Simple Stochastic Parity Games[⋆]

Krishnendu Chatterjee, Marcin Jurdziński, and Thomas A. Henzinger

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, USA

Abstract. Many verification, planning, and control problems can be modeled as games played on state-transition graphs by one or two players whose conflicting goals are to form a path in the graph satisfying their own objectives. The focus here is on simple stochastic parity games, that is, two-player games with turn-based probabilistic transitions and ω-regular objectives formalized as parity (Rabin chain) winning conditions. An efficient translation from simple stochastic parity games to nonstochastic parity games is given. As many algorithms are known for solving the latter, the translation yields efficient algorithms for computing the states of a simple stochastic parity game from which a player can win with probability 1.
An important special case of simple stochastic parity games are the Markov decision processes with Büchi objectives. For this special case a first provably subquadratic algorithm is given for computing the states from which the single player has a strategy to achieve a Büchi objective with probability 1. For game graphs with m edges the algorithm works in time $O(m\sqrt{m})$, compared with $O(mn)$ best algorithm known before. Interestingly, a similar technique sheds light on the important question of the computational complexity of solving simple Büchi games and yields the first provably subquadratic algorithm, with a running time of $O(n^2/\log n)$ for games with n vertices and $O(n)$ edges.

1 Introduction

Many verification, AI planning, and control problems can be formalized as state-transition graphs, and solved by finding paths in those graphs that meet certain criteria. Uncertainty about a process evolution is often modeled by probabilistic transitions, and then instead of searching for paths we are interested in measuring the probability that a path satisfies a given criterion, or finding controllers that maximize this probability. For decades there have been several separated communities studying such problems in the context of stochastic games [13], Markov decision processes (MDP's) [9], AI planning, and model checking. Only recently some unification has been attempted. MDP's can be naturally viewed as 1-player stochastic games and the book of Filar and Vrieze [8] provides a

⋆ This research was supported in part by the DARPA grant F33615-C-98-3614, the ONR grant N00014-02-1-0671, the NSF grants CCR-9988172 and CCR-0225610, and the Polish KBN grant 7-T11C-027-20.

unified rigorous treatment of the theories of MDP's and stochastic games. They coin the term Competitive Markov Decision Processes to encompass both 1- and 2-player stochastic games.

We suggest to cast various games based on state-transition models into a unified framework. For that purpose we use the following parameters:

– Number of players: "$1/2$": Markov chains; 1: nondeterministic state-transition systems; "$1\,1/2$": Markov decision processes; 2: game graphs; "$2\,1/2$": stochastic games.
– The players' knowledge about the course of the game: simple (or turn-based) games: the state determines who plays next; concurrent games: the players choose moves simultaneously and independently, without knowing each other's choices [13, 1, 5].
– Winning objectives: qualitative (ω-regular) objectives [14]: finite objectives (reachability and safety), or infinite objectives (liveness, such as Büchi or general parity conditions); quantitative (reward) objectives [8]: discounted reward, or limiting average reward, or total reward.
– Winning criteria: qualitative criteria [5]: sure winning, almost-sure winning (with probability 1), or limit-sure winning (with probability arbitrarily close to 1); quantitative criteria: exact probability of winning, or expected reward.

We mention a few notable examples of models and problems studied in various communities that fit into the above categorization.

– Summable MDP's [9, 8]: concurrent $1\,1/2$-player games with maximum expected discounted reward.
– Mean-payoff games [17]: simple 2-player games with maximum limiting average reward.
– Parity games [12, 14]: simple 2-player games with parity objectives.
– Quantitative simple stochastic games [2]: simple $2\,1/2$-player games with reachability objectives and exact probability of winning.
– Qualitative concurrent ω-regular games [4]: concurrent $2\,1/2$-player games with parity objectives and various qualitative winning criteria.
– Quantitative concurrent ω-regular games [6]: concurrent $2\,1/2$-player games with parity objectives and exact probability of winning.

In earlier work [4, 11] we have studied the complexity of algorithms for solving concurrent parity games. In particular, we have given efficient reductions from the problem of solving concurrent Büchi and co-Büchi games (under the almost-sure winning criterion) to the extensively studied problem of solving turn-based parity games [15, 10, 16]. In this paper we focus on the following three types of games:

– *Qualitative simple stochastic parity games*: simple $2\,1/2$-player games with parity objectives and almost-sure winning criterion.
– *Qualitative Büchi MDP's*: simple $1\,1/2$-player games with Büchi winning objectives and almost-sure winning criterion.
– *Simple Büchi games*: simple 2-player games with Büchi winning objectives.

We use n to denote the number of vertices and m to denote the number of edges of a parity game graph. Our main results can be summarized as follows.

Theorem 1. *Every qualitative simple stochastic parity game with priorities in the set* $\{0, 1, 2, \ldots, d-1\}$ *can be translated to a simple parity game with the same set of priorities, with* $O(dn)$ *vertices, and* $O(d(m+n))$ *edges, and hence it can be solved in time* $O(d(m+n) \cdot (nd)^{\lceil d/2 \rceil})$.

Theorem 2 (Pure memoryless determinacy). *From every vertex of a simple stochastic parity game either one player has a pure memoryless strategy to win with probability 1, or there is a* $\delta > 0$*, such that the other player has a pure memoryless strategy to win with probability at least* δ.

Corollary 1. *For simple stochastic parity games the almost-sure and limit-sure winning criteria coincide.*

Theorem 3. *Qualitative Büchi MDPs can be solved in time* $O(m\sqrt{m})$.

This implies also a complexity improvement for solving MDP's with reachability objectives under the almost-sure winning criterion, for which the best algorithm so far had $O(mn)$ running time [3]. Interestingly, the novel technique we use for Büchi MDP's allows us to shed some light on the important problem of finding subquadratic algorithms for simple Büchi games.

Theorem 4. *Simple Büchi games on graphs with* $O(n)$ *edges can be solved in time* $O(n^2/\log n)$.

This result and reductions in [11] prove that concurrent games with constant number of actions and with reachability and Büchi objectives under the almost-sure and limit-sure winning criteria can also be solved in subquadratic time (the best algorithms so far had $O(n^2)$ running time [4]).

2 Simple Stochastic Parity Games

Given $n \in \mathbb{N}$, we write $[n]$ for the set $\{0, 1, 2, \ldots, n\}$ and $[n]_+$ for the set $\{1, 2, \ldots, n\}$. A *2½-player game* (or *simple stochastic game*, or SSG) $G = (V, E, (V_\Box, V_\Diamond, V_\bigcirc))$ consists of a directed graph (V, E) and a partition $(V_\Box, V_\Diamond, V_\bigcirc)$ of the vertex set V. For technical convenience we assume that every vertex has at least one outgoing edge. For simplicity we only consider the case when G is binary. An *infinite path* in G is a infinite sequence $\langle v_0, v_1, v_2, \ldots \rangle$ of vertices such that $(v_k, v_{k+1}) \in E$ for all $k \in \mathbb{N}$. We write Ω for the set of all infinite paths. The game is played with three players that move a token from vertex to vertex so that an infinite path is formed: from vertices in $V_\Box$, player Even ($\Box$) moves the token along an outgoing edge to a successor vertex; from vertices in $V_\Diamond$, player Odd ($\Diamond$) moves the token; and from vertices in $V_\bigcirc$, player Random ($\bigcirc$) moves the token. If there are two outgoing edges, then player Random always moves the token to one of the two successor vertices with probability $1/2$. Since player

Random does not have a proper choice of moves, as the other two players do, we use the $1/2$-player terminology for player Random. The *2-player games* are the special case of the $2^1/_2$-player games with $V_{\bigcirc} = \emptyset$. The $1^1/_2$*-player games* (or MDP's) are the special case of the $2^1/_2$-player games with $V_{\Diamond} = \emptyset$. In other words, in 2-player games, the only players are Even and Odd; and in $1^1/_2$-player games, the only players are Even and Random.

Strategies. For a finite set A, a probability distribution on A is a function $f : A \to [0,1]$ such that $\sum_{a \in A} f(a) = 1$. We denote the set of probability distributions on A by $\mathcal{D}(A)$. A *mixed strategy* for player Even is a function $\sigma : V^* V_{\Box} \to \mathcal{D}(V)$, such that for a finite sequence $\overline{v} \in V^* V_{\Box}$ of vertices, representing the history of the play so far, $\sigma(\overline{v})$ is the next move to be chosen by player Even. A strategy must prescribe only available moves, i.e., $(v,u) \notin E$ then $\sigma(\overline{w} \cdot v)(u) = 0$ for all $\overline{w} \in V^*$ and $v \in V_{\Box}$. A strategy σ is *pure* if for all $\overline{w} \in V^*$ and $v \in V_{\Box}$, there is a vertex u such that $(v,u) \in E$ and $\sigma(\overline{w} \cdot v)(u) = 1$. The strategies for player Odd are defined analogously. We write $\varSigma$ and $\varPi$ for the sets of all strategies for players Even and Odd, respectively. A *memoryless strategy* is a strategy which does not depend on the history of the play but only on the current vertex. A pure memoryless strategy for player Even can be represented as a function $\sigma : V_{\Box} \to V$ such that $(v, \sigma(v)) \in E$ for all $v \in V_{\Box}$. For an initial vertex v, and two strategies $\sigma \in \varSigma$ and $\pi \in \varPi$ for players Even and Odd, respectively, we define $\mathit{Outcome}(v, \sigma, \pi) \subseteq \varOmega$ to be the set of paths that can be followed when a play starts from vertex v and the players use the strategies σ and π. Formally, $\langle v_0, v_1, v_2, \ldots \rangle \in \mathit{Outcome}(v, \sigma, \pi)$ if $v_0 = v$, and for all $k \geq 0$, we have that $v_k \in V_{\bigcirc}$ implies $(v_k, v_{k+1}) \in E$, $v_k \in V_{\Box}$ implies $\sigma(v_0, v_1, \ldots, v_k)(v_{k+1}) > 0$, and $v_k \in V_{\Diamond}$ implies $\pi(v_0, v_1, \ldots, v_k)(v_{k+1}) > 0$. Once a starting vertex v and strategies $\sigma \in \varSigma$ and $\pi \in \varPi$ for the two players have been chosen, the probabilities of events are uniquely defined, where an *event* $\mathcal{A} \subseteq \varOmega$ is a measurable set of paths. For a vertex v and an event $\mathcal{A} \subseteq \varOmega$, we write $\Pr_v^{\sigma,\pi}[\mathcal{A}]$ for the probability that a path belongs to $\mathcal{A}$ if the game starts from v and the players use the strategies σ and π.

Winning objectives. A *winning objective* for a SSG G is a set $\mathcal{W} \subseteq \varOmega$ of infinite paths. We consider the following winning objectives.

- *Büchi objective.* For a set $T \subseteq V$ of *target* vertices, the Büchi objective is defined as $\text{Büchi}(T) = \{\langle v_0, v_1, v_2 \ldots \rangle \in \varOmega : v_k \in T \text{ for infinitely many } k \geq 0\}$.
- *Parity objective.* Let $p : V \to [d]$ be a function that assigns a *priority* $p(v)$ to every vertex $v \in V$, where $d \in \mathbb{N}$. For an infinite path $\overline{v} = \langle v_0, v_1, \ldots \rangle \in \varOmega$, we define $\text{Inf}(\overline{v}) = \{\, i \in [d] : p(v_k) = i \text{ for infinitely many } k \geq 0 \,\}$. The *Even parity objective* is defined as $\text{Parity}(p) = \{\overline{v} \in \varOmega : \min\big(\text{Inf}(\overline{v})\big) \text{ is even}\}$, and the *Odd parity objective* as $\text{co-Parity}(p) = \{\, \overline{v} \in \varOmega \,:\, \min\big(\text{Inf}(\overline{v})\big) \text{ is odd} \,\}$.

Note that for a priority function $p : V \to [1]$ with only two priorities (0 and 1), an even parity objective $\text{Parity}(p)$ is equivalent to the Büchi objective $\text{Büchi}(p^{-1}(0))$, i.e., the target set consists of the vertices with priority 0. A $2^1/_2$*-player parity game* (or parity SSG) is a pair (G, p), where G is a $2^1/_2$-player game and p is a priority function. If G is a 2-player (resp. $1^1/_2$-player) game, then

(G, p) is a *2-player parity game* (resp. $1^1\!/_2$*-player parity game*, or parity MDP). A 2*-player Büchi game* is a pair (G, T), where G is a 2-player game and T is a set of target vertices. If G is a $1^1\!/_2$-player game, then (G, T) is a $1^1\!/_2$*-player Büchi game* (or Büchi MDP).

Winning criteria. Consider an SSG G with winning objective $\mathcal{W}$. We say that a strategy $\sigma \in \Sigma$ for player Even is

– *sure winning* from vertex v if for all strategies $\pi \in \Pi$ of player Odd, we have $Outcome(v, \sigma, \pi) \subseteq \mathcal{W}$;
– *almost-sure winning* from vertex v if for all strategies $\pi \in \Pi$ of player Odd, we have $\Pr_v^{\sigma,\pi}[\mathcal{W}] = 1$;
– *positive-probability winning* from vertex v if there is a $\delta > 0$, such that for all strategies $\pi \in \Pi$ of player Odd, we have $\Pr_v^{\sigma,\pi}[\mathcal{W}] \geq \delta$.

The definitions for player Odd are similar. We shall see that player Even has an almost-sure winning strategy for $\mathcal{W}$ from v if and only player Odd does not have a positive-probability winning strategy for $\Omega \setminus \mathcal{W}$ from v. For 2-player parity games all the three above winning criteria coincide, i.e., existence of a positive-probability winning strategy for a player implies existence of a sure winning strategy for him. In this paper we consider the dual criteria of almost-sure winning (i.e., winning with probability 1) and positive-probability winning, for $2^1\!/_2$- and $1^1\!/_2$-player games, and the criterion of sure winning for (nonstochastic) 2-player games:

– The problem of *solving a* $2^1\!/_2$*-player parity game* (resp. $1^1\!/_2$*-player game*) (G, p) is to compute the set of vertices of G from which the player Even has an almost-sure winnning strategy for the objective $\mathcal{W} = \text{Parity}(p)$.
– The problem of *solving a* 2*-player parity game* (G, p) is to compute the set of vertices of G from which the player Even has a sure winnning strategy for the objective $\mathcal{W} = \text{Parity}(p)$.

3 Solving $2^1\!/_2$-player Parity Games

The main result of this section is an algorithm for solving $2^1\!/_2$-player parity games, which is obtained by an efficient reduction to 2-player parity games, i.e., a proof of Theorem 1. As in our earlier work [11], the key technical tool for the correctness proof of the reduction is the notion of ranking functions, which witness the existence of winning strategies for the players. Our ranking functions are closely related to the semantics of the μ-calculus formulas that express the winning sets of *concurrent* stochastic parity games [4], but due to the lack of concurrency in our games, the defining conditions for our ranking functions are considerably simpler. Two corollaries of our proof are of independent interest. First, we establish the existence of pure memoryless winning strategies for both players in simple stochastic parity games (Theorem 2.) This is in contrast to concurrent games, where players need mixed strategies with infinite memory [4]. Second, in simple stochastic parity games the almost-sure and limit-sure winning criteria coincide (Corollary 1) which is not the case for concurrent games [4].

3.1 Ranking functions

In this subsection we provide a characterization of the "universal" parity MDP problem. We define certain sufficient conditions for establishing that for *all* strategies π, the Markov chain M_π satisfies the parity condition with probability 1, or with probability at least $\delta > 0$. These sufficient conditions are then used in the next subsection to prove correctness of our solution for $2^1\!/\!_2$-player parity games: a strategy σ is winning for a player if and only if the parity MDP G_σ is a solution to the universal parity MDP problem.

Consider a parity MDP $M = (V, E, (V_\Diamond, V_\bigcirc), p : V \to [d])$. Without loss of generality assume that d is even. A *ranking function* for player Even labels vertices with $(d/2)$-tuples of natural numbers: $\varphi = (\varphi^1, \varphi^3, \dots, \varphi^{d-1}) : V \to [n]^{d/2} \cup \{\infty\}$, for some $n \in \mathbb{N}$. For succinctness, for all odd $k \in [d]$ we write $\overrightarrow{\varphi}^k(v)$ to denote the tuple $(\varphi^1(v), \varphi^3(v), \dots, \varphi^k(v))$. We often call $\varphi(v)$ the *rank* of vertex v, and we call $\overrightarrow{\varphi}^k(v)$ the k-th rank of vertex v. A ranking function for player Odd is a function $\psi = (\psi^0, \psi^2, \dots, \psi^d) : V \to [n]^{d/2+1} \cup \{\infty\}$; we use similar notational conventions as with ranking functions for player Even. For all $v \in V_\bigcirc$, we write $\Pr_v[\overrightarrow{\varphi}^k_<]$ (resp. $\Pr_v[\overrightarrow{\varphi}^k_\leq]$) for the one-step probability of reaching from vertex v a successor u of v such that $\overrightarrow{\varphi}^k(u) <_{\text{lex}} \overrightarrow{\varphi}^k(v)$ (resp. $\overrightarrow{\varphi}^k(u) \leq_{\text{lex}} \overrightarrow{\varphi}^k(v)$). In other words, $\Pr_v[\overrightarrow{\varphi}^k_<]$ is the probability in vertex v of strictly decreasing the k-th rank in one step, and $\Pr_v[\overrightarrow{\varphi}^k_\leq]$ is the probability of not increasing the k-th rank. Moreover, we write $\Pr_v[\varphi_{<\infty}]$ (or for notational convenience $\Pr_v[\overrightarrow{\varphi}^{-1}_\leq]$) for the one-step probability of reaching from v a successor u of v such that $\varphi(u) \neq \infty$. We always use these notations in the context of expressions such as $\Pr_v[\overrightarrow{\varphi}^k_\leq] = 1$ or $\Pr_v[\overrightarrow{\varphi}^k_<] \geq \varepsilon$. By slight abuse of notation, for vertices $v \in V_\Box$ we also write $\Pr_v[\overrightarrow{\varphi}^k_<]$ and $\Pr_v[\overrightarrow{\varphi}^k_\leq]$ in such expressions, and then we mean those expressions to hold if and only if they hold for *all* mixed one-step strategies in vertex v. It is easy to verify that if either of the two expressions above holds for all pure one-step strategies in v, then it also holds for all mixed one-step strategies.

Definition 1 (Almost-sure ranking). *A ranking function $\varphi : V \to [n]^{d/2} \cup \{\infty\}$ for player Even is an almost-sure ranking if there is an $\varepsilon \geq 0$ such that for every vertex v with $\varphi(v) \neq \infty$, the following condition C_v holds:*

- *$p(v)$ even:* $\bigvee_{\text{odd } i \in [p(v)]} (\Pr_v[\overrightarrow{\varphi}^{i-2}_\leq] = 1 \wedge \Pr_v[\overrightarrow{\varphi}^i_<] \geq \varepsilon) \vee (\Pr_v[\overrightarrow{\varphi}^{p(v)-1}_\leq] = 1)$,
- *$p(v)$ odd:* $\bigvee_{\text{odd } i \in [p(v)]} (\Pr_v[\overrightarrow{\varphi}^{i-2}_\leq] = 1 \wedge \Pr_v[\overrightarrow{\varphi}^i_<] \geq \varepsilon)$.

Proposition 1. *Let $k \in [d]$ be an odd priority. Then for every vertex v with $\varphi(v) \neq \infty$ the following conditions hold.*

(a) If $p(v) = k$, then in one step from vertex v the k-th rank decreases with probability at least ε.

(b) If $p(v) > k$, then in one step from vertex v either the k-th rank decreases with probability at least ε, or the k-th rank does not increase (with probability 1).

Proof. If $p(v) = k$, then a disjunct $(\mathrm{Pr}_v[\overrightarrow{\varphi}^{k-2i-2}_{\leq}] = 1 \wedge \mathrm{Pr}_v[\overrightarrow{\varphi}^{k-2i}_{<}] \geq \varepsilon)$ of C_v must hold for some $i \geq 0$. From $\mathrm{Pr}_v[\overrightarrow{\varphi}^{k-2i}_{<}] \geq \varepsilon$ it follows immediately, however, that $\mathrm{Pr}_v[\overrightarrow{\varphi}^{k}_{<}] \geq \varepsilon$, which proves (a).

If $p(v) > k$, then either one of the above disjuncts holds and then the k-th rank decreases with probability at least ε, or $\mathrm{Pr}_v[\overrightarrow{\varphi}^{k+2i}_{\leq}] = 1$ holds, for some $i \geq 0$, which implies $\mathrm{Pr}_v[\overrightarrow{\varphi}^{k}_{\leq}] = 1$, i.e., the k-th rank does not increase, which concludes the proof of (b). ∎

Lemma 1. *Let φ be an almost-sure ranking for a parity MDP. Then for every (mixed) strategy of player Odd, the Even parity objective is satisfied with probability 1 from every vertex v with $\varphi(v) \neq \infty$.*

Proof. Once the strategy for player Odd is fixed, a play in the PMDP is an infinite random walk. We argue that this random walk satisfies the Even parity objective with probability 1. From our discussion above it follows that the conditions expressed in the definition of an almost-sure ranking hold for all mixed one-step strategies of player Odd, and since our reasoning below is carried out using only those conditions, it applies to all mixed strategies for player Odd.

In order to prove that with probability 1, the lowest priority occurring infinitely often is even, it suffices to show that for every odd priority $k \in [d]$, if vertices of priority k keep occurring in the random walk, then with probability 1 eventually a vertex of lower priority occurs. First note that from the definition of an almost-sure ranking, it follows that all successors of a vertex with finite rank have finite rank: one of the conditions $\mathrm{Pr}_v[\overrightarrow{\varphi}^{i}_{\leq}] = 1$ must hold so the i-th rank cannot increase in any step and thus the rank stays finite.

Let $k \in [d]$ be odd. For the sake of contradiction assume that from some point on vertices of priority k keep occurring, but no vertex of a lower priority ever occurs. In this case Proposition 1 implies that in every step either the k-th rank does not increase with probability 1, or it decreases with probability at least ε, and moreover, in every step from a vertex of priority k the k-th rank decreases with probability at least ε. As there are only $N = (n+1)^{(k+1)/2}$ different values of a k-th rank, within at most N visits to a vertex of priority k the k-th rank must decrease to $(0, \ldots, 0)$ with probability at least ε^N. Thus with probability 1 a vertex with k-th rank $(0, \ldots, 0)$ is eventually reached. This, however, contradicts the assumption that priority k occurs infinitely often, because no vertex of priority k can have its k-th rank equal to $(0, \ldots, 0)$, since by Proposition 1(a) a step from such a vertex has to decrease the k-th rank with positive probability and $(0, \ldots, 0)$ is the smallest existing rank. ∎

Definition 2 (Positive-probability ranking). *A ranking function $\psi : V \to [n]^{d/2+1} \cup \{\infty\}$ for player Odd is a positive-probability ranking if there is an $\varepsilon \geq 0$ such that for every vertex v with $\psi(v) \neq \infty$, the following condition D_v holds:*

- *$p(v)$ even: $(\mathrm{Pr}_v[\overrightarrow{\psi}^{0}_{<}] \geq \varepsilon) \vee \bigvee_{\text{even } i \in [p(v)]_+} (\mathrm{Pr}_v[\overrightarrow{\psi}^{i-2}_{\leq}] = 1 \wedge \mathrm{Pr}_v[\overrightarrow{\psi}^{i}_{<}] \geq \varepsilon)$,*
- *$p(v)$ odd: $(\mathrm{Pr}_v[\overrightarrow{\psi}^{0}_{<}] \geq \varepsilon) \vee \bigvee_{\text{even } i \in [p(v)]_+} (\mathrm{Pr}_v[\overrightarrow{\psi}^{i-2}_{\leq}] = 1 \wedge \mathrm{Pr}_v[\overrightarrow{\psi}^{i}_{<}] \geq \varepsilon) \vee (\mathrm{Pr}_v[\overrightarrow{\psi}^{p(v)}_{\leq}] = 1)$.*

A proof similar to that of Lemma 1 can be used to prove the following Lemma.

Lemma 2. *Let ψ be a positive-probability ranking for a parity MDP. Then there is a $\delta > 0$, such that for every (mixed) strategy of player Even, the Odd parity objective is satisfied with probability at least δ from every vertex v with $\psi(v) \neq \infty$.*

3.2 The reduction

Given a 2½-player parity game $G = (V, E, (V_\Box, V_\Diamond, V_\bigcirc), p : V \to [d])$, we construct a 2-player parity game $\overline{G}$ with the same set $[d]$ of priorities. For every vertex $v \in V_\Box \cup V_\Diamond$, there is a vertex $\overline{v} \in \overline{V}$ with "the same" outgoing edges, i.e., $(v, u) \in E$ if and only if $(\overline{v}, \overline{u}) \in \overline{E}$. Each random vertex $v \in V_\bigcirc$ is substituted by the gadget presented in Figure 3.2.

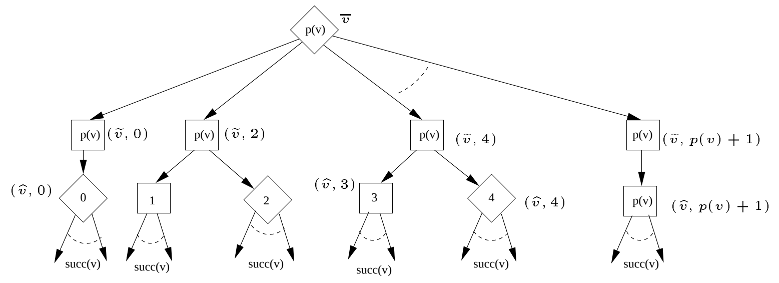

Fig. 1. Gadget for reducing 2½-player parity games to 2-player parity games.

More formally, the players play the following 3-step game in $\overline{G}$ from vertex $\overline{v}$ of priority $p(v)$. First, in vertex $\overline{v}$ player Odd chooses a successor $(\widetilde{v}, k)$, each of priority $p(v)$, where $k \in [p(v) + 1]$ is even. Then in vertex $(\widetilde{v}, k)$ player Even chooses from at most two successors: vertex $(\widehat{v}, k - 1)$ of priority $k - 1$ if $k > 0$, or vertex $(\widehat{v}, k)$ of priority k if $k \leq p(v)$. Finally, in a vertex $(\widehat{v}, k)$ the choice is between all vertices $\overline{u}$ such that $(v, u) \in E$, and it belongs to player Even if k is odd, and to player Odd if k is even.

Lemma 3 (Correctness of the reduction). *For every vertex v in G, if player Even (resp. Odd) has a sure winning strategy from vertex $\overline{v}$ in $\overline{G}$, then player Even (resp. Odd) has an almost-sure (resp. positive-probability) winning strategy from v.*

Proof. We prove the claim for player Even. The case of player Odd is similar and is omitted here. If $\overline{W_\Box}$ is the set of vertices from which player Even has a winning strategy in $\overline{G}$, then there is a ranking function (also called a *progress measure* [10]) $\overline{\varphi} : \overline{V} \to [n]^{d/2} \cup \{\infty\}$ (where $n \leq |\overline{V}|$) such that $\overline{\varphi}(w) \neq \infty$ for all $w \in \overline{W_\Box}$. This ranking function induces a memoryless winning strategy $\overline{\sigma}$ for player Even in $\overline{G}$ [10]. We define a ranking function φ and a memoryless strategy for player Even σ for G by setting $\varphi(v) = \overline{\varphi}(\overline{v})$ and $\sigma(v) = \overline{\sigma}(\overline{v})$ for every $v \in V$. Taking the strategy subgraph of σ in G we obtain a parity MDP M. In order to prove the claim, by Lemma 1 it suffices to argue that φ is an almost-sure ranking for M.

First we verify that the ranking condition C_v holds for all vertices $v \notin V_{\bigcirc}$ with $\varphi(v) \neq \infty$. Then by the definition of a ranking function [10], if (v, u) is an edge in M, then $\varphi^{p(v)}(v) >_{\text{lex}} \varphi^{p(v)}(u)$ if $p(v)$ is odd, and $\varphi^{p(v)-1}(v) \geq_{\text{lex}} \varphi^{p(v)-1}(u)$ if $p(v)$ is even. In the former case the disjunct $(\Pr_v[\overrightarrow{\varphi}^{p(v)-2}_{\leq}] = 1 \wedge \Pr_v[\overrightarrow{\varphi}^{p(v)}_{<}] \geq \varepsilon)$ holds, and in the latter the disjunct $(\Pr_v[\overrightarrow{\varphi}^{p(v)-1}_{\leq}] = 1)$ holds.

We prove that the ranking condition C_v holds for all vertices $v \in V_{\bigcirc}$ with $\varphi(v) \neq \infty$. Let $k \in [p(v)]_+$. Since vertex $\overline{v}$ belongs to player Odd, the edge leading to vertex $(\widetilde{v}, k)$ must be in M. Vertex $(\widetilde{v}, k)$ belongs to player Even, so either the edge leading to vertex $(\widehat{v}, k-1)$ or the one leading to vertex $(\widehat{v}, k)$ belongs to M. In the former case, by analyzing the inequalities between $\overline{\varphi}$-ranks of vertices on the path from $\overline{v}$ to the successor of $(\widehat{v}, k-1)$ in M which hold by the definition of a ranking function [10], we can deduce that $\Pr_v[\overrightarrow{\varphi}^{k-1}_{<}] \geq 1/2$ holds. In the latter case, we get that $\Pr_v[\overrightarrow{\varphi}^{k-1}_{\leq}] = 1$ holds. Considering all edges that lead out of vertex $\overline{v}$ in M, we conclude that the following condition holds: $(\Pr_v[\overrightarrow{\varphi}_{<\infty}] = 1) \wedge \bigwedge_{\text{odd } i \in [p(v)]} (\Pr_v[\overrightarrow{\varphi}^{i}_{\leq}] = 1 \vee \Pr_v[\overrightarrow{\varphi}^{i}_{<}] \geq 1/2)$. This condition can be shown to imply the ranking condition C_v using the two simple properties that if $i, j \in [p(v)]$ are odd and $i < j$, then $\Pr_v[\overrightarrow{\varphi}^{j}_{\leq}] = 1$ implies $\Pr_v[\overrightarrow{\varphi}^{i}_{\leq}] = 1$, and $\Pr_v[\overrightarrow{\varphi}^{i}_{<}] \geq \varepsilon$ implies $\Pr_v[\overrightarrow{\varphi}^{j}_{<}] \geq \varepsilon$. ∎

4 An $O(m\sqrt{m})$ Algorithm for $1^1\!/_2$-player Büchi Games

In this section we consider $1^1\!/_2$-player games with Büchi winning objectives, i.e., Büchi MDP's. There are two players, Even and Random. We write T for the set of target vertices, which player Even attempts to visit infinitely often. By $\mathcal{W}_{\Box}$ we denote the set of vertices from which player Even has an almost-sure winning strategy, and by $\mathcal{W}_{\bigcirc}$ the set from which the Büchi objective is violated with positive probability for all strategies of player Even. We call these sets the winning sets for player Even and Random, respectively. The main result of this section is an $O(n\sqrt{n})$ algorithm for computing $\mathcal{W}_{\Box}$ and $\mathcal{W}_{\bigcirc}$ for a $1^1\!/_2$-player Büchi game with n vertices and $O(n)$ edges. This proves Theorem 3 since a game graph with m edges can be easily converted in $O(m + n)$ time to an equivalent game graph with $O(m)$ vertices and $O(m)$ edges.

In the rest of the paper we use the following notations for a graph $G = (V, E)$ and a set $S \subseteq V$ of vertices. We write $\mathit{succ}(v, G) = \{ u \in V : (v, u) \in E \}$ for the set of immediate successors of vertex v. We define $\mathit{In}(S, G) = \{ (v, u) \in E : v \notin S \text{ and } u \in S \}$ to be the set of edges that enter set S and $\mathit{Source}(S, G) = \{ v \in V : (v, u) \in \mathit{In}(S, G) \text{ for some } u \}$ is the set of sources of edges that enter S. We write $\mathit{Reach}(S, G)$ for the set of vertices from which there is a path in graph G to a vertex in S. Let $(V_{\Box}, V_{*})$ be a partition of the set V of vertices of graph G (player $\Box$ moves from vertices in $V_{\Box}$ and player $*$ moves from vertices in V_{*}, where $* \in \{\Diamond, \bigcirc\}$). We inductively define the set $Attr_\Box(S, G)$ of vertices from which player $\Box$ has a strategy to reach set S in the following way. Set $R_0 = S$, and for $k \geq 0$, set: $R_{k+1} = R_k \cup \{\, v \in V_\Box \;:\; (v,u) \in E \text{ for some } u \in R_k \,\} \cup \{\, v \in V_* \;:\; u \in R_k \text{ for all } (v,u) \in E \,\}$. We set $Attr_\Box(S,G) = \bigcup_k R_k$. We define the set $Attr_*(S,G)$ in a similar way. We fix a graph G until the end of the paper and by a slight abuse of notation instead of putting graphs as the second parameters of all the above definitions we will write a subset of vertices of the graph G to stand for the subgraph of G induced by the set of vertices.

Algorithm 1 Classical algorithm for Büchi MDP's

```
Input : $1\frac{1}{2}$-player Büchi game $(G,T)$. Output: $W_\bigcirc$ and $W_\Box = V \setminus W_\bigcirc$.
1. $G_0 := G$; 2. $W_0 := \emptyset$; 3. $i := 0$
4. repeat
      4.1 $W_{i+1}$ := One-Iteration-Of-The-Classical-Algorithm$(V_i)$
      4.2 $V_{i+1} := V_i \setminus W_{i+1}$; $i := i+1$
   until $W_i = \emptyset$
5. $W_\bigcirc := \bigcup_{k=1}^{i} W_k$

Procedure One-Iteration-Of-The-Classical-Algorithm
Input: set $V_i \subseteq V$. Output: set $W_{i+1} \subseteq V_i$.
1. $R_i := Reach(T \cap V_i, V_i)$; 2. $Tr_i := V_i \setminus R_i$; 3. $W_{i+1} := Attr_\bigcirc(Tr, V_i)$
```

The *classical algorithm* (Algorithm 1 [3]) works as follows. First it finds the set of vertices R from which the target set T is reachable. The rest of the vertices $Tr = V \setminus R$ (a "trap" for player Even) are identified as winning for player Random. Then the set of vertices W, from which player Random has a strategy to reach its winning set Tr, is computed. Set W is identified as a subset of the winning set for player Random and it is removed from the vertex set. The algorithm then iterates on the reduced game graph. In every iteration it performs a *backward* search from the current target set to find vertices which can reach the target set. Each iteration takes $O(n)$ time.

Our *improved algorithm* (Algorithm 2) differs from the classical algorithm by selectively performing a backward search as the classical algorithm does, or a cheap *forward* exploration of edges from vertices that are good candidates to be included in the winning set of player Random. In Step 4.1 of the improved algorithm, if the number of edges entering the set of vertices included in the previous iteration into the winning set of player Random is at least as big as a constant k, then we run an iteration of the classical algorithm. Otherwise, i.e., if this number is smaller than k, then let S be the set of sources of edges that enter the set of vertices winning for player Random in the previous iteration. The vertices in S are considered as candidates to be included into the winning set of player Random in the current iteration. In Step 4.2.2.1 (procedure *Dovetail-Explore*) a dovetailing exploration of edges is performed from all vertices in S. From all vertices $v \in S$, up to ℓ edges are explored in a round-robin fashion.

If the forward exploration of edges from a vertex $v \in S$ terminates before ℓ edges are explored, and none of the explored vertices is in the target set, then the explored subgraph is included in the winning set of player Random. If Step 4.2.2.1 fails to identify a winning set of player Random, then in Step 4.2.4 an iteration of the classical algorithm is executed. The winning set discovered by the iteration of the classical algorithm in Step 4.2.4 must contain at least ℓ edges as otherwise it would have been discovered in Step 4.2.2.1.

Algorithm 2 Improved Algorithm for Büchi MDPs

```
Input: $1^1/_2$-player Büchi game $(G, T)$. Output: $W_\bigcirc$ and $W_\Box = V \setminus W_\bigcirc$.
1. $G_0 := G$; 2. $W_0 := \emptyset$; 3. $i := 0$
4. repeat
        4.1 if $|In(W_i, V_i \cup W_i)| \geq k$ then
            4.1.1 $W_{i+1} =:$ One-Iteration-Of-The-Classical-Algorithm$(V_i)$
        4.2 else $(|In(W_i, V_i \cup W_i)| < k)$
            4.2.1 $U := \emptyset$
            4.2.2 repeat
                    4.2.2.1 $Tr_i = $ Dovetail-Explore$(V_i \setminus U, Source(W_i \cup U, V_i \cup W_i))$
                    4.2.2.2 $U =: U \cup Attr_\bigcirc(Tr_i, V_i \setminus U)$
                until $(|In(W_i \cup U, V_i \cup W_i)| \leq k$ and $Tr_i \neq \emptyset)$
            4.2.3 if $Tr_i \neq \emptyset$ then $W_{i+1} := W_i \cup U$
            4.2.4 else $W_{i+1} := U \cup ($One-Iteration-Of-The-Classical-Algorithm$(V_i \setminus U))$
        4.3 $V_{i+1} := V_i \setminus W_{i+1}$
        4.4 $i := i + 1$
    until $W_i = \emptyset$
5. $W_\bigcirc := \bigcup_{k=1}^{i} W_k$

Procedure Dovetail-Explore
Input: set $S \subseteq V_i$ and graph $G_i = (V_i, E_i)$. Output: set $Tr_i \subseteq V_i$.
1. $Tr_i := \emptyset$; 2. $ec := \ell$
3. repeat
        3.1 $ec := ec - 1$
        3.2 for each vertex $v \in S$
            extend the sub-graph $R_v \subseteq V_i$ by exploring a new edge
        3.3 if there is $v \in S$, s.t. for all $u \in R_v$, $succ(u, V_i) \subseteq R_v$ and $R_v \cap T \cap V_i = \emptyset$
            then return $Tr_i := R_v$
    until $ec = 0$
4. return $Tr_i$
```

Lemma 4. *Algorithm 2 correctly computes the sets* $\mathcal{W}_\bigcirc$ *and* $\mathcal{W}_\Box$.

Proof. We prove by induction that W_i computed in any iteration of the improved algorithm satisfies $W_i \subseteq \mathcal{W}_\bigcirc$. Base case: $W_0 = \emptyset \subseteq \mathcal{W}_\bigcirc$.
Inductive case: we argue that $W_i \subseteq \mathcal{W}_\bigcirc$ implies $W_{i+1} \subseteq \mathcal{W}_\bigcirc$.

1. If Step 4.1 is executed, then $W_{i+1} \subseteq \mathcal{W}_\bigcirc$ by correctness of the classical algorithm.

2. If Step 4.2.2 is executed, then a nonempty set R_v is included in Tr_{i+1} in Step 3.3 of procedure *Dovetail-Explore*. By the condition in Step 3.3 of *Dovetail-Explore*, no vertex in R_v can reach a vertex outside of R_v, and since $R_v \cap T \cap V_i = \emptyset$, we conclude that T_i cannot be reached from any vertex in R_v. Therefore $R_v \subseteq \mathcal{W}_{\bigcirc}$ and $Tr_{i+1} \subseteq \mathcal{W}_{\bigcirc}$. Hence $U \subseteq \mathcal{W}_{\bigcirc}$ and $W_{i+1} \subseteq \mathcal{W}_{\bigcirc}$.
3. If Steps 4.2.2 and 4.2.4 are executed, then $U \subseteq \mathcal{W}_{\bigcirc}$ and the correctness of an iteration of the classical algorithm imply $W_{i+1} \subseteq \mathcal{W}_{\bigcirc}$.

The other inclusion $\mathcal{W}_{\bigcirc} \subseteq W_{\bigcirc}$ follows from the correctness of the classical algorithm, because the termination condition of the loop in Step 4 implies that $Tr_i = \emptyset$ holds so the last iteration was an iteration of the classical algorithm. ∎

Lemma 5. *The total work in Step 4.2.2.1 of Algorithm 2 is* $O(kn)$*.*

Proof. Consider the following two cases.

1. If a nonempty set of vertices R_v is included in the set Tr_i in Step 3.3 of *Dovetail-Explore*, and the number of edges in the induced subgraph R_v is e_i, then the total work done in Step 3 is $O(ke_i)$, because $|Source(W_i \cup U, V_i \cup W_i)| \leq k$. Since e_i edges are removed from the graph and the number of all edges in the graph is $O(n)$, the total work over all iterations of *Dovetail-Explore* when a nonempty set R_v is included in a set Tr_i is $O(kn)$.
2. If $Tr_i = \emptyset$ after executing the procedure *Dovetail-Explore*, then the work done there is $O(k\ell)$. Whenever this happens, the subgraph induced by the set of vertices Tr_i discovered by the following iteration of the classical algorithm must have more than ℓ edges. This can happen at most $O(n/\ell)$ times, because the number of edges in the graph is $O(n)$. Hence the total work over all iterations is $O((n/\ell)k\ell) = O(kn)$. ∎

Lemma 6. *The total work in Step 4.2.2.2 of Algorithm 2 is* $O(n)$ *and in Step 4.2.2 it is* $O(kn)$*.*

Lemma 7. *The total work in Step 4.1 of Algorithm 2 is* $O(n^2/k)$ *and in Step 4.2.4 it is* $O(n^2/\ell)$*.*

Lemma 8. *Algorithm 2 solves* $1^1\!/_2$*-player Büchi games with* n *vertices and* $O(n)$ *edges in time* $O(n\sqrt{n})$*.*

Proof. Correctness follows from Lemma 4. By Lemmas 6 and 7 the work in Steps 4.1, 4.2.2, and 4.2.4 is $O(n^2/k + kn + n^2/\ell)$. Take $k = \ell = \sqrt{n}$ to get the $O(n\sqrt{n})$ bound for the total work. ∎

5 An $O(n^2/\log n)$ Algorithm for 2-player Büchi Games

In this section we consider 2-player games of the form (G, T), where T for the set of target vertices. By $\mathcal{W}_{\Box}$ we denote the set of vertices from which player Even has a strategy to visit a state in T infinitely often, and by $\mathcal{W}_{\Diamond}$ the set of vertices from which player Odd can avoid visiting T infinitely often. These are the winning sets for player Even and Odd, respectively. By determinacy of parity games [7] we have $\mathcal{W}_\Diamond = V \setminus \mathcal{W}_\Box$. Inspired by the algorithm of the previous section, we provide an algorithm for computing the set $\mathcal{W}_\Diamond$ in time $O(n^2/\log n)$ if G has n vertices and $O(n)$ edges, a proof of Theorem 4. A graph is *binary* if every vertex has at most two successors. For simplicity we present the algorithm for the case when the game graph is binary. (Observe that a game graph with $O(n)$ edges can be easily converted into an equivalent binary game graph with $O(n)$ vertices and edges.)

The *classical algorithm* for solving 2-player Büchi games (Algorithm 3 [15]) is very similar to the classical algorithm of the previous section. The only difference is that in step 1 of each iteration i we compute set R_i to be the set of vertices from which player Even has a strategy to reach set T, i.e., $R_i = \mathit{Attr}_\Box(T, V_i)$; and in step 3 the set W_{i+1} is the set of vertices from which player Odd has a strategy to reach set Tr_i. Then the winning set of player Even is obtained as the union of the sets W_i over all iterations.

Note that in step 1 of every iteration i an $O(n)$ *backward* alternating search is performed to compute the set R_i. The key idea of our *improved algorithm* (Algorithm 4) is to perform a cheap *forward* exploration of edges in some iterations in order to discover subsets of the winning set for player Odd. The improved algorithm for 2-player Büchi games differs from the improved algorithm of the previous section in the way the forward exploration is performed. In order to detect a trap for player Even in which player Odd has a winning strategy, we need to consider all successors of every vertex in the forward exploration. Let S be the set of sources of edges entering the winning set of player Odd discovered in the previous iteration, and let $|S| \leq k$. The vertices in set S are new candidates to be included in the winning set of player Odd. From these vertices a BFS of depth $\log \ell$ is performed in Step 4.2.2.1 of Algorithm 4. In step 4.2.2.4 we check if the explored subgraph contains a trap for player Even in which player Odd has a winning strategy. If no such trap is detected then one iteration of the classical algorithm is executed. The key for the subquadratic bound of our algorithm is the observation that if step 4.2.2 fails to identify a non-empty winning subset for player Odd, then the set discovered by the following iteration of the classical algorithm has at least $\log \ell$ vertices.

Algorithm 3 Classical Algorithm for 2-player Büchi Games

```
    **Input :** 2-player Büchi game $(G, T)$. **Output:** $W_\Diamond$ and $W_\Box = V \setminus W_\Diamond$.
    [Steps 1.–4. are the same as in Algorithm 1]
    5. $W_\Diamond := \bigcup_{k=1}^{i} W_k$

**Procedure** *One-Iteration-Of-The-Classical-Algorithm*
    **Input:** set $V_i \subseteq V$. **Output:** set $W_{i+1} \subseteq V_i$.
    1. $R_i := \mathit{Attr}_\Box(T \cap V_i, V_i)$; 2. $\mathit{Tr}_i := V_i \setminus R_i$; 3. $W_{i+1} := \mathit{Attr}_\Diamond(\mathit{Tr}_i, V_i)$
```

Algorithm 4 **Improved Algorithm for 2-player Büchi Games**

```
Input : 2-player Büchi game $(G, T)$. Output: $W_\Diamond$ and $W_\Box = V \setminus W_\Diamond$.
[Steps 1.–3. and 4.1 are the same as in Algorithm 2]
4. repeat
        4.2 else $(|In(W_i, V_i \cup W_i)| < k)$
            4.2.1 $Tr_i := \emptyset$
            4.2.2 for each vertex $v \in Source(W_i, V_i \cup W_i)$
                4.2.2.1 Find the reachable subgraph $R_v$ by a BFS of depth $\log \ell$
                4.2.2.2 Let $F_v$ denote the set of vertices at depth $\log \ell$
                4.2.2.3 $T'_v := \{v \in V_\Diamond \cap F_v : succ(v, G_i) \cap R_v = \emptyset\} \cup (V_\Box \cap F_v)$
                4.2.2.4 $R'_v := Attr_\Box((R_v \cap T \cap V_i) \cup T'_v, R_v)$
                4.2.2.5 $Tr_i := Tr_i \cup (R_v \setminus R'_v)$
            4.2.3 if $Tr_i \neq \emptyset$ then $W_{i+1} := Attr_\Diamond(Tr_i, V_i)$
            4.2.4 else $W_{i+1} :=$ One-Iteration-Of-The-Classical-Algorithm$(V_i)$
    until $W_i = \emptyset$
5. $W_\Diamond := \bigcup_{k=1}^{i} W_k$
```

We say that a set of vertices S is an *Even trap* in a graph G if for all $v \in S$, we have $succ(v, G) \subseteq S$ if $v \in V_\Box$, and $succ(v, G) \cap S \neq \emptyset$ if $v \in V_\Diamond$. It is easy to verify that if $P \subseteq V$ then the set $V \setminus Attr_\Box(P, V)$, i.e., the complement of an *Even attractor*, is always an Even trap in the graph induced by vertices in set V. Intuitively, player Odd can prevent player Even from leaving an Even trap, and hence if $S \cap T = \emptyset$ then a trap is included in the winning set of player Odd in the Büchi game (G, T); we call such a set an *Even trap winning for player Odd.*

Lemma 9. *Algorithm 4 correctly computes the sets $\mathcal{W}_\Diamond$ and $\mathcal{W}_\Box$.*

Proof. We prove by induction that W_i computed in any iteration of the improved algorithm satisfies $W_i \subseteq \mathcal{W}_\Diamond$. Base case: $W_0 = \emptyset \subseteq \mathcal{W}_\Diamond$.
Inductive case: we argue that $W_i \subseteq \mathcal{W}_\Diamond$ implies $W_{i+1} \subseteq \mathcal{W}_\Diamond$. We consider three cases as in the proof of Lemma 4.

1. Cases 1 and 3 are similar to those in the proof of Lemma 4.
2. We argue that if Steps 4.2 and 4.2.3 get executed in iteration i then every nonempty set $R_v \setminus R'_v$ included into set Tr_{i+1} is an Even trap winning for player Odd in the subgraph of G induced by the set of vertices V_i. It is an Even trap because for every vertex $u \in V_\Box \cap (R_v \setminus F_v)$, we have $succ(u, V_i) \subseteq R_v$, and as a complement of an Even attractor it is an Even trap in the subgraph induced by set R_v. The set $R_v \setminus R'_v$ is moreover an Even trap *winning for player Odd* since by step 4.2.2.4 all target vertices in set R_v are included in the set R'_v.

The rest of the argument is similar to the proof of Lemma 4. ∎

Lemma 10. *Let R_v be a set computed in Step 4.2.2.1. Then every Even trap S winning for player Odd whose all vertices are reachable from vertex v, and such that $v \in S$, and $|S| < \log \ell$, is contained in $R_v \setminus R'_v$, and hence it is discovered in Step 4.2.2.*

Lemma 11. *The total work in Step 4.1 of Algorithm 4 is $O(n^2/k)$, in Step 4.2.2 it is $O(k\ell n)$, and in Step 4.2.3 it is $O(n)$.*

Lemma 12. *Algorithm 4 solves 2-player Büchi games with n vertices and $O(n)$ edges in time $O(n^2/\log n)$.*

Proof. Correctness follows from Lemma 9. The work of Step 4.2.4 is $O(n^2/\log \ell)$ by Lemma 10. The work of Steps 4.1, 4.2.2, and 4.2.4 is $O(n^2/k + k\ell n + n^2/\log \ell)$ by Lemma 11. Take $\ell = n^\varepsilon$ with $0 < \varepsilon < 1$ and $k = \log n$ to get the $O(n^2/\log n)$ upper bound for the total work. By Lemma 11 the work in Step 4.2.3 is $O(n)$, hence the time complexity of Algorithm 4 is $O(n^2/\log n)$. ∎

References

1. R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49:672–713, 2002.
2. A. Condon. The complexity of stochastic games. *Information and Computation*, 96:203–224, 1992.
3. L. de Alfaro. Computing minimum and maximum reachability times in probabilistic systems. In *CONCUR'99*, volume 1664 of *LNCS*, pages 66–81. Springer, 1999.
4. L. de Alfaro and T. A. Henzinger. Concurrent ω-regular games. In *LICS'00*, pages 141–154. IEEE Computer Society Press, 2000.
5. L. de Alfaro, T. A. Henzinger, and O. Kupferman. Concurrent reachability games. In *FOCS'98*, pages 564–575, 1998.
6. L. de Alfaro and R. Majumdar. Quantitative solution of omega-regular games. In *STOC'01*, pages 675–683. ACM Press, 2001.
7. E. A. Emerson and C. Jutla. Tree automata, μ-calculus and determinacy. In *FOCS'91*, pages 368–377. IEEE Computer Society Press, 1991.
8. J. Filar and K. Vrieze. *Competitive Markov Decision Processes.* Springer, 1997.
9. R. A. Howard. *Dynamic Programming and Markov Processes.* Wiley, 1960.
10. M. Jurdziński. Small progress measures for solving parity games. In *STACS'00*, volume 1770 of *LNCS*, pages 290–301. Springer, 2000.
11. M. Jurdziński, O. Kupferman, and T. A. Henzinger. Trading probability for fairness. In *CSL'02*, volume 2471 of *LNCS*, pages 292–305. Springer, 2002.
12. Robert McNaughton. Infinite games played on finite graphs. *Annals of Pure and Applied Logic*, 65(2):149–184, 1993.
13. L. S. Shapley. Stochastic games. *Proceedings Nat. Acad. of Science USA*, 39:1095–1100, 1953.
14. W. Thomas. On the synthesis of strategies in infinite games. In *STACS'95*, volume 900 of *LNCS*, pages 1–13. Springer, 1995.
15. W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, Beyond Words, chapter 7, pages 389–455. Springer, 1997.
16. J. Vöge and M. Jurdziński. A discrete strategy improvement algorithm for solving parity games. In *CAV'00*, volume 1855 of *LNCS*, pages 202–215. Springer, 2000.
17. U. Zwick and M. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158:343–359, 1996.