BRICS

Basic Research in Computer Science

BRICS RS-00-48 Jurdziński & Vöge: A Discrete Stratety Improvement Algorithm for Solving Parity Games

A Discrete Stratety Improvement Algorithm for Solving Parity Games

Marcin Jurdziński
Jens Vöge

BRICS Report Series **RS-00-48**

ISSN 0909-0878 **December 2000**

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark
Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/00/48/`

A Discrete Strategy Improvement Algorithm for Solving Parity Games*

Marcin Jurdziński†
BRICS‡
Dept. of Computer Sci.
University of Aarhus
Denmark

Jens Vöge§
Lehrstuhl für Informatik VII
Fachgruppe Informatik
RWTH Aachen
Germany

December 2000

Abstract

A discrete strategy improvement algorithm is given for constructing winning strategies in parity games, thereby providing also a new solution of the model-checking problem for the modal μ-calculus. Known strategy improvement algorithms, as proposed for stochastic games by Hoffman and Karp in 1966, and for discounted payoff games and parity games by Puri in 1995, work with real numbers and require solving linear programming instances involving high precision arithmetic. In the present algorithm for parity games these difficulties are avoided by the use of discrete

*A preliminary version of this paper [21] was published in: *Computer Aided Verification, 12th International Conference, CAV 2000, Proceedings*, volume 1855 of *Lecture Notes in Computer Science*, pages 202–215, Chicago, IL, USA, July 2000, Springer-Verlag.

†Address: BRICS, Department of Computer Science, University of Aarhus, Ny Munkegade Building 540, DK-8000 Aarhus C, Denmark. Email: `mju@brics.dk`. A part of the work reported here was done while the author was a research assistant at Lehrstuhl für Informatik VII, RWTH Aachen, Germany.

‡Basic Research in Computer Science, Centre of Danish Science Foundation.

§Address: RWTH Aachen, Lehrstuhl für Informatik VII, D-52056 Aachen, Germany. Email: `voege@informatik.rwth-aachen.de`.

Supported by the Deutsche Forschungsgemeinschaft (DFG), project Th 352/5-3.

vertex valuations in which information about the relevance of vertices and certain distances is coded. An efficient implementation is given for a strategy improvement step. Another advantage of the present approach is that it provides a better conceptual understanding and easier analysis of strategy improvement algorithms for parity games. However, so far it is not known whether the present algorithm works in polynomial time. The long standing problem whether parity games can be solved in polynomial time remains open.

1 Introduction

The study of the computational complexity of solving parity games has two main motivations. One is that the problem is polynomial time equivalent to the modal μ-calculus model checking [8, 19], and hence better algorithms for parity games may lead to better model checkers, which is a major objective in computer aided verification.

The other motivation is the intriguing status of the problem from the point of view of structural complexity theory. It is one of the few natural problems which is in NP $\cap$ co-NP [8] (and even in UP $\cap$ co-UP [10]), and is not known to have a polynomial time algorithm, despite substantial effort of the community (see [8, 1, 18, 11] and references there). Other notable examples of such problems include simple stochastic games [3, 4], mean payoff games [6, 23], and discounted payoff games [23]. There are polynomial time reductions of parity to mean payoff games [16, 10], mean payoff to discounted payoff games [23], and discounted payoff to simple stochastic games [23]. Parity games, as the simplest of them all, seem to be the most plausible candidate for trying to find a polynomial time algorithm.

A strategy improvement algorithm has been proposed for solving stochastic games by Hoffman and Karp [9] in 1966. Puri in his PhD thesis [16] has adapted the algorithm for discounted payoff games. Puri also provided a polynomial time reduction of parity games to mean payoff games, and advocated the use of the algorithm for solving parity games, and hence for the modal μ-calculus model checking.

In our opinion Puri's strategy improvement algorithm for solving parity games has two drawbacks.

- The algorithm uses high precision arithmetic, and needs to solve linear programming instances: both are typically costly operations.

An implementation (by the first author) of Puri's algorithm, using a linear programming algorithm of Meggido [13], proved to be prohibitively slow.

- Solving parity games is a discrete, graph theoretic problem, but the crux of the algorithm is manipulation of real numbers, and its analysis is crucially based on continuous methods, such as Banach's fixed point theorem.

The first one makes the algorithm inefficient in practice, the other one obscures understanding of the algorithm.

Our discrete strategy improvement algorithm remedies both shortcomings of Puri's algorithm mentioned above, while preserving the overall structure of the generic strategy improvement algorithm. We introduce discrete values (such as tuples of: vertices, sets of vertices and natural numbers denoting lengths of paths in the game graph) which are being manipulated by the algorithm, instead of their encodings into real numbers. (One can show a precise relationship between behaviour of Puri's and our algorithms; we will treat this issue elsewhere.)

The first advantage of our approach is that we avoid solving linear programming instances involving high precision arithmetic. Instead, a shortest paths instance needs to be solved in every strategy improvement step of the algorithm. The shortest paths instances occurring in this context have discrete weights recording relevance of vertices and distances in the game graph. We develop an algorithm exploiting the special structure of these instances instead of using standard shortest paths algorithms. Our algorithm gives an efficient implementation of a single improvement step of the strategy improvement algorithm. Its running time is $O(n \cdot m)$, where n is the number of vertices, and m is the number of edges in the game graph. In comparison, a naive application of Bellman-Ford algorithm [5] gives $O(n^2 \cdot m)$ running time.

The other advantage is more subjective: we believe that it is easier to analyze the discrete data maintained by our algorithm, rather than its subtle encodings into real numbers involving infinite geometric series [16]. The classical continuous reasoning involving Banach fixed point theorem gives an elegant proof of correctness of the algorithm in a more general case of discounted payoff games [16], but we think that in the case of parity games it blurs an intuitive understanding of the underlying discrete structure. However, the long standing open question whether a strategy improvement algorithm works in polynomial time [4] remains unanswered. Nevertheless, we hope that our discrete analysis of the algorithm may help either to find a proof of polynomial time termination, or to come up with a family of examples on which the algorithm requires exponential number of steps. Any of those results would mark a substantial progress in understanding the computational complexity of parity games.

So far, for all families of examples we have considered the strategy improvement algorithm needs only linear number of strategy improvement steps. Notably, a linear number of strategy improvements suffices for several families of difficult examples for which other known algorithms need exponential time.

The rest of this chapter is organized as follows. In section 2 we define the infinite parity games and we establish their equivalence to finite cycle-domination games. In section 3 we sketch the idea of a generic strategy improvement algorithm and we state general postulates which guarantee correctness of the algorithm. Then in section 4 we give a specific proposal for the ingredients of a generic strategy improvement algorithm. In section 5 we prove that these ingredients satisfy the postulates of section 3. In this way we get a purely discrete strategy improvement algorithm for solving parity games. In section 6 we give a specialized shortest paths algorithm for the shortest paths instances occurring in our strategy improvement algorithm. In this way we obtain an efficient $O(n \cdot m)$ implementation of a strategy improvement step, where n is the number of vertices, and m is the number of edges in the game graph. Finally, in section 7 we discuss partial results and open questions concerning the time complexity of strategy improvement algorithms.

2 Parity games

2.1 Infinite parity games

A *game graph* $G = \big(V, E, (M_{\text{Even}}, M_{\text{Odd}}), p\big)$ of a parity game consists of a directed graph (V, E), a partition $(M_{\text{Even}}, M_{\text{Odd}})$ of the set of vertices V, and a *priority function* $p : V \to \{0, 1, \ldots, k\}$ for some $k \in \mathbb{N}$. We restrict ourselves to finite parity game graphs and for technical convenience we assume that every vertex has at least one out-going edge.

An *infinite parity game* G^∞ is a two-person infinite duration path-forming game played on graph G. The two players, Even and Odd, keep moving a token along edges of the game graph: player Even moves the token from vertices in M_{Even}, and player Odd moves the token from vertices in M_{Odd}. A *play* of G^∞ is an infinite path $\langle v_0, v_1, v_2, \ldots \rangle$ in the game graph G arising in this way. A play $P = \langle v_0, v_1, v_2, \ldots \rangle$ is *winning* for player Even if the biggest priority occurring infinitely often in P (i.e., the biggest number occurring infinitely often in $\langle p(v_0), p(v_1), p(v_2), \ldots \rangle$) is even; otherwise P is winning for player Odd.

A *memoryless strategy* for player Even is a function $\sigma : M_{\mathrm{Even}} \to V$ such that $\big(v, \sigma(v)\big) \in E$ for all $v \in M_{\mathrm{Even}}$. (We consider only memoryless strategies here so for brevity we just write strategies to denote memoryless strategies throughout the paper.) A play $\langle v_0, v_1, v_2, \ldots \rangle$ is *consistent* with a strategy σ for player Even if $v_{\ell+1} = \sigma(v_\ell)$ for all $\ell \in \mathbb{N}$, such that $v_\ell \in M_{\mathrm{Even}}$. Strategies for player Odd are defined analogously. If σ is a strategy for player Even (Odd) then we write G_σ to denote the game graph obtained from G by removing all the edges (v, w), such that $v \in M_{\mathrm{Even}}$ ($v \in M_{\mathrm{Odd}}$) and $\sigma(v) \neq w$; we write E_σ for the set of edges of G_σ. If σ is a strategy for player Even and τ is a strategy for player Odd then we write $G_{\sigma\tau}$ for $(G_\sigma)_\tau$; we write $E_{\sigma\tau}$ for its set of edges.

Note that if σ is a strategy for player Even and τ is a strategy for player Odd then for every vertex v, there is a unique play $P_{\sigma\tau}(v)$ starting from v and consistent with both σ and τ. We say that a strategy σ for player Even is a *winning strategy* from a vertex v if for every strategy τ for player Odd, the unique play $P_{\sigma\tau}(v)$ starting from v and consistent with both σ and τ is winning for player Even. A strategy σ is a winning strategy from a set of vertices W if it is winning from every vertex in W. Winning strategies for player Odd are defined analogously.

Remark. In literature on parity games (see for example [20, 22, 10]) a different definition of a winning strategy is used; here we call it "a strategy winning against arbitrary strategies." We say that a strategy σ for player Even (Odd) is a *strategy winning against arbitrary strategies* from a vertex v if every play starting from v and consistent with σ is winning for player Even (Odd). We argue that the two definitions are equivalent for parity games.

Proposition 2.1 A strategy σ for player Even (Odd) is a winning strategy from a vertex v if and only if σ is a strategy winning against arbitrary strategies from v.

Proof. The "if" part is obvious.

We prove the "only if" part for player Even; the proof for player Odd is analogous. Suppose that σ is a winning strategy for player Even from

v. Then the biggest priority on every cycle in G_σ reachable from v in G_σ is even. Let P be an infinite play consistent with σ. We argue that the biggest priority occurring infinitely often on P is even. Using a simple stacking technique (see for example [23], page 347, or [10], page 122) we can decompose play P into a finite path and an infinite number of simple cycles in G_σ. (The decomposition may be different if the game graph is infinite, but the proposition can be proved in a very similar way also for parity games with infinite game graphs.) Note that biggest priorities in all simple cycles in the decomposition are even because all the cycles are cycles in G_σ reachable from v in G_σ. Hence the maximal priority occurring infinitely often in P is even, i.e., P is a winning play for player Even. [Proposition 2.1] ■ [Remark] □

Theorem 2.2 (Memoryless Determinacy [7, 15])
For every parity game graph G, there is a unique partition $(W_{\text{Even}}, W_{\text{Odd}})$ of the set of vertices of G, such that there is a winning strategy for player Even from W_{Even} in G^∞ and a winning strategy for player Odd from W_{Odd} in G^∞.

We call sets W_{Even} and W_{Odd} the *winning sets* of player Even and player Odd, respectively. The problem of *deciding the winner in parity games* is, given a parity game graph and a vertex of the graph, to determine whose winning set the vertex belongs to. By *solving* a parity game we mean finding the partition $(W_{\text{Even}}, W_{\text{Odd}})$.

2.2 Finite cycle-domination games

For the purpose of development and reasoning about our discrete strategy improvement algorithm for solving parity games it is technically convenient to reformulate a bit the standard definition of parity games outlined above. Below we define finite cycle-domination games and we give a simple translation of every parity game into an equivalent cycle-domination game.

A game graph $G = \big(V, E, (M_\oplus, M_\ominus), (R_+, R_-)\big)$ of a cycle-domination game consists of a directed graph (V, E), where $V = \{1, 2, \ldots, n\}$ for some $n \in \mathbb{N}$, and of two partitions (R_+, R_-) and $(M_\oplus, M_\ominus)$ of the set of vertices V. We sometimes refer to the numbers identifying vertices as their *priorities*, and we use the standard $\leq$ order on natural numbers to compare them.

A *finite cycle-domination game* G^ω is a two-person finite path-forming game played on graph G. The two players, player $\oplus$ and player $\ominus$, play similarly to players Even and Odd in infinite parity games by moving a token along edges of the game graph G: player $\oplus$ moves the token from vertices in $M_\oplus$, and player $\ominus$ moves the token from vertices in $M_\ominus$. The difference from infinite parity games is that a play is finished as soon as the path constructed so far by the players contains a cycle. In other words, a play in G^ω is a finite path $P = \langle v_0, v_1, \ldots, v_\ell \rangle$ in the game graph G, such that $v_i \neq v_j$ for all $1 \le i < j < \ell$, and there is a $k < \ell$, such that $v_k = v_\ell$. It follows that $\langle v_k, v_{k+1}, \ldots, v_\ell \rangle$ is the only cycle in play P; we write $\texttt{Cycle}(P)$ to denote this unique cycle, and we call it the cycle of P. We define the *cycle dominating value* $\lambda(P)$ of play P to be $\max^{\le}\big(\texttt{Cycle}(P)\big) = \max^{\le}\{v_k, v_{k+1}, \ldots, v_\ell\}$, i.e., the vertex with biggest priority in the cycle of P. We say that a play P in G^ω is winning for player $\oplus$ if $\lambda(P) \in R_+$, otherwise $\lambda(P) \in R_-$ and play P is winning for player $\ominus$. Strategies and winning strategies for both players in G^ω are defined similarly as in infinite parity games.

Given a parity game graph $G = \big(V, E, p, (M_{\mathrm{Even}}, M_{\mathrm{Odd}})\big)$ we construct a cycle-domination game graph $\overline{G} = \big(V, E, (R_+, R_-), (M_\oplus, M_\ominus)\big)$, such that the games G^∞ and $\overline{G}^\omega$ are equivalent in the following sense.

Proposition 2.3 A strategy σ is a winning strategy for player Even (player Odd) from a vertex v in G^∞ if and only if σ is a winning strategy for player $\oplus$ (player $\ominus$) from v in $\overline{G}^\omega$.

Construction of $\overline{G}$: Let $M_\oplus = M_{\mathrm{Even}}$, $M_\ominus = M_{\mathrm{Odd}}$, and

$$R_+ = \big\{\, v \in V \ :\ p(v) \text{ is even} \,\big\}, \text{ and } R_- = \big\{\, v \in V \ :\ p(v) \text{ is odd} \,\big\}.$$

We introduce a total order relation $\le$ on V called the *relevance* order. Let the relevance order $\le$ be an arbitrary total order extending the pre-order induced by priorities, i.e., such that $p(v) \le p(w)$ implies $v \le w$ for all $v, w \in V$. Note that we use the same symbol "$\le$" for the standard order on natural numbers and for the relevance order; in fact we identify vertices in V with integers in the set $\big\{1, 2, \ldots, |V|\big\}$ via the unique order-isomorphism between the total orders $(V, \le)$ and $\big(\big\{1, 2, \ldots, |V|\big\}, \le\big)$.

Proof (of Proposition 2.3). The first condition, i.e., that σ is a winning strategy for player Even (player Odd) in G^∞ is equivalent to saying that for every cycle in G_σ reachable from v in G_σ, the biggest priority occurring on the cycle is even (odd). The other condition, i.e., that σ is a winning strategy for player $\oplus$ (player $\ominus$) in $\overline{G}^\omega$ is equivalent to saying that for every cycle in $\overline{G}_\sigma$, the most relevant vertex in the cycle belongs to R_+ (R_-). It follows immediately from the construction of $\overline{G}$ that the biggest priority occurring on a cycle in G_σ is even (odd), if and only if the most relevant vertex in the same cycle in $\overline{G}_\sigma$ belongs to R_+ (R_-), and so we are done. [Proposition 2.3] ■

3 Generic strategy improvement algorithm

In order to develop a strategy improvement algorithm for solving cycle-domination games we define the problem of solving cycle-domination games as an "optimization problem." Suppose we have a pre-order $\sqsubseteq$ on the set $\texttt{Strategies}_\oplus \subseteq (M_\oplus \to V)$ of strategies for player $\oplus$, satisfying the following two postulates.

P1. There is a maximum element in the pre-order $(\texttt{Strategies}_\oplus, \sqsubseteq)$, i.e., there is a strategy $\sigma \in \texttt{Strategies}_\oplus$, such that $\kappa \sqsubseteq \sigma$ for all $\kappa \in \texttt{Strategies}_\oplus$.

P2. If σ is a maximum element in the pre-order $(\texttt{Strategies}_\oplus, \sqsubseteq)$ then σ is a winning strategy for player $\oplus$ from every vertex of her winning set.

The problem of *solving a cycle-domination game with respect to* $\sqsubseteq$ is, given a cycle-domination game, to find a strategy for player $\oplus$ which is a maximum element in the pre-order $(\texttt{Strategies}_\oplus, \sqsubseteq)$.

Suppose we also have an operator

$$\texttt{Improve} : \texttt{Strategies}_\oplus \to \texttt{Strategies}_\oplus,$$

satisfying the following two postulates:

I1. If σ is not a maximum element in the pre-order $(\texttt{Strategies}_\oplus, \sqsubseteq)$ then $\sigma \sqsubseteq \texttt{Improve}(\sigma)$ and $\texttt{Improve}(\sigma) \not\sqsubseteq \sigma$.

I2. If σ is a maximum element in the pre-order $(\texttt{Strategies}_\oplus, \sqsubseteq)$ then we have $\texttt{Improve}(\sigma) = \sigma$.

A generic strategy improvement algorithm is the following procedure.

Strategy improvement algorithm
$\quad\texttt{pick a strategy}\ \sigma\ \texttt{for player}\ \oplus$
$\quad$**while** $\sigma \neq \texttt{Improve}(\sigma)$ **do** $\sigma := \texttt{Improve}(\sigma)$

Note that postulate I1. guarantees that the algorithm terminates, because there are only finitely many strategies for player $\oplus$. Postulate I2. implies that when the algorithm terminates then the strategy σ is a maximum element in the pre-order $(\mathtt{Strategies}_{\oplus}, \sqsubseteq)$. Altogether, we get the following.

Theorem 3.1
If a pre-order $(\mathtt{Strategies}_{\oplus}, \sqsubseteq)$ *satisfies postulates P1. and P2., and an* `Improve` *operator satisfies postulates I1. and I2. then strategy improvement algorithm is a correct algorithm for solving cycle-domination games with respect to* $\sqsubseteq$.

4 Discrete strategy improvement algorithm

Below we give a particular proposal for a pre-order $(\mathtt{Strategies}_{\oplus}, \sqsubseteq)$ satisfying postulates P1. and P2., and an `Improve` operator satisfying postulates I1. and I2. These definitions are based on discrete valuations assigned to strategies and hence give rise to a purely discrete strategy improvement algorithm for solving cycle-domination games.

4.1 Play values

For every $w \in V$, we define $V_{>w} = \{\, v \in V \,:\, v > w \,\}$, and $V_{<w} = \{\, v \in V \,:\, v < w \,\}$.

Let $P = \langle v_1, v_2, \ldots, v_\ell \rangle$ be a play, and let $v_k = \lambda(P)$, i.e., the cycle value of P is the k-th element of P. Let $\mathtt{Prefix}(P) = \{v_1, v_2, \ldots, v_{k-1}\}$, i.e., $\mathtt{Prefix}(P)$ is the set of vertices in play P occurring before the loop value of P. We define the *primary path value* $\pi(P)$ of play P to be $\mathtt{Prefix}(P) \cap V_{>\lambda(P)}$, i.e., the set of vertices occurring in P with priorities bigger than $\lambda(P)$. We define the *secondary path value* $\#(P)$ of play P to be $\left|\mathtt{Prefix}(P)\right|$, i.e., the length of the path from the initial vertex of P to $\lambda(P)$. The *path value* of play P is defined to be the ordered pair $\big(\pi(P), \#(P)\big) \in \wp(V) \times \mathbb{N}$. The *play value* $\Theta(P)$ of play P is defined to be the ordered pair $\Big(\lambda(P), \big(\pi(P), \#(P)\big)\Big) \in V \times \big(\wp(V) \times \mathbb{N}\big)$. We write `PlayValues` to denote the image of the function $\Theta : \mathtt{Plays} \to V \times \big(\wp(V) \times \mathbb{N}\big)$. Note that

$$\mathtt{PlayValues} \subseteq \big\{\, \big(v, (B, k)\big) \,:\, B \subseteq V_{>v} \,\big\}. \tag{1}$$

4.2 Linear order $\trianglelefteq$ on `PlayValues`

For all $v \in V$, we define

$$\texttt{Reward}(v) = \begin{cases} -v & \text{if } v \in R_-, \\ v & \text{if } v \in R_+. \end{cases}$$

For all $v, w \in V$, we define $v \preceq w$ to hold if and only if $\texttt{Reward}(v) \leq \texttt{Reward}(w)$. We write $v \prec w$ if $v \preceq w$ and $v \neq w$. Note that $\preceq$ is a linear order on V, and $m \prec p$ for all $m \in R_-$ and $p \in R_+$.

For $B, C \in \wp(V)$, such that $B \neq C$, we define

$$\texttt{MaxDiff}(B, C) = \max^{\preceq}\big((B \setminus C) \cup (C \setminus B)\big).$$

We define $B \prec C$ to hold if and only if either

- $\texttt{MaxDiff}(B, C) \in (B \setminus C) \cap R_-$, or
- $\texttt{MaxDiff}(B, C) \in (C \setminus B) \cap R_+$.

We write $B \preceq C$ if $B \prec C$ or $B = C$. For $\ell \in V$, we write $B \preceq_\ell C$ if $B \cap V_{>\ell} \preceq C \cap V_{>\ell}$. We write $B =_\ell C$ if $B \preceq_\ell C$ and $C \preceq_\ell B$. We write $B \prec_\ell C$ if $B \preceq_\ell C$ and $C \not\preceq_\ell B$.

For all $\ell \in V$, and $(B, k), (B', k') \in \wp(V) \times \mathbb{N}$, we define the relation $(B, k) \trianglelefteq_\ell (B', k')$ to hold if and only if either

- $B \prec_\ell B'$, or
- $B =_\ell B'$ and either
  - $\ell \in R_-$ and $k \leq k'$, or
  - $\ell \in R_+$ and $k' \leq k$.

We write $(B, k) \triangleleft_\ell (B', k')$ if $(B, k) \trianglelefteq_\ell (B', k')$ and $(B', k') \not\trianglelefteq_\ell (B, k)$.

For all $\big(\ell, (B, k)\big), \big(\ell', (B', k')\big) \in V \times \big(\wp(V) \times \mathbb{N}\big)$, we define the relation $\big(\ell, (B, k)\big) \trianglelefteq \big(\ell', (B', k')\big)$ to hold if and only if either

- $\ell \prec \ell'$, or
- $\ell = \ell'$ and $(B, k) \trianglelefteq_\ell (B', k')$.

For $\xi, \xi' \in V \times \big(\wp(V) \times \mathbb{N}\big)$, we write $\xi \triangleleft \xi'$ if $\xi \trianglelefteq \xi'$ and $\xi' \not\trianglelefteq \xi$.

Note that (1) guarantees that $\trianglelefteq$ is a linear order on `PlayValues`. It is a linear pre-order on $V \times \big(\wp(V) \times \mathbb{N}\big)$.

Example. Note that $\xi_1 = \big(3, \{1\}, 2\big)$ and $\xi_2 = (3, \{2\}, 2)$ are not play values, because $\{1\} \not\subseteq V_{>3}$ and $\{2\} \not\subseteq V_{>3}$, respectively. We have $\xi_1 \trianglelefteq \xi_2$ and $\xi_2 \trianglelefteq \xi_1$, even though $\xi_1 \neq \xi_2$. [Example] □

4.3 Pre-order $\sqsubseteq$ on $\texttt{Strategies}_\oplus$

We write $\texttt{Valuations}$ to denote the set $(V \to \texttt{PlayValues})$ of functions assigning a play value to every vertex of the game graph. We extend the $\trianglelefteq$ order to a partial order on $\texttt{Valuations}$ by defining it point-wise, i.e., for $\Xi, \Xi' \in \texttt{Valuations}$ we define $\Xi \trianglelefteq \Xi'$ to hold if $\Xi(v) \trianglelefteq \Xi'(v)$ for all $v \in V$. We write $\Xi \triangleleft \Xi'$ if $\Xi \trianglelefteq \Xi'$ and $\Xi \neq \Xi'$.

Valuation Ω_σ. For every strategy $\sigma \in \texttt{Strategies}_\oplus$ we define $\Omega_\sigma \in \texttt{Valuations}$ in the following way:

$$\Omega_\sigma(v) \ = \ \min{}^{\trianglelefteq}\big\{ \ \Theta(P) \ : \ P \in \texttt{Plays}_\sigma(v) \ \big\},$$

where $\texttt{Plays}_\sigma(v)$ is the set of plays starting from vertex v and consistent with σ. Finally, for $\sigma, \sigma' \in \texttt{Strategies}_\oplus$ we define $\sigma \sqsubseteq \sigma'$ to hold if and only if $\Omega_\sigma \trianglelefteq \Omega_{\sigma'}$. We write $\sigma \sqsubset \sigma'$ if $\Omega_\sigma \triangleleft \Omega_{\sigma'}$.

4.4 An $\texttt{Improve}$ operator

We say that a set $I \subseteq E$ is unambiguous if $(v, w) \in I$ and $(v, u) \in I$ imply that $w = u$. For every unambiguous set of edges I, we define an operator $\texttt{Switch}_I : \texttt{Strategies}_\oplus \to \texttt{Strategies}_\oplus$ as follows:

$$\big[\texttt{Switch}_I(\sigma)\big](v) = \begin{cases} w & \text{if } (v, w) \in I \text{ for some } w \in V, \\ \sigma(v) & \text{otherwise.} \end{cases}$$

We say that an edge $(v, w) \in E$ is an *improvement* for a strategy σ if

$$\Omega_\sigma\big(\sigma(v)\big) \triangleleft \Omega_\sigma(w). \tag{2}$$

We define a (non-deterministic) operator $\texttt{Improve} : \texttt{Strategies}_\oplus \to \texttt{Strategies}_\oplus$ as follows:

$$\texttt{Improve}(\sigma) = \begin{cases} \texttt{Switch}_I(\sigma) & \text{for some set } I \neq \emptyset \text{ of improvements for } \sigma, \\ \sigma & \text{if there are no improvements for } \sigma. \end{cases}$$

Let $I_1, I_2, \ldots, I_t \subseteq E$, and let σ be a strategy for player $\oplus$. Define $\sigma_0 = \sigma$, and $\sigma_k = \texttt{Switch}_{I_k}(\sigma_{k-1})$ for $k > 0$. We say that $P = \langle I_1, I_2, \ldots, I_t \rangle$ is a *strategy improvement policy* for σ if:

- for all $k \leq t$, we have that $I_k \neq \emptyset$ is an unambiguous set of improvements for σ_{k-1}, and
- there are no improvements for σ_t.

4.5 A few technical definitions

Pre-orders $\trianglelefteq_\ell$ on $\mathcal{M}(V) \times \mathbb{N}$. For technical reasons, for every $\ell \in V$, we extend the pre-order $\trianglelefteq_\ell$ on $\wp(V) \times \mathbb{N}$ to $\mathcal{M}(V) \times \mathbb{N}$, where $\mathcal{M}(V)$ is the set of multi-sets of elements of V. The definitions of $\preceq_\ell$, and $\trianglelefteq_\ell$ are the same as for $\wp(V) \times \mathbb{N}$; the only difference is that `MaxDiff` for $B, C \in \mathcal{M}(V)$ is defined by

$$\texttt{MaxDiff}(B, C) = \max^{\leq}\big((B \setminus_{\mathcal{M}} C) \cup (C \setminus_{\mathcal{M}} B)\big),$$

where $\setminus_{\mathcal{M}}$ is the multiset difference, and $B \prec C$ is defined to holds if and only if either

- $\texttt{MaxDiff}(B, C) \in (B \setminus_{\mathcal{M}} C) \cap R_-$, or
- $\texttt{MaxDiff}(B, C) \in (C \setminus_{\mathcal{M}} B) \cap R_+$.

Example. If $3 \in R_-$ then we have $\{1, 3, 3\} \preceq \{1, 2, 3\}$ because

$$\texttt{MaxDiff}\big(\{1, 3, 3\}, \{1, 2, 3\}\big) = 3$$

and $3 \in \big(\{1, 3, 3\} \setminus_{\mathcal{M}} \{1, 2, 3\}\big) \cap R_-$. [Example] $\square$

Pre-orders $\widetilde{\trianglelefteq}_\ell$ on $\mathcal{M}(V) \times \mathbb{N}$. We define the following weakening of the $\trianglelefteq_\ell$ pre-order. For $\ell \in V$ and $(B, k), (B', k') \in \mathcal{M}(V) \times \mathbb{N}$, we define $(B, k)\widetilde{\trianglelefteq}_\ell(B', k')$ to hold if and only if $B \preceq_\ell B'$. We write $(B, k) \cong_\ell (B', k')$ if $(B, k)\widetilde{\trianglelefteq}_\ell(B', k')$ and $(B', k')\widetilde{\trianglelefteq}_\ell(B, k)$. Note that $\widetilde{\trianglelefteq}_\ell$ is a pre-order on $\mathcal{M}(V) \times \mathbb{N}$, and that $\widetilde{\trianglelefteq}_\ell$ is coarser than $\trianglelefteq_\ell$, i.e., $(B, k) \trianglelefteq_\ell (B', k')$ implies $(B, k)\widetilde{\trianglelefteq}_\ell(B', k')$. Note, for example, that if $2 \in R_+$ then we have $\big(\{2, 3\}, 1\big) \cong_2 \big(\{1, 3\}, 2\big)$, but $\big(\{2, 3\}, 1\big) \not\trianglelefteq_2 \big(\{1, 3\}, 2\big)$.

Operation $\boxplus$ on $\mathcal{M}(V) \times \mathbb{N}$. For $(B, k), (B', k') \in \mathcal{M}(V) \times \mathbb{N}$, we define $(B, k) \boxplus (B', k') = \big(B \cup_{\mathcal{M}} B', k + k'\big)$, where $\cup_{\mathcal{M}}$ is the multi-set union. We also use the following shorthand: if $C \in \wp(V)$ then $(B, k) \boxplus C$ is defined to be equal to $(B, k) \boxplus \big(C, |C|\big)$, i.e., we have that $(B, k) \boxplus C = \big(B \cup_{\mathcal{M}} C, k + |C|\big)$. Moreover, if $v \in V$ then we often write $(B, k) \boxplus v$ instead of $(B, k) \boxplus \{v\}$. In other words, we have that $(B, k) \boxplus v = (B \cup_{\mathcal{M}} \{v\}, k + 1)$.

5 Correctness of the algorithm

In order to prove correctness of our discrete strategy improvement algorithm, i.e., to establish Theorem 3.1, it suffices to argue that the definitions of subsection 4 satisfy postulates P1., P2., I1., and I2.

5.1 Valuation Ω_σ and shortest paths

Let $\Omega_\sigma = (\Lambda_\sigma, \Pi_\sigma) : V \to V \times \big(\wp(V) \times \mathbb{N}\big)$. We establish the following characterization of Ω_σ.

1. Let $T_\sigma = \{t \in V : \text{there is a cycle } C \text{ in } G_\sigma, \text{ such that } \max^{\leq}(C) = t\}$. Then for all $v \in V$, we have that

$$\Lambda_\sigma(v) = \min{}^{\preceq}\{\, t \in T_\sigma \;:\; \text{there is a path from } v \text{ to } t \text{ in } G_\sigma \,\}. \quad (3)$$

2. For $v, t \in V$, we define $\texttt{Path}_\sigma(v,t)$ to be the set of sets of vertices $S \in \wp(V)$, such that $S \cup \{t\}$ is the set of vertices occurring on a (simple) path from v to t in G_σ. Let $v \in V$ and let $\ell = \Lambda_\sigma(v)$. Then we have

$$\Pi_\sigma(v) = \min{}^{\trianglelefteq_\ell}\Big\{ \big(S \cap V_{>\ell}, |S|\big) \;:\; S \in \texttt{Path}_\sigma(v,\ell) \Big\}. \quad (4)$$

Clause 1. is straightforward. Note that for all $(v,w) \in E_\sigma$ we have

$$\Lambda_\sigma(v) \preceq \Lambda_\sigma(w). \quad (5)$$

We argue that clause 2. holds. Let $P \in \texttt{Plays}_\sigma(v)$, such that $\Theta(P) = \Omega_\sigma(v)$. Let $R = \texttt{Prefix}(P)$ and let $C = \texttt{Cycle}(P)$. Then $\Lambda_\sigma(v) = \lambda(P)$ and

$$\Pi_\sigma(v) \;=\; \big(\pi(P), \#(P)\big) \;=\; \big((R \cup C) \cap V_{>\ell}, |R|\big),$$

i.e., $\Pi_\sigma(v) = \big(R \cap V_{>\ell}, |R|\big)$; the last equality holds because $\max^{\leq}(C) = \lambda(P)$. Therefore, we have that

$$\begin{aligned}
\Pi_\sigma(v) &= \min{}^{\trianglelefteq_\ell}\Big\{ \big(\pi(P), \#(P)\big) \;:\; P \in \texttt{Plays}_\sigma(v) \text{ and } \lambda(P) = \ell \Big\} \\
&= \min{}^{\trianglelefteq_\ell}\Big\{ \big(S \cap V_{>\ell}, |S|\big) \;:\; S \in \texttt{Path}_\sigma(v,\ell) \Big\}.
\end{aligned}$$

As a result of the above we get a characterization of $\Pi_\sigma(v)$ as a solution of the following shortest paths problem instance. For $\ell \in T_\sigma$ consider the subgraph $G_\sigma^\ell = (V^\ell, E_\sigma^\ell)$ of G_σ induced by $V^\ell = \{v \in V : \Lambda_\sigma(v) = \ell\}$. With every edge $(v,w) \in E_\sigma^\ell$ we associate the weight $\big(\{v\} \cap V_{>\ell}, 1\big) \in \mathcal{M}(V) \times \mathbb{N}$. The set of weights of paths in G_σ^ℓ is $\mathcal{M}(V) \times \mathbb{N}$ with $\boxplus$ as the operation of adding weights. Shortest paths are taken with respect to the $\trianglelefteq_\ell$ pre-order on $\mathcal{M}(V) \times \mathbb{N}$.

Remark. Section 26.4 of the book by Cormen et al. [5] describes an algebraic structure called a closed semi-ring that allows to define a shortest paths problem and devise algorithms for finding shortest paths. We leave it as an exercise to the reader to verify that for every $\ell \in V$, the set $\mathcal{M}(V) \times \mathbb{N}$ with the *extension operator* $\boxplus$ and the *summation operator* $\max^{\trianglelefteq_\ell}$ forms a closed semi-ring. [Remark] $\square$

Note that from $\Lambda_\sigma(v) = \ell$ for all $v \in V^\ell$, it follows that all cycles in G_σ^ℓ have "non-negative" weight, i.e., they have weight $\trianglelefteq_\ell$-bigger than $(\emptyset, 0)$ if $\ell \in R_-$, and $\trianglelefteq_\ell$-bigger than $(\emptyset, +\infty)$ if $\ell \in R_+$. Therefore, shortest paths to ℓ exist from each vertex $v \in V^\ell$, they are simple paths, and $\Pi_\sigma(v)$ is the weight of the shortest path from v to ℓ in G_σ^ℓ.

Observe, that for $v \neq \ell$, the characterization of $\Pi_\sigma(v)$ as the weight of a shortest path from v to ℓ implies, that if $w \in V^\ell$ is a successor of v on a shortest path from v to ℓ in G_σ^ℓ then

$$\Pi_\sigma(v) \ =_\ell \Pi_\sigma(w) \ \boxplus \ v, \tag{6}$$

and for all $(v, u) \in E_\sigma^\ell$ we have

$$\Pi_\sigma(v) \ \trianglelefteq_\ell \Pi_\sigma(u) \ \boxplus \ v. \tag{7}$$

Note that we have

$$\Pi_\sigma(\ell) = (\emptyset, 0). \tag{8}$$

Moreover, observe that by definition of $\ell \in T_\sigma$, there is a cycle C in G_σ^ℓ, such that $\max^{\leq}(C) = \ell$, and there are no cycles with "negative" weight. Therefore, if w is the successor of ℓ with a shortest path to ℓ then $\Pi_\sigma(w) = (\emptyset, k)$ for some $k \in \mathbb{N}$, and hence we have

$$\Pi_\sigma(\ell) \ \cong_\ell \Pi_\sigma(w) \ \boxplus \ \ell, \tag{9}$$

and for all $(\ell, u) \in E_\sigma^\ell$ we have

$$\Pi_\sigma(\ell) \mathrel{\widetilde{\trianglelefteq}_\ell} \Pi_\sigma(u) \ \boxplus \ \ell. \tag{10}$$

5.2 Locally progressive valuations

Strategy $\overline{\sigma}$ for player $\ominus$. Let σ be a strategy for player $\oplus$. We define a strategy $\overline{\sigma}$ for player $\ominus$ in the following way. For every $v \in M_\ominus$, we set:

$$\overline{\sigma}(v) = w, \text{ such that } \Omega_\sigma(w) \trianglelefteq \Omega_\sigma(u) \text{ for all } u \in \text{succ}(v),$$

i.e., $\overline{\sigma}(v)$ is defined to be a successor of v which minimizes the value of Ω_σ with respect to $\trianglelefteq$. Note that from the characterization of Ω_σ from the previous subsection it follows that for all $(v, w) \in E_{\sigma\overline{\sigma}}$, we have

$\Lambda_\sigma(v) = \Lambda_\sigma(w)$. Moreover, claims (6) and (9) hold, respectively, for all $(v, w) \in E_{\sigma\overline{\sigma}}$, depending on whether $\Lambda_\sigma(v) \neq v$ or $\Lambda_\sigma(v) = v$, respectively. These observations together with claim (8) motivate the following notion of a locally progressive valuation.

Locally progressive valuation. Let $\Xi = (\Lambda, \Pi) : V \to V \times \big(\wp(V) \times \mathbb{N}\big)$ be a valuation. We define $\texttt{Prog}(\Xi, e)$ to hold for $e = (v, w) \in E$, if and only if:

1. $\Lambda(v) = \Lambda(w)$, and
2. if $\Lambda(v) \neq v$ then $\Pi(v) =_{\Lambda(v)} \Pi(w) \boxplus v$, and
3. if $\Lambda(v) = v$ then $\Pi(v) \cong_{\Lambda(v)} \Pi(w) \boxplus v$ and $\Pi(v) = (\emptyset, 0)$.

We say that Ξ is a *locally progressive valuation* for strategies σ and τ if and only if $\texttt{Prog}(\Xi, e)$ holds, for all $e \in E_{\sigma\tau}$.

The following fact follows immediately from the definition of strategy $\overline{\sigma}$ and from (3), (6), (8), and (9).

Proposition 5.1 Valuation Ω_σ is a locally progressive valuation for σ and $\overline{\sigma}$.

Valuation $\Theta_{\sigma\tau}$. Note that if σ and τ are strategies for player $\oplus$ and for player $\ominus$, respectively, then for every vertex $v \in V$, there is a unique play $P_{\sigma\tau}(v)$ starting from v and consistent with σ and τ. We write $\Theta_{\sigma\tau}$ for a valuation defined by $\Theta_{\sigma\tau}(v) = \Theta\big(P_{\sigma\tau}(v)\big)$.

In the following lemma we establish that a locally progressive valuation for σ and τ is a local characterization of the valuation $\Theta_{\sigma\tau}$.

Lemma 5.2 A valuation Ξ is a locally progressive valuation for σ and τ if and only if $\Xi = \Theta_{\sigma\tau}$.

Proof. Let $v \in V$ and let $P_{\sigma\tau}(v) = \langle v_1, v_2, \ldots, v_\ell\rangle$ be the unique play starting from v and consistent with σ and τ. It is easy to verify that $\Theta_{\sigma\tau}$ is a locally progressive valuation for σ and τ. We show that if Ξ is a locally progressive valuation for σ and τ then $\Xi(v) = \Theta\big(P_{\sigma\tau}(v)\big)$.

Let $C = \{v_k, v_{k+1}, \ldots, v_{\ell-1}\}$ be the cycle of $P_{\sigma\tau}(v)$, i.e., let $v_k = v_\ell$ and $k < \ell$. Let $\Xi = (\Lambda, \Pi)$, where $\Lambda : V \to V$ and $\Pi : V \to \big(\mathcal{M}(V) \times \mathbb{N}\big)$. By definition of a locally progressive valuation we get that $\Lambda(v_1) = \Lambda(v_2) = \cdots = \Lambda(v_\ell)$. Let $\ell = \Lambda(v)$. We claim that $\lambda\big(P_{\sigma\tau}(v)\big) = \Lambda(v)$, i.e., that $\max^{\leq}(C) = \ell$.

By definition of a locally progressive valuation we get that $\Pi(v_i) \cong_\ell \Pi(v_{i+1}) \boxplus v_i$ for all $i \in \{k, k+1, \ldots, \ell-1\}$. This implies that

$$\Pi(v_k) \ \cong_\ell \ \Pi(v_\ell) \boxplus \{v_k, v_{k+1}, \ldots, v_{\ell-1}\} \ = \ \Pi(v_k) \boxplus C.$$

It follows that $\max^{\leq}(C) \leq \ell$. Note that $\max^{\leq}(C) < \ell$ implies that $\Pi(v_i) =_\ell \Pi(v_{i+1}) \boxplus v_i$ for all $i \in \{k, k+1, \ldots, \ell-1\}$, i.e., that $\Pi(v_k) =_\ell \Pi(v_k) \boxplus C$ holds. This, however, is impossible because $|C| > 0$. Therefore, we get that $\max^{\leq}(C) = \ell$.

Now we argue that $\Pi(v)$ is equal to the path value of $P_{\sigma\tau}(v)$. By definition of a locally progressive valuation we get that $\Pi(v_i) =_\ell \Pi(v_{i+1}) \boxplus v_i$ for all $i \in \{1, 2, \ldots, k-1\}$, and that $\Pi(v_k) = (\emptyset, 0)$. Therefore, we get that

$$\Pi(v) \ =_\ell \ (\emptyset, 0) \ \boxplus \ \texttt{Prefix}\big(P_{\sigma\tau}(v)\big) \ =_\ell \ \Big(\pi\big(P_{\sigma\tau}(v)\big), \#\big(P_{\sigma\tau}(v)\big)\Big).$$

[Lemma 5.2] ■

A best counter-strategy against σ. Note that by Proposition 5.1, an immediate corollary of Lemma 5.2 is that

$$\Omega_\sigma = \Theta_{\sigma\overline{\sigma}}. \tag{11}$$

Hence by definition of Ω_σ we get that $\Theta_{\sigma\overline{\sigma}} \trianglelefteq \Theta_{\sigma\tau}$, for all strategies τ for player $\ominus$. In other words, $\overline{\sigma}$ is a best counter-strategy against σ.

5.3 Locally under-progressive valuations

Motivated by conditions (5), (7), (8), and (10) satisfied by Ω_σ, we introduce a relaxation of the notion of a locally progressive valuation called a *locally under-progressive valuation.*

Locally under-progressive valuation. Let $\Xi = (\Lambda, \Pi) : V \to V \times \big(\wp(V) \times \mathbb{N}\big)$ be a valuation. We define $\texttt{UnderProg}(\Xi, e)$ to hold for $e = (v, w) \in E$, if and only if:

1. $\Lambda(v) \preceq \Lambda(w)$, and

if $\Lambda(v) = \Lambda(w)$ then:

2. if $\Lambda(v) \neq v$ then $\Pi(v) \trianglelefteq_{\Lambda(v)} \Pi(w) \ \boxplus \ v$, and
3. if $\Lambda(v) = v$ then $\Pi(v) \mathrel{\widetilde{\trianglelefteq}}_{\Lambda(v)} \Pi(w) \ \boxplus \ v$ and $\Pi(v) = (\emptyset, 0)$.

We say that Ξ is a *locally under-progressive valuation* for strategy σ if and only if $\texttt{UnderProg}(\Xi, e)$ holds, for all $e \in E_\sigma$.

The following fact follows immediately from (5), (7), (8), and (10).

Proposition 5.3 Valuation Ω_σ is a locally under-progressive valuation for strategy σ.

In the following proposition we collect a couple of simple facts about locally under-progressive valuations and relations $\trianglelefteq$ and $\widetilde{\trianglelefteq}_\ell$.

Proposition 5.4 Let $\Xi = (\Lambda, \Pi) : V \to V \times \big(\wp(V) \times \mathbb{N}\big)$ be a valuation.

1. If $\texttt{UnderProg}\big(\Xi, (v, w)\big)$ holds and $\Lambda(v) = \Lambda(w)$ then we have that $\Pi(v) \mathrel{\widetilde{\trianglelefteq}_{\Lambda(v)}} \Pi(w) \boxplus v$ holds.
2. If $\texttt{UnderProg}\big(\Xi, (v, w)\big)$ holds and $\Xi(w) \trianglelefteq \Xi(u)$ then we have that $\texttt{UnderProg}\big(\Xi, (v, u)\big)$ holds.

In the next lemma we establish that a locally under-progressive valuation Ξ for a strategy σ is a witness that all plays starting from a vertex v and consistent with σ, have value at least as big as $\Xi(v)$ with respect to the $\trianglelefteq$ order.

Lemma 5.5 If Ξ is a locally under-progressive valuation for a strategy σ then $\Xi \trianglelefteq \Omega_\sigma$.

Before we prove Lemma 5.5 we collect the following important properties of relations $\trianglelefteq_\ell$ and $\widetilde{\trianglelefteq}_\ell$.

Proposition 5.6 Let $\ell \in V$, and $B, C \in \mathcal{M}(V)$, and $k \in \mathbb{N}$.

1. If $\max^{\leq}(C) \geq \ell$ then
$$\max{}^{\leq}(C) \succeq \ell \quad \text{if and only if} \quad (B, k) \mathrel{\widetilde{\trianglelefteq}_\ell} (B, k) \boxplus C.$$
2. If $\max^{\leq}(C) \neq \ell$ then
$$\max{}^{\leq}(C) \succ \ell \quad \text{if and only if} \quad (B, k) \vartriangleleft_\ell (B, k) \boxplus C.$$

Proof. Assume first that $\max^{\leq}(C) > \ell$. Then $(B, k) \mathrel{\widetilde{\vartriangleleft}_\ell} (B, k) \boxplus C$ holds if and only if $B \prec_\ell B \cup_{\mathcal{M}} C$ holds, if and only if $\max^{\leq}(C) \in R_+$ holds, if and only if $\max^{\leq}(C) \succ \ell$ holds. Observe also, that $(B, k) \vartriangleleft_\ell (B, k) \boxplus C$

holds if and only if $B \prec_\ell B \cup_{\mathcal{M}} C$ holds, if and only if $(B,k) \widetilde{\lhd}_\ell (B,k) \boxplus C$ holds.

Assume that $\max^{\leq}(C) = \ell$. Then $(B,k) \widetilde{\unlhd}_\ell (B,k) \boxplus C$ holds because $B =_\ell B \cup_{\mathcal{M}} C$; and $\max^{\leq}(C) \succeq \ell$ obviously holds.

Assume finally that $\max^{\leq}(C) < \ell$. Then $B =_\ell B \cup_{\mathcal{M}} C$ and therefore by definition of $\unlhd_\ell$ we have that $(B,k) \lhd_\ell (B,k) \boxplus C$ holds if and only if $\ell \in R_-$, if and only if $\max^{\leq}(C) \succ \ell$. [Proposition 5.6] ■

Proof (of Lemma 5.5)**.** Let $v \in V$, and let P be a play starting from v and consistent with σ. It suffices to show that $\Xi(v) \unlhd \Theta(P)$.

Let $\Xi = (\Lambda, \Pi) : V \to V \times \big(\mathcal{M}(V) \times \mathbb{N}\big)$. First we show that $\Lambda(v) \preceq \lambda(P)$. From definition of a locally under-progressive valuation it follows that the values of Λ on vertices in play P are non-decreasing with respect to $\preceq$ order, so they are all equal on the cycle $\texttt{Cycle}(P)$ of P. Hence, if we define ℓ to be the value of Λ on vertices in $\texttt{Cycle}(P)$ then we have $\Lambda(v) \preceq \ell$. Therefore, it suffices to prove that $\ell \preceq \lambda(P)$.

If $\ell \not\in \texttt{Cycle}(P)$ then applying the definition of a locally under-progressive valuation around the cycle of P we get that

$$\Pi(w) \ \unlhd_\ell \ \Pi(w) \ \boxplus \ \texttt{Cycle}(P),$$

for some vertex $w \in \texttt{Cycle}(P)$. Note that from $\ell \not\in \texttt{Cycle}(P)$ it follows that $\lambda(P) \neq \ell$, and therefore clause 2. of Proposition 5.6, with $C = \texttt{Cycle}(P)$, implies that $\lambda(P) = \max^{\leq}\big(\texttt{Cycle}(P)\big) \succeq \ell$.

If $\ell \in \texttt{Cycle}(P)$ then applying the definition of a locally under-progressive valuation around the cycle of P, together with clause 1. of Proposition 5.4, we get that

$$\Pi(w) \ \widetilde{\unlhd}_\ell \ \Pi(w) \ \boxplus \ \texttt{Cycle}(P),$$

for some vertex $w \in \texttt{Cycle}(P)$. Note that from $\ell \in \texttt{Cycle}(P)$ it follows that $\lambda(P) \geq \ell$, and therefore clause 1. of Proposition 5.6 implies that $\lambda(P) = \max^{\leq}\big(\texttt{Cycle}(P)\big) \succeq \ell$.

If $\Lambda(v) \prec \lambda(P)$ then we have $\Xi(v) \lhd \Theta(P)$ and we are done. Assume then that $\ell = \Lambda(v) = \lambda(P)$. We show that in this case $\Pi(v) \lhd_\ell \big(\pi(P), \#(P)\big)$, which immediately implies that $\Xi(v) \unlhd \Theta(P)$. By applying the definition of a locally under-progressive valuation along $\texttt{Prefix}(P)$ we get

$$\Pi(v) \ \unlhd_\ell \ \Pi(\ell) \ \boxplus \ \texttt{Prefix}(P),$$

and we also have $\Pi(\ell) = (\emptyset, 0)$ because $\Lambda(\ell) = \ell$, and hence

$$\Pi(v) \ \unlhd_\ell \ \Big(\texttt{Prefix}(P), \big|\texttt{Prefix}(P)\big|\Big) \ =_\ell \ \big(\pi(P), \#(P)\big).$$

[Lemma 5.5] ■

5.4 Strategy improvement

Lemma 5.7 For every strategy σ for player $\oplus$, we have $\sigma \sqsubseteq \texttt{Improve}(\sigma)$.

Proof. It suffices to show that Ω_σ is a locally under-progressive valuation for $\texttt{Improve}(\sigma)$. Then by Lemma 5.5 we get that $\Omega_\sigma \trianglelefteq \Omega_{\texttt{Improve}(\sigma)}$, i.e., that $\sigma \sqsubseteq \texttt{Improve}(\sigma)$.

We argue that Ω_σ is a locally under-progressive valuation for the strategy $\texttt{Improve}(\sigma)$. By Proposition 5.3 we have that Ω_σ is a locally under-progressive valuation for σ. Therefore, it suffices to check that for all $v \in M_\oplus$, the predicate $\texttt{UnderProg}$ holds for Ω_σ along the edge $\big(v, [\texttt{Improve}(\sigma)](v)\big)$. Note that by definition of the $\texttt{Improve}$ operator we have

$$\Omega_\sigma\big([\texttt{Improve}(\sigma)](v)\big) \trianglerighteq \Omega_\sigma\big(\sigma(v)\big),$$

for all $v \in M_\oplus$. Note that the $\texttt{UnderProg}$ predicate holds for Ω_σ along the edges $\big(v, \sigma(v)\big)$, for all $v \in M_\oplus$, because Ω_σ is a locally progressive valuation for σ. It then follows from clause 2. of Proposition 5.4 that the $\texttt{UnderProg}$ predicate for Ω_σ holds along every edge $\big(v, [\texttt{Improve}(\sigma)](v)\big)$, for all $v \in M_\oplus$. [Lemma 5.7] ■

5.5 Maximum strategies

Lemma 5.8 For every strategy σ for player $\oplus$, the following are equivalent:

1. strategy σ is not a maximum element in $(\texttt{Strategies}_\oplus, \sqsubseteq)$,
2. $\texttt{Improve}(\sigma) \neq \sigma$,
3. $\texttt{Improve}(\sigma) \not\sqsubseteq \sigma$.

Locally over-progressive valuation. Let $\Xi = (\Lambda, \Pi) : V \to V \times \big(\wp(V) \times \mathbb{N}\big)$ be a valuation. We define $\texttt{OverProg}(\Xi, e)$ to hold for $e = (v, w) \in E$ if and only if:

1. $\Lambda(v) \succeq \Lambda(w)$, and

if $\Lambda(v) = \Lambda(w)$ then:

2. if $\Lambda(v) \neq v$ then $\Pi(v) \trianglerighteq_{\Lambda(v)} \Pi(w) \boxplus v$, and

3. if $\Lambda(v) = v$ then $\Pi(v) \mathrel{\widetilde{\trianglerighteq}}_{\Lambda(v)} \Pi(w) \boxplus v$ and $\Pi(v) = (\emptyset, 0)$.

We say that Ξ is a *locally over-progressive valuation* for strategy τ for player $\ominus$ if and only if $\texttt{OverProg}(\Xi, e)$ holds for all $e \in E_\tau$.

Valuation $\mho_\sigma$. For every strategy $\tau \in \texttt{Strategies}_\ominus$ we define $\mho_\tau \in \texttt{Valuations}$ in the following way:

$$\mho_\tau(v) \;=\; \max{}^{\trianglelefteq}\big\{\, \Theta(P) \;:\; P \in \texttt{Plays}_\tau(v) \,\big\},$$

where $\texttt{Plays}_\tau(v)$ is the set of plays starting from vertex v and consistent with τ.

In the following lemma we show that a locally over-progressive valuation Ξ for a strategy τ is a witness that all plays starting from vertex v and consistent with τ have value at most as big as $\Xi(v)$ with respect to the $\trianglelefteq$ order on valuations.

Lemma 5.9 If Ξ is a locally over-progressive valuation for a strategy τ then $\mho_\tau \trianglelefteq \Xi$.

The above lemma is proved similarly to Lemma 5.5, using the following simple facts instead of Proposition 5.4.

Proposition 5.10 Let $\Xi = (\Lambda, \Pi) : V \to V \times \big(\wp(V) \times \mathbb{N}\big)$ be a valuation.

1. If $\texttt{OverProg}\big(\Xi, (v, w)\big)$ holds and $\Lambda(v) = \Lambda(w)$ then we have that $\Pi(v) \mathrel{\widetilde{\trianglerighteq}}_{\Lambda(v)} \Pi(w) \boxplus v$ holds.
2. If $\texttt{OverProg}\big(\Xi, (v, w)\big)$ holds and $\Xi(w) \trianglerighteq \Xi(u)$ then we have that $\texttt{OverProg}\big(\Xi, (v, u)\big)$ holds.

Proof (of Lemma 5.8)**.**
$1 \Rightarrow 2$**.** We claim that it suffices to argue that if $\texttt{Improve}(\sigma) = \sigma$ then Ω_σ is a locally over-progressive valuation for $\overline{\sigma}$. By Lemma 5.9 we then have

$$\mho_{\overline{\sigma}} \trianglelefteq \Omega_\sigma. \tag{12}$$

This, however, implies that $\kappa \sqsubseteq \sigma$ for all strategies κ for player $\oplus$. It is so because $\Omega_\kappa \trianglelefteq \mho_\tau$ holds for all strategies κ and τ, hence in particular $\Omega_\kappa \trianglelefteq \mho_{\overline{\sigma}}$ holds, so altogether we get $\Omega_\kappa \trianglelefteq \mho_{\overline{\sigma}} \trianglelefteq \Omega_\sigma$.

We argue that Ω_σ is a locally over-progressive valuation for $\overline{\sigma}$. By Proposition 5.1 we know that $\texttt{Prog}$ predicate for Ω_σ holds along all edges $(v, w) \in E$, such that $\sigma(v) = w$ or $\overline{\sigma}(v) = w$. Therefore, it suffices to check that the `OverProg` predicate for Ω_σ holds along all edges $(v,w) \in E$, such that $v \in M_\oplus$. Note that from $\texttt{Improve}(\sigma) = \sigma$ it follows that

$$\Omega_\sigma\big(\sigma(v)\big) \trianglerighteq \Omega_\sigma(w),$$

for all $v \in M_\oplus$ and $w \in \mathrm{succ}(v)$, and hence by clause 2. of Proposition 5.10 we get that $\texttt{OverProg}\big(\Omega_\sigma, (v,w)\big)$ holds.

$2 \Rightarrow 3$. Note that if $\texttt{Improve}(\sigma) \neq \sigma$ then for some $v \in M_\oplus$, the predicate `Prog` does not hold for Ω_σ along the edge $\big(v, [\texttt{Improve}(\sigma)](v)\big)$. Therefore, by Lemma 5.2 we have that $\Omega_\sigma \neq \Theta_{\texttt{Improve}(\sigma)\overline{\texttt{Improve}(\sigma)}}$, i.e., by (11) we have that $\Omega_\sigma \neq \Omega_{\texttt{Improve}(\sigma)}$. Recall that $\trianglelefteq$ is a partial order on the set of valuations, hence it must be the case that $\Omega_{\texttt{Improve}(\sigma)} \not\trianglelefteq \Omega_\sigma$ because by Lemma 5.7 we have $\Omega_\sigma \trianglelefteq \Omega_{\texttt{Improve}(\sigma)}$.

$3 \Rightarrow 1$. Straightforward. [Lemma 5.8] ■

5.6 Proving the postulates P1., P2., I1., and I2.

P1. Lemmas 5.7 and 5.8 imply that after a finite number of iterations of the strategy improvement algorithm it must be the case that $\texttt{Improve}(\sigma) = \sigma$. By Lemma 5.8 such a strategy σ is a maximum element in the pre-order $(\texttt{Strategies}_\oplus, \sqsubseteq)$.

P2. Let σ be a maximum element in $(\texttt{Strategies}_\oplus, \sqsubseteq)$. Then from Lemma 5.8 it follows that $\texttt{Improve}(\sigma) = \sigma$. By definition of Ω_σ, strategy σ is a winning strategy for player $\oplus$ from the set

$$W_\oplus^\sigma = \big\{\, v \in V \;:\; \Omega_\sigma(v) = \big(\ell, (B,k)\big) \text{ and } \ell \in R_+ \,\big\}.$$

We claim that $W_\oplus^\sigma$ is the winning set for player $\oplus$. It suffices to argue that player $\ominus$ has a winning strategy from the set $V \setminus W_\oplus^\sigma$. Note that

$$V \setminus W_\oplus^\sigma = \big\{\, v \in V \;:\; \Omega_\sigma(v) = \big(\ell, (B,k)\big) \text{ and } \ell \in R_- \,\big\},$$

and by (12) we have $\mho_{\overline{\sigma}} \trianglelefteq \Omega_\sigma$. This means, however, that $\overline{\sigma}$ is a winning strategy for player $\ominus$ from $V \setminus W_\oplus^\sigma$.

I1. Immediate from Lemmas 5.7 and 5.8.

I2. Immediate from Lemma 5.8.

6 Efficient implementation

We are given a graph $G = (V, E)$ with a linear order $\leq$ on vertices, and a vertex $t \in V$, such that:

$$\text{for every } v \in V\text{, there is a path from } v \text{ to } t \text{ in } G, \tag{13}$$

and

$$\text{for every cycle } C \text{ in } G\text{, we have that } \max^{\leq}(C) \succeq t. \tag{14}$$

Every edge $(v, w) \in E$ has the weight $\big(\{v\} \cap V_{>t}, 1\big) \in \wp(V_{>t}) \times \mathbb{N}$. The weights are ordered by $\trianglelefteq_t$ and added to each other with $\boxplus$. The task is for all vertices $v \in V$, to determine (the weight of) a shortest path from v to t in G. For all $v \in V$, let $\Sigma(v) \subseteq \wp(V_{>t})$ be the set of vertices with priority bigger than t occurring in a shortest path from v to t in G.

We use the following algorithm to compute the shortest paths.

Shortest paths
1. $E := E \setminus \big(\{t\} \times V\big)$
2. **for all** $r \in V_{>t}$ `in` $\leq$`-descending order`
3.   **if** $r \in R_+$ **then**
4. $W := \big\{\, w \in V$ `: there is a path from` w `to` t `in` $(V \setminus r, E) \,\big\}$
5. $E := E \setminus \big((W \cup \{r\}) \times (V \setminus W)\big)$
6.   **if** $r \in R_-$ **then**
7. $U := \big\{\, u \in V$ `: there is a path from` u `to` r `in` $(V, E) \,\big\}$
8. $E := E \setminus \big((U \setminus \{r\}) \times (V \setminus U)\big)$
9. **if** $t \in R_+$ **then** `find longest distances from every vertex to` t
10. **if** $t \in R_-$ **then** `find shortest distances from every vertex to` t

The algorithm works as follows. First we remove edges going out of t since we are only interested in paths from all vertices to t (line 1.). Then in every iteration of the **for all** loop (lines 2.–8.) we remove edges that cannot possibly belong to a shortest path from a vertex v to t. After the **for all** loop has been executed, from every vertex there are only paths to t containing the same set of vertices in $V_{>t}$ as a shortest path. Therefore, in order to find shortest paths from every vertex to t it suffices to find longest distances to t if $t \in R_+$, and shortest distances to t if $r \in R_-$ (lines 9. or 10.). Finding longest distances in the case when $t \in R_+$ can be done efficiently because in this case the graph is acyclic.

Let us analyze the first iteration of the **for all** loop, i.e., let $r = \max^{\leq}(V)$ and let $r > t$. There are two cases to consider: $r \in R_+$ and $r \in R_-$.

Suppose that $r \in R_+$. Then for all $v \in V$, by the definition of $\trianglelefteq_t$ and by the maximality of r with respect to $\leq$, a shortest path from v to t should avoid visiting r if possible. More formally, for every $v \in V$, we have that:

$$r \notin \Sigma(v) \text{ if and only if there is a path from } v \text{ to } t \text{ in } G \text{ in which } r \text{ does not occur.} \tag{15}$$

Therefore, the set W computed in line 4. contains all vertices $v \in V$, such that r does not occur on a shortest path from v to t. On the other hand, the set $V \setminus W$ consists of vertices from which a visit to r is "unavoidable" on a shortest path to t. It is then not hard to see that by removing in line 5. all the edges leading from W to $V \setminus W$ we only disconnect paths which cannot be shortest paths to t. Let G' be the graph after removing edges in line 5. Therefore, for every $v \in V$, we have that:

$$G' \text{ contains a shortest path from } v \text{ to } t \text{ in } G. \tag{16}$$

Moreover, after performing the deletion of edges in line 5., no path from W to t contains r, and all the paths from $V \setminus W$ to t contain r. In other words, we have that:

$$\text{if } r \in \Sigma(v) \text{ then } r \text{ occurs on every path from } v \text{ to } t \text{ in } G', \tag{17}$$

and

$$\text{if } r \notin \Sigma(v) \text{ then } r \text{ does not occur on any path from } v \text{ to } t \text{ in } G'. \tag{18}$$

Observe also, that by performing the deletion in line 5. we disconnect all cycles containing r, because then r can only be reached from vertices in $V \setminus W$, and only vertices in W are reachable form r, i.e., we have:

$$\text{there is no cycle in } G' \text{ containing } r. \tag{19}$$

We omit considering the other case, i.e., $r = \max^{\leq}(V_{>t}) \in R_-$. Instead, motivated by claims (16), (17), (18), and (19), established above for $r = \max^{\leq}(V_{>t}) \in R_+$, we argue that the following invariant is maintained by all iterations of the **for all** loop.

For $x \in V_{>t}$, let $G(x)$ be the graph after the body of **for all** loop has been performed for all $r \in V_{>t}$, such that $r > x$.

Proposition 6.1 Let $r \in V_{>t}$. For all $s \in V_{>t}$, such that $s > r$, and for all $v \in V$, we have:

1. $G(r)$ contains a shortest path from v to t in G,
2. if $s \in \Sigma(v)$ then s occurs on every path from v to t in $G(r)$,
3. if $s \notin \Sigma(v)$ then s does not occur on any path from v to t in $G(r)$,
4. there is no cycle in $G(r)$ containing s.

Proof. We prove the proposition by induction on r. Claims 1.–4. hold trivially for $r = \max^{\leq}(V)$. Assume as the induction hypothesis that clauses 1.–4. hold for some $r \in V_{>t}$. We show then, that clauses 1.–4. hold for $r' = \max^{\leq}(V_{<r})$, by analyzing what happens when the body of the **for all** loop is performed. We consider two cases.

- $r \in R_+$.

  Note that by clauses 1.-3. of the induction hypothesis, and by definition of $\trianglelefteq_t$, we have:

  $r \in \Sigma(v)$ if and only if r occurs on every path from v to t in $G(r)$.

  With this analogue of claim (15), the arguments needed to prove clauses 1.–4. for r' are the same as the ones we have used to establish claims (16)–(19).

- $r \in R_-$.

  Note that by clauses 1.-3. of the induction hypothesis, and by definition of $\trianglelefteq_t$, we have:

  $r \in \Sigma(v)$ if and only if r occurs on some path from v to t in $G(r)$. (20)

  We argue that:

  there is no cycle in $G(r)$ containing r. (21)

  Suppose the contrary is the case, i.e., that there is a cycle C containing r in $G(r)$. Then by clause 4. of the induction hypothesis we have that $\max^{\leq}(C) = r$, which together with $r \in R_-$ implies that $\max^{\leq}(C) \prec t$, a contradiction with (14). Note that (21) establishes clause 4. for r'.

  A corollary of (21) is that if $r \in \Sigma(v)$ then all paths from v to t and containing r are simple paths. Moreover, by clause 1. of the induction hypothesis there is a path from r to t in $G(r)$, and hence, claim (20) implies the following:

$$r \in \Sigma(v) \text{ if and only if there is a path from } v \text{ to } r \text{ in } G(r). \tag{22}$$

Therefore, the set U computed in line 7. contains all vertices $v \in V$, such that r occurs on a shortest path from v to t, and the set $V \setminus U$ contains all vertices from which a visit to r is not possible on a shortest path to t. Observe, that by removing edges leading from $U \setminus \{r\}$ to $V \setminus U$ we only disconnect paths which cannot be shortest paths to t. This establishes clause 1. for r'.

Note also, that by definition of U no path from $V \setminus U$ to t contains r, and after deletions of edges in line 8. we have that all paths from U to t contain r. This establishes clauses 3. and 2. for r'. [Proposition 6.1] ■

Theorem 6.2
An improvement step of our discrete strategy improvement algorithm can be performed in time $O(n \cdot m)$.

7 Time complexity

In the analysis of the running time of a strategy improvement algorithm there are two parameters of major interest:

1. the time needed to perform a single strategy improvement step,
2. the number of strategy improvement steps needed.

By Theorem 6.2 our discrete strategy improvement algorithm achieves a satisfactory bound on the former parameter.

A satisfactory analysis of the latter parameter is missing in our work. Despite the long history of strategy improvement algorithms for stochastic and payoff games [9, 4, 16] very little is known about the number of strategy improvement steps needed. The best upper bounds are exponential [4, 16] but to our best knowledge no examples are known which require more than linear number of improvement steps. We believe that our purely discrete description of strategy improvement gives new insights into the behaviour of the algorithm in the special case of parity games. The two long standing questions: whether there is a polynomial time algorithm for solving parity games [8], and, more concretely, whether a strategy improvement algorithm for parity games terminates in polynomial time [4, 16], remain open. Below we discuss some disjoint observations we have come up with so far, and some questions which we believe are worth pursuing.

Proposition 7.1 For every initial strategy, there is a strategy improvement policy of length at most n. Moreover, there is such a policy switching exactly one edge in every improvement step.

Proof. Let us fix a strategy κ which is a maximum element in the partial order $(\mathtt{Strategies}_{\oplus}, \sqsubseteq)$. Note that it suffices to show that for every strategy σ which is not a maximum element in $(\mathtt{Strategies}_{\oplus}, \sqsubseteq)$, there is an improvement $(v, w) \in E$ for σ, such that $w = \kappa(v)$.

Claim 7.2 If $I \subseteq E$ contains no improvement for σ then $\mathtt{Switch}_I(\sigma) \sqsubseteq \sigma$.

We argue how the proposition follows from the claim. Suppose for the sake of contradiction that for all $v \in M_{\oplus}$, such that $\sigma(v) \neq \kappa(v)$, we have $\Omega_\sigma\big(\kappa(v)\big) \trianglelefteq \Omega_\sigma\big(\sigma(v)\big)$. Let $I = \big\{ \big(v, \kappa(v)\big) : v \in M_{\oplus} \text{ and } \sigma(v) \neq \kappa(v) \big\}$. Note that $\mathtt{Switch}_I(\sigma) = \kappa$. Hence by Claim 7.2 we get that $\kappa \sqsubseteq \sigma$ which contradicts the assumption that κ is a maximum element in $(\mathtt{Strategies}_{\oplus}, \sqsubseteq)$ and that σ is not.

Proof (of Claim 7.2)**.** Note that for all $(v, w) \in I$ we have that

$$\Omega_\sigma(w) \trianglelefteq \Omega_\sigma\big(\sigma(v)\big). \tag{23}$$

By Proposition 5.1 we have that $\mathrm{Prog}(\Omega_\sigma, e)$ holds for all edges e in the graph of strategies σ and $\overline{\sigma}$. From (23) and from clause 2. of Proposition 5.10 it follows that $\mathtt{OverProg}(\Omega_\sigma, e)$ holds for all edges e in the graph of strategies $\mathtt{Switch}_I(\sigma)$ and $\overline{\sigma}$. Let $\sigma' = \mathtt{Switch}_I(\sigma)$. Note that in the (one-player) game $G_{\sigma'}$ we have $\mho_{\overline{\sigma}} = \Theta_{\sigma'\overline{\sigma}}$. Therefore, by applying Lemma 5.9 to $G_{\sigma'}$ we get that $\Theta_{\sigma'\overline{\sigma}} \trianglelefteq \Omega_\sigma$. Note that by definition of $\Omega_{\sigma'}$ we have that $\Omega_{\sigma'} \trianglelefteq \Theta_{\sigma'\overline{\sigma}}$, and hence we get $\Omega_{\sigma'} \trianglelefteq \Omega_\sigma$ which concludes the proof. [Claim 7.2] ■ [Proposition 7.1] ■

This contrasts with an algorithm for solving parity games based on progress measures [11], for which there are families of examples on which every policy requires an exponential number of steps.

Melekopoglou and Condon [14] exhibit families of examples of simple stochastic games for which several natural strategy improvement policies switching only one switchable vertex in every strategy improvement step have exponential length.

Problem 7.3 Are there families of examples of parity games for which there exist strategy improvement policies of super-polynomial length?

Examples of Melekopoglou and Condon [14] are Markov decision processes, i.e., one-player simple stochastic games [3]. It is an open question whether the strategy improvement algorithm using the standard policy, i.e., switching all switchable vertices in every strategy improvement step, works in polynomial time for one-player simple stochastic games [14]. In contrast, our discrete strategy improvement algorithm terminates in polynomial time for one-player parity games.

Proposition 7.4 The discrete strategy improvement algorithm terminates after $O(n^3)$ strategy improvement steps for one-player parity games.

Most algorithms for solving parity games studied in literature have roughly $O\big((n/d)^d\big)$ or $O\big((n/d)^{d/2}\big)$ worst-case running time bounds, where d is the number of different priorities assigned to vertices. The best upper bound we can give at the moment for the number of strategy improvement steps needed by our discrete strategy improvement algorithm is the trivial one, i.e., the number of different strategies for player 0, which can be $2^{\Omega(n)}$.

Proposition 7.5 The discrete strategy improvement algorithm terminates after $\prod_{v \in V_0} \text{out-deg}(v)$ many strategy improvement steps.

There is, however, a variation of the strategy improvement algorithm for parity games, for which the number of strategy improvement steps is bounded by $O\big((n/d)^d\big)$.

Proposition 7.6 There is a strategy improvement algorithm for parity games for which all policies have length $O\big((n/d)^d\big)$, and every strategy improvement step can be performed in $n^{O(1)}$ time.

Note that in every strategy improvement step the current valuation Ω_σ strictly improves with respect to $\trianglelefteq$ in at least one vertex. We say that a strategy improvement step is *substantial* if in the current valuation the loop value for some vertex strictly improves. Observe that there can be at most $O(n^2)$ substantial strategy improvement steps. It follows that in search for superpolynomial examples one has to manufacture gadgets allowing long sequences of non-substantial strategy improvement steps.

We have collected a little experimental evidence that in practice most improvement steps are non-substantial. There are few interesting scalable families of hard examples of parity games known in literature. Using an implementation of our discrete strategy improvement algorithm by Schmitz and Vöge [17] we have run some experiments on families of examples taken from [2] and from [11], and on a family of examples mentioned in [11] which make Zielonka's version [22] of the McNaughton's algorithm [12] work in exponential time. For all these families only linear number of strategy improvement steps were needed and, interestingly, the number of non-substantial strategy improvement steps was in all cases constant, i.e., not dependent of the size of the game graph.

We conclude with a number of questions to pursue.

Problem 7.7 Does our discrete algorithm, with a policy switching in every strategy improvement step a maximal set of improvements (i.e., switching all switchable vertices), terminate in polynomial time? If not, exhibit families of examples for which there are policies of exponential length.

Problem 7.8 Are there polynomial time computable heuristics for constructing policies of polynomial length?

Problem 7.9 Define and study other partial orders on the set of strategies and other strategy improvement operators.

Problem 7.10 Develop other algorithms than a strategy improvement algorithm for solving the optimization problem of solving cycle-domination games with respect to a partial order on the set of strategies.

References

[1] A. Browne, E. M. Clarke, S. Jha, D. E. Long, and W. Marrero. An improved algorithm for the evaluation of fixpoint expressions. *Theoretical Computer Science*, 178(1–2):237–255, May 1997. A preliminary version appeared in Proceedings of CAV'94, volume 818 of LNCS, Springer-Verlag.

[2] Nils Buhrke, Helmut Lescow, and Jens Vöge. Strategy construction in infinite games with Streett and Rabin chain winning conditions. In Tiziana Margaria and Bernhard Steffen, editors, *Tools and Algorithms for Construction and Analysis of Systems, Second International Workshop, TACAS '96*, volume 1055 of *LNCS*, pages 207–224, Passau, Germany, 27–29 March 1996. Springer-Verlag.

[3] Anne Condon. The complexity of stochastic games. *Information and Computation*, 96:203–224, 1992.

[4] Anne Condon. On algorithms for simple stochastic games. In Jin-Yi Cai, editor, *Advances in Computational Complexity Theory*, volume 13 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 51–73. American Mathematical Society, 1993.

[5] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. MIT Press/McGraw-Hill, 1990.

[6] A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *Int. Journal of Game Theory*, 8(2):109–113, 1979.

[7] E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy (Extended abstract). In *32nd Annual Symposium on Foundations of Computer Science*, pages 368–377, San Juan, Puerto Rico, 1–4 October 1991. IEEE Computer Society Press.

[8] E. A. Emerson, C. S. Jutla, and A. P. Sistla. On model-checking for fragments of μ-calculus. In Costas Courcoubetis, editor, *Computer Aided Verification, 5th International Conference, CAV'93*, volume 697 of *LNCS*, pages 385–396, Elounda, Greece, June/July 1993. Springer-Verlag.

[9] A. Hoffman and R. Karp. On nonterminating stochastic games. *Management Science*, 12:359–370, 1966.

[10] Marcin Jurdziński. Deciding the winner in parity games is in UP $\cap$ co-UP. *Information Processing Letters*, 68(3):119–124, November 1998.

[11] Marcin Jurdziński. Small progress measures for solving parity games. In Horst Reichel and Sophie Tison, editors, *STACS 2000,*

17th Annual Symposium on Theoretical Aspects of Computer Science, Proceedings, volume 1770 of *Lecture Notes in Computer Science*, pages 290–301, Lille, France, February 2000. Springer-Verlag.

[12] Robert McNaughton. Infinite games played on finite graphs. *Annals of Pure and Applied Logic*, 65(2):149–184, 1993.

[13] Nimrod Megiddo. Towards a genuinely polynomial algorithm for linear programming. *SIAM Journal on Computing*, 12:347–353, 1983.

[14] Mary Melekopoglou and Anne Condon. On the complexity of the policy improvement algorithm for Markov decision processes. *ORSA Journal of Computing*, 6(2):188–192, 1994. Operations Research Society of America.

[15] A. W. Mostowski. Games with forbidden positions. Technical Report 78, University of Gdańsk, 1991.

[16] Anuj Puri. *Theory of Hybrid Systems and Discrete Event Systems.* PhD thesis, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, December 1995. Memorandum No. UCB/ERL M95/113.

[17] Dominik Schmitz and Jens Vöge. Implementation of a strategy improvement algorithm for finite-state parity games (Extended abstract). Fifth International Conference on Implementation and Application of Automata, CIAA 2000, Proceedings, London, Ontario, Canada, July 2000. Available at `http://www-i7.informatik.rwth-aachen.de/~voege/omega/`.

[18] Helmut Seidl. Fast and simple nested fixpoints. *Information Processing Letters*, 59(6):303–308, September 1996.

[19] Colin Stirling. Local model checking games (Extended abstract). In Insup Lee and Scott A. Smolka, editors, *CONCUR'95: Concurrency Theory, 6th International Conference*, volume 962 of *LNCS*, pages 1–11, Philadelphia, Pennsylvania, 21–24 August 1995. Springer-Verlag.

[20] Wolfgang Thomas. On the synthesis of strategies in infinite games. In *12th Annual Symposium on Theoretical Aspects of Computer Science*, volume 900 of *LNCS*, pages 1–13, Munich, Germany, 2–4 March 1995. Springer-Verlag.

[21] Jens Vöge and Marcin Jurdziński. A discrete strategy improvement algorithm for solving parity games (Extended abstract). In E. A. Emerson and A. P. Sistla, editors, *Computer Aided Verification, 12th International Conference, CAV 2000, Proceedings*, volume 1855 of *Lecture Notes in Computer Science*, pages 202–215, Chicago, IL, USA, July 2000. Springer-Verlag.

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

[23] Uri Zwick and Mike Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158:343–359, 1996.

Recent BRICS Report Series Publications

RS-00-48 **Marcin Jurdziński and Jens Vöge.** ***A Discrete Stratety Improvement Algorithm for Solving Parity Games.*** **December 2000. 31 pp. Extended abstract appears in Emerson and Sistla, editors, *Computer-Aided Verification: 11th International Conference*, CAV '00 Proceedings, LNCS 1855, 2000, pages 202–215.**

RS-00-47 **Lasse R. Nielsen.** ***A Denotational Investigation of Defunctionalization.*** **December 2000. 50 pp. Presented at *16th Workshop on the Mathematical Foundations of Programming Semantics*, MFPS '00 (Hoboken, New Jersey, USA, April 13–16, 2000).**

RS-00-46 **Zhe Yang.** ***Reasoning About Code-Generation in Two-Level Languages.*** **December 2000.**

RS-00-45 **Ivan B. Damgård and Mads J. Jurik.** ***A Generalisation, a Simplification and some Applications of Paillier's Probabilistic Public-Key System.*** **December 2000. 18 pp. Appears in Kim, editor, *Fourth International Workshop on Practice and Theory in Public Key Cryptography*, PKC '01 Proceedings, LNCS 1992, 2001, pages 119–136. This revised and extended report supersedes the earlier BRICS report RS-00-5.**

RS-00-44 **Bernd Grobauer and Zhe Yang.** ***The Second Futamura Projection for Type-Directed Partial Evaluation.*** **December 2000. To appear in *Higher-Order and Symbolic Computation*. This revised and extended report supersedes the earlier BRICS report RS-99-40 which in turn was an extended version of Lawall, editor, *ACM SIGPLAN Workshop on Partial Evaluation and Semantics-Based Program Manipulation*, PEPM '00 Proceedings, 2000, pages 22–32.**

RS-00-43 **Claus Brabrand, Anders Møller, Mikkel Ricky Christensen, and Michael I. Schwartzbach.** ***PowerForms: Declarative Client-Side Form Field Validation.*** **December 2000. 21 pp. Appears in *World Wide Web Journal*, 4(3), 2000.**

RS-00-42 **Claus Brabrand, Anders Møller, and Michael I. Schwartzbach.** ***The <bigwig> Project.*** **December 2000. 25 pp.**