve the respective benchmark. TO and NS stand for "timeout" and "not supported".

Second Benchmark. As a second benchmark, we consider a game that models a faulty robot. Intuitively, the robot is supposed to mix two chemicals to obtain a mixture that has a specific density. However, the robot is faulty and might leak some extra amounts of chemicals in each step. This can be formally modeled by a pair (a, b) where a is the amount of the first substance, and b is the amount of the second one. In each turn, as the REACH player, the robot can add at most 1 liter from the combination of the chemicals to the mixture. On the other hand, the faultiness of the robot is modeled by an adversary that can increase each of a and b by 0.2 in every other step. The objective of the robot is to make at least 10 liters of the mixture that contains at least 10% of the first chemical, i.e. the goal is to reach (a, b) such that $a > 9b$ and $a + b > 10$. Given that both players have infinite action-spaces, the game is not supported by rpgoslve.

Results. We ran our tool together with the baselines on this game. While our method solves the game in 44 seconds, GenSys does not terminate in 5 minutes. Once more, this example shows the efficiency of our approach for solving reachability games that are difficult to solve for other approaches.

6 Conclusion

We introduced ranking certificates for reachability games, a new sound and complete proof rule for proving existence of winning strategies of REACH player in infinite-state reachability games. We then presented an automated method for solving infinite-state polynomial reachability games using ranking certificates and template-based synthesis. Our automated method is the first to provide: (i) soundness and semi-completeness guarantees, (ii) sub-exponential runtime, and (iii) support for polynomial reachability games. Our prototype outperforms state-of-the-art tools and for the first time solves the classical Cinderella-Stepmother game with general bucket capacity of $2 - \epsilon$, for any symbolic $\epsilon > 0$. Looking forward, extending our method to broader objectives (e.g., ω-regular and quantitative objectives) and to non-polynomial games are interesting directions of future work.

Acknowledgements

This research was supported by Vienna Science and Technology Fund (WWTF), State of Lower Austria [Grant ID 10.47379/ICT25017], ERC CoG 863818 (ForM-SMArt), and Austrian Science Fund (FWF) 10.55776/COE12.

References

[Ahuja *et al.*, 2022] Hansin Ahuja, Lynnette Hui Xian Ng, and Kokil Jaidka. Using graph-aware reinforcement learning to identify winning strategies in diplomacy games (student abstract). In *AAAI*, 2022.

[Alur and Dill, 1994] Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 1994.

[Alur *et al.*, 1997] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. In *FOCS*, 1997.

[Asadi *et al.*, 2021] Ali Asadi, Krishnendu Chatterjee, Hongfei Fu, Amir Kafshdar Goharshady, and Mohammad Mahdavi. Polynomial reachability witnesses via stellensätze. In *PLDI*, 2021.

[Asarin *et al.*, 1994] Eugene Asarin, Oded Maler, and Amir Pnueli. Symbolic controller synthesis for discrete and timed systems. In *Hybrid Systems*, 1994.

[Baier *et al.*, 2021] Christel Baier, Norine Coenen, Bernd Finkbeiner, Florian Funke, Simon Jantsch, and Julian Siber. Causality-based game solving. In *CAV*, 2021.

[Beyene *et al.*, 2014] Tewodros A. Beyene, Swarat Chaudhuri, Corneliu Popeea, and Andrey Rybalchenko. A constraint-based approach to solving games on infinite graphs. In *POPL*, 2014.

[Bodlaender *et al.*, 2012] Marijke H. L. Bodlaender, Cor A. J. Hurkens, Vincent J. J. Kusters, Frank Staals, Gerhard J. Woeginger, and Hans Zantema. Cinderella versus the wicked stepmother. In *Theoretical Computer Science - 7th IFIP TC 1/WG 2.2 International Conference, TCS 2012*, 2012.

[Bouajjani *et al.*, 1997] Ahmed Bouajjani, Javier Esparza, and Oded Maler. Reachability analysis of pushdown automata: Application to model-checking. In *CONCUR*, 1997.

[Chandra *et al.*, 1981] Ashok K. Chandra, Dexter Kozen, and Larry J. Stockmeyer. Alternation. *J. ACM*, 1981.

[Chatterjee and Henzinger, 2014] Krishnendu Chatterjee and Monika Henzinger. Efficient and dynamic algorithms for alternating büchi games and maximal end-component decomposition. *J. ACM*, 2014.

[Chatterjee *et al.*, 2025] Krishnendu Chatterjee, Amir Kafshdar Goharshady, Ehsan Kafshdar Goharshady, Mehrdad Karrabi, Milad Saadat, Maximilian Seeliger, and Djordje Zikelic. Polyqent: A polynomial quantified entailment solver. In *ATVA*, 2025.

[Chatterjee *et al.*, 2026] Krishnendu Chatterjee, Ehsan Kafshdar Goharshady, Mehrdad Karrabi, Maximilian Seeliger, and Đorđe Žikelić. Automated approach for solving infinite-state polynomial reachability games, 2026.

[Chatterjee, 2008] Krishnendu Chatterjee. Linear time algorithm for weak parity games. *CoRR*, abs/0805.1391, 2008.

[Cimatti *et al.*, 2013] Alessandro Cimatti, Alberto Griggio, Bastiaan Schaafsma, and Roberto Sebastiani. The MathSAT5 SMT Solver. In *TACAS*, 2013.

[Colón and Sipma, 2001] Michael Colón and Henny Sipma. Synthesis of linear ranking functions. In *TACAS*, 2001.

[de Alfaro *et al.*, 1998] Luca de Alfaro, Thomas A. Henzinger, and Orna Kupferman. Concurrent reachability games. In *FOCS*, 1998.

[de Moura and Bjørner, 2008] Leonardo Mendonça de Moura and Nikolaj S. Bjørner. Z3: an efficient SMT solver. In *TACAS*, 2008.

[Faella and Parlato, 2023] Marco Faella and Gennaro Parlato. Reachability games modulo theories with a bounded safety player. In *AAAI*, 2023.

[Farzan and Kincaid, 2018] Azadeh Farzan and Zachary Kincaid. Strategy synthesis for linear arithmetic games. *Proc. ACM Program. Lang.*, 2018.

[Floyd, 1967] Robert W. Floyd. Assigning meanings to programs. *Proceedings of Symposium on Applied Mathematics*, 1967.

[Heim and Dimitrova, 2024] Philippe Heim and Rayna Dimitrova. Solving infinite-state games via acceleration. *Proc. ACM Program. Lang.*, 2024.

[Li *et al.*, 2024] Junkang Li, Bruno Zanuttini, and Véronique Ventos. Opponent-model search in games with incomplete information. In *AAAI*, 2024.

[Martin, 1975] Donald A Martin. Borel determinacy. *Annals of Mathematics*, 1975.

[Mazala, 2001] René Mazala. Infinite games. In *Automata, Logics, and Infinite Games*, 2001.

[Podelski and Rybalchenko, 2004] Andreas Podelski and Andrey Rybalchenko. A complete method for the synthesis of linear ranking functions. In *VMCAI*, 2004.

[Rice, 1953] Henry Gordon Rice. Classes of recursively enumerable sets and their decision problems. *Transactions of the American Mathematical society*, 1953.

[Samuel *et al.*, 2021] Stanly Samuel, Deepak D'Souza, and Raghavan Komondoor. Gensys: a scalable fixed-point engine for maximal controller synthesis over infinite state spaces. In *ESEC/FSE*, 2021.

[Schmuck *et al.*, 2024] Anne-Kathrin Schmuck, Philippe Heim, Rayna Dimitrova, and Satya Prakash Nayak. Localized attractor computations for infinite-state games. In *CAV*, 2024.

[Skrynnik *et al.*, 2024] Alexey Skrynnik, Anton Andreychuk, Maria Nesterova, Konstantin Yakovlev, and Aleksandr Panov. Learn to follow: Decentralized lifelong multi-agent pathfinding via planning and learning. In *AAAI*, 2024.

[Sokota *et al.*, 2021] Samuel Sokota, Edward Lockhart, Finbarr Timbers, Elnaz Davoodi, Ryan D'Orazio, Neil Burch,

Martin Schmid, Michael Bowling, and Marc Lanctot. Solving common-payoff games with approximate policy iteration. In *AAAI*, 2021.

[Steeples *et al.*, 2021] Thomas Steeples, Julian Gutierrez, and Michael J. Wooldridge. Mean-payoff games with ω-regular specifications. In *AAMAS*, 2021.

[Unno *et al.*, 2023] Hiroshi Unno, Tachio Terauchi, Yu Gu, and Eric Koskinen. Modular primal-dual fixpoint logic solving for temporal verification. *Proc. ACM Program. Lang.*, 2023.

[Vorobeychik and Singh, 2012] Yevgeniy Vorobeychik and Satinder Singh. Computing stackelberg equilibria in discounted stochastic games. In *AAAI*, 2012.

[Xu *et al.*, 2022] Zhiwei Xu, Dapeng Li, Bin Zhang, Yuan Zhan, Yunpeng Bai, and Guoliang Fan. Mingling foresight with imagination: Model-based cooperative multi-agent reinforcement learning. In *NeurIPS*, 2022.

[Yu *et al.*, 2022] Xiaopeng Yu, Jiechuan Jiang, Wanpeng Zhang, Haobin Jiang, and Zongqing Lu. Model-based opponent modeling. *Advances in Neural Information Processing Systems*, 2022.

[Zhang *et al.*, 2021] Weinan Zhang, Xihuai Wang, Jian Shen, and Ming Zhou. Model-based multi-agent policy optimization with adaptive opponent-wise rollouts. In *IJCAI*, 2021.