A subexponential lower bound for the Random Facet algorithm for Parity Games

Oliver Friedmann [*] *Thomas Dueholm Hansen* [†] *Uri Zwick* [‡]

Abstract

Parity Games form an intriguing family of infinite duration games whose solution is equivalent to the solution of important problems in automatic verification and automata theory. They also form a very natural subclass of *Deterministic Mean Payoff Games*, which in turn is a very natural subclass of turn-based *Stochastic Mean Payoff Games*. It is a major open problem whether these game families can be solved in polynomial time.

The currently fastest algorithms for the solution of all these games are adaptations of the randomized algorithms of Kalai and of Matoušek, Sharir and Welzl for *LP-type* problems, an abstract generalization of linear programming. We refer to the algorithm of Matoušek, Sharir and Welzl as the *Random Facet* algorithm. The expected running time of these algorithms is *subexponential* in the size of the game, i.e., $2^{O(\sqrt{n \log n})}$, where n is the number of vertices in the game. Matoušek constructed a family of abstract optimization problems such that the expected running time of the Random Facet algorithm, when run on a random instance from this family, is close to the subexponential upper bound given above. This shows that in the abstract setting, the $2^{O(\sqrt{n \log n})}$ upper bound on the complexity of the Random Facet algorithm is essentially tight.

It is not known, however, whether the abstract optimization problems constructed by Matoušek correspond to games of any of the families mentioned above. (It is, in fact, unlikely that they do.) There was some hope, therefore, that the Random Facet algorithm, when applied to, say, parity games, may run in polynomial time. We show, unfortunately, that this is not the case by constructing explicit parity games on which the expected running time of the Random Facet algorithm is close to the subexponential upper bound.

Our parity games are obtained by modifying the parity games used recently by Friedmann to show that the standard deterministic *policy iteration* algorithm for parity games runs in exponential time. They, in a sense, implement a *randomized counter*. They are also the first *explicit* LP-type problems on which the Random Facet algorithm is not polynomial.

[*]Department of Computer Science, University of Munich, Germany. E-mail: `Oliver.Friedmann@gmail.com`.

[†]Department of Computer Science, Aarhus University, Denmark. Supported by the Center for Algorithmic Game Theory, funded by the Carlsberg Foundation. E-mail: `tdh@cs.au.dk`.

[‡]School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel. E-mail: `zwick@tau.ac.il`.

1 Introduction

Parity games are infinite-duration full-information two-player games played on finite graphs with integer priorities assigned to their vertices. The two players, called *even* and *odd*, construct an infinite path in the game graph. Even wins if the largest priority that appears an infinite number of times on the path is even. Odd wins otherwise. For a more detailed and formal definition, see the next section.

The problem of *solving* a parity game, i.e., determining which of the two players has a *winning strategy*, is known to be equivalent to the problems of determinization and complementation of ω-tree automata, and to the problem of μ-calculus model checking. (See [EJ91],[EJS93],[Sti95],[GTW02].)

Parity games are also interesting as they form a very special subclass of *Deterministic Mean Payoff Games* (see [EM79] [GKK88] [ZP96]), which in turn form a very special subclass of turn-based *Stochastic Mean Payoff Games* (see [Con92],[AM09]). For the reduction from parity games to mean payoff games, see [Pur95]. While it is known that the decision problems corresponding to these games belong to NP $\cap$ co-NP, and even to UP $\cap$ co-UP ([Jur98]), it is a major open problem whether any of these game families can be solved in polynomial time.

There are various exponential time algorithms for solving parity games (see, e.g., [Zie98],[Jur00]). It was hoped, until recently, that the *policy iteration* algorithms of Vöge and Jurdziński [VJ00] or Schewe [Sch08], which behave extremely well in practice (see [FL09]), would solve parity games in polynomial time. Friedmann [Fri09] has recently shown, however, that this is not the case by exhibiting parity games on which these policy iteration algorithms take exponential time. The fastest deterministic algorithm currently known for solving parity games runs in $2^{O(\sqrt{n}\log n)}$ time, where n is the number of vertices in the game (see [JPZ08]). No such *deterministic* subexponential algorithms are known for the more general classes of deterministic or stochastic mean payoff games.

Slightly faster randomized algorithms for the solution of parity games can be obtained by adapting the algorithms of Kalai [Kal92, Kal97] and of Matoušek, Sharir and Welzl [MSW96] for solving *LP-type* problems, an abstract generalization of linear programming problems. The first to note the applicability of these abstract optimization algorithms to the families of games considered here was Ludwig [Lud95]. Petersson and Vorobyov [PV01] and Björklund *et al.* [BSV03],[BV05],[BV07] adapted these algorithms to work on parity games and mean payoff games. Halman [Hal07] gave a formal proof that the optimization problems involved in solving all these type of games are of LP-type. The best known upper bound on the expected running time of these algorithms on n-vertex parity games is $2^{O(\sqrt{n\log n})}$. (The number of *vertices* in a game corresponds to the *combinatorial dimension* of the equivalent LP-type problem. The number of *edges* corresponds to the number of *constraints*.)

The randomized algorithms of Kalai [Kal92, Kal97] and of Matoušek, Sharir and Welzl [MSW96] are the fastest known *combinatorial* algorithms for solving linear programs. For a discussion of the relation between these algorithms, see [Gol95]. It is a major open problem whether their expected running times on linear programs is polynomial or not.

Matoušek [Mat94] constructed a family of abstract optimization problems such that the Random Facet algorithm, i.e., the algorithm of Matoušek, Sharir and Welzl [MSW96], when run on a *random* instance from the family, has an expected running time close to the subexponential upper bound given above. It is known, however, that the hard instances in this family do not correspond to linear programs (see Gärtner [Gär02]), and they are also unlikely to correspond, say, to parity game instances.

We construct explicit parity games such that the expected running time of the Random Facet algorithm on an n-vertex game is $2^{\tilde{\Omega}(\sqrt{n})}$, where $\tilde{\Omega}(f) = \Omega(f/\log^c n)$, for some constant c. This matches, up to a polylogarithmic factor in the exponent, the upper bound given in [MSW96]. Our games also provide the first *explicit* construction of LP-type problems on which the Random Facet algorithm is not polynomial.

The rest of the paper is organized as follows. In the next section we define parity games and the other families of games considered in this paper. In Section 3 we give a general overview of the family of

policy iteration algorithms to which the Random Facet algorithm belongs. In Section 4 we describe the adaptation of the Random Facet algorithm to parity games and other families of games. In Sections 5 and 6 we give a high level, and then a complete description, of the parity games used to obtain our lower bound. In Section 7 we analyze the expected running time of the Random Facet algorithm on these games and show that it is of the form $2^{\tilde{\Omega}(\sqrt{n})}$. We end in Section 8 with some concluding remarks and open problems.

2 Parity games and payoff games

A *parity game* is tuple $G = (V_0, V_1, E, \Omega)$, where V_0 is the set of vertices controlled by player 0, V_1 is the set of vertices controlled by player 1, $E \subseteq V \times V$, where $V = V_0 \cup V_1$, is the set of edges, and $\Omega : V \to \mathbb{N}$ is a function that assigns a *priority* to each vertex. We let $E_i(G) = E \cap (V_i \times V)$, for $i = 0, 1$, denote the edges emanating from V_i. We assume that each vertex has at least one outgoing edge.

The two players, called 0 and 1, construct an infinite path in the game graph, called a *play*, as follows. They start at an initial vertex $v_0 \in V$. If the path constructed so far is $v_0 v_1 \dots v_n$, and $v_n \in V_i$, then player i selects a vertex $w \in V$ such that $(v_n, w) \in E$ and the play continues with $v_0 \dots v_n w$. Every play has a unique winner given by the *parity* of the largest priority that occurs infinitely often. Player 0 wins if this priority is even, and player 1 wins if it is odd.

A *strategy* σ for player i is a subset $\sigma \subseteq E_i(G)$ such that each vertex $v \in V_i$ has exactly one outgoing edge in σ. A play $v_0 v_1 \dots$ *conforms* to a strategy σ for player i if whenever $v_j \in V_i$, then $(v_j, v_{j+1}) \in \sigma$. A strategy σ for player i is a *winning strategy*, from v_0, if player i wins every play that begins at v_0 and conforms to σ.

Parity games are known to be *determined*, i.e., from any vertex v, exactly one of the players has a winning strategy (see [EJ91]). We let $W_i \subseteq V$, for $i = 0, 1$, be set of vertices from which player i has a winning strategy. Furthermore it is known that player i has a single strategy σ_i that is winning for i from all vertices of W_i. *Solving* a parity game means determining the sets W_0 and W_1 and a pair of winning strategies σ_0 and σ_1 for both players.

A *Deterministic Mean Payoff Game* (DMPG) is a tuple $G = (V_0, V_1, E, r)$, where V_0 and V_1 are again the sets of vertices controlled by players 0 and 1 respectively, $E \subseteq V \times V$, where $V = V_0 \cup V_1$, is the set of edges, and $r : E \to \mathbb{R}$ is an immediate *reward* function defined on the edges. The two players again construct an infinite path $v_0 v_1 \dots$ in the game graph $(V_0 \cup V_1, E)$. The outcome of a play, paid by player 1 to player 0, is then $\liminf_{n\to\infty} \frac{1}{n} \sum_{j=0}^{n-1} r(v_j, v_{j+1})$. Player 0 tries to *maximize* this outcome, while player 1 tries to *minimize* it. The two players are thus trying to maximize or minimize the limiting *average reward per turn*.

Every vertex v in a DMPG has a *value* $val(v)$, such that player 0 has a strategy σ_0 that ensures that the outcome of every play that starts at v conforms to σ_0 is at least $val(v)$, while player 1 has a strategy σ_1 that ensures that the outcome of every play that starts at v conforms to σ_1 is at most $val(v)$. (In particular, the outcome of the unique play that conforms to both σ_0 and σ_1 is exactly $val(v)$.) Such strategies are said to be optimal strategies from v. Again, both players have optimal strategies that are optimal from all vertices.

A *Deterministic Discounted Payoff Game* (DDPG) is similar to a DMPG. The only difference is that the outcome of a play is now defined to be $(1 - \lambda) \sum_{j=0}^{\infty} \lambda^j r(v_j, v_{j+1})$, where $0 < \lambda < 1$ is a *discount factor*. Discount factors have interesting economical interpretations. Discounted games are slightly simpler to work with, as the discount factor ensures the convergence of the sum $\sum_{j=0}^{\infty} \lambda^j r(v_j, v_{j+1})$, no matter what the players do. When $\lambda \to 1$, the value of the discounted game tends to the value of the mean payoff game.

A parity game $G = (V_0, V_1, E, \Omega)$ can be easily reduced into a DMPG $G' = (V_0, V_1, E, r)$, where for

every $(u, v) \in E$ we have $r(u, v) = (-n)^{\Omega(u)}$, where $n = |V|$. Player 0 has a winning strategy from vertex v in G if and only if she has a strategy that guarantees a *positive* outcome starting from v in G'. (The correctness of the reduction follows from the fact that a cycle in the game graph has a positive mean value if and only if the largest priority on one of its vertices is even.) Parity games thus form a very well-behaved subfamily of DMPGs.

Turn-based *Stochastic Mean Payoff Game* (SMPG) form an even more general family of games. A SMPG is a tuple $G = (V_0, V_1, V_R, E, p, r)$, where V_0 and V_1 are the sets of vertices controlled by players 0 and 1, V_R is a set of randomization vertices, controlled by nature, $p : E_R \to [0, 1]$ is a function that assigns a probability to each edge of $E_R = E \cap (V_R \times V)$, and $r : E \to \mathbb{R}$ is a function that assigns an immediate reward to each edge of the graph. For every $u \in V_R$ we have $\sum_{(u,v)\in E} p(u, v) = 1$. The two players again construct an infinite path in the game graph. When the last vertex reached is in V_i, where $i \in \{0, 1\}$, player i chooses the next edge. When the last vertex reached is in V_R, the next edge is chosen according to the probabilities specified by p. The two players try to maximize, respectively minimize, the long term *expected* average reward per turn. Turn-based *Stochastic Discounted Payoff Game* (SDPG) are defined similarly.

It is a tantalizing open problem whether parity games and the other families of games defined here can be solved in polynomial time. The Random Facet algorithm was considered to be a strong candidate for solving these games in polynomial time. We show, unfortunately, that this is not the case, even for parity games, the easiest of these families.

3 Policy iteration algorithms

In this section we describe a family of algorithms that can be used to solve the various games defined in the previous section. The RandomFacet algorithm, described in the next section, is a member of this family. We start by explaining how these algorithms work on DDPGs and then explain how to adapt them for the other families of games.

Policy iteration (also known as *strategy improvement*) algorithms were first developed by Howard [How60] who used them to solve infinite-horizon *Markov Decision Processes* (MDPs), a one-player variant of the stochastic two-player games defined in the previous section. They were adapted for the solution of two-player games by many researchers. (See, e.g., [HK66],[Con92],[Pur95].)

Let $G = (V_0, V_1, E, r)$ be a DDPG game with discount factor λ. Let σ and τ be policies for players 0 and 1. For every starting vertex v_0, there is a single play, i.e., infinite path, $v_0 v_1 \ldots$ that conforms to both σ and τ. The value $(1 - \lambda) \sum_{j=0}^{\infty} \lambda^j r(v_j, v_{j+1})$ of this play is defined to be the *valuation* $val_{\sigma,\tau}(v_0)$ of these pair of strategies on v_0.

For a given game G and a strategy σ of player 0, we let $\tau_{G,\sigma}$ be an *optimal counter-strategy* for player 1, i.e., a strategy τ that minimizes $val_{\sigma,\tau}(v)$, for every $v \in V$. Such optimal counter-strategies are known to exist and they can be found in polynomial time (see, e.g. [MTZ10].) We let $val_\sigma(v_0) = val_{\sigma,\tau_{G,\sigma}}(v_0)$, for every $v_0 \in V$, be the valuation of σ on v_0, and let $val_\sigma = (val_\sigma(v_1), val_\sigma(v_2), \ldots, val_\sigma(v_n))$ be the *valuation vector* of σ, where $v_1, v_2, \ldots, v_n$ is some fixed ordering of the vertices of V_0.

Let $x, y \in \mathbb{R}^n$. We say that $x \trianglelefteq y$ if and only if $x_j \leq y_j$, for every $1 \leq j \leq n$. We say that $x \triangleleft y$ if and only if $x \trianglelefteq y$ and $x \neq y$. Let σ and σ' be two strategies of player 0. In general, the valuation vectors val_σ and $val_{\sigma'}$ may be incomparable. If $val_\sigma \triangleleft val_{\sigma'}$ we say that σ' is better than σ.

If σ is a strategy for player 0 and $e = (u, v) \notin \sigma$, we let $\sigma[e]$ be the strategy obtained by replacing the edge emanating in σ from u by e. (I.e., $\sigma[e] = \sigma \cup \{e\} \setminus \{e'\}$, where $e' = (u, v') \in \sigma$.) It can be shown that the valuations val_σ and $val_{\sigma[e]}$ are always comparable, i.e., either $val_{\sigma[e]} \trianglelefteq val_\sigma$ or $val_\sigma \trianglelefteq val_{\sigma[e]}$. If $val_\sigma \triangleleft val_{\sigma[e]}$, we say that e is an *improving switch* with respect to σ.

If $e_1, e_2, \ldots, e_k \notin \sigma$ are edges emanating from different vertices, we let $\sigma[e_1, \ldots, e_k] = \sigma[e_1] \ldots [e_k]$ be the strategy obtained by performing the *multi-switch* $\{e_1, e_2, \ldots, e_k\}$, i.e., by switching each one of $e_1, e_2, \ldots, e_k$. We say that $\{e_1, e_2, \ldots, e_k\}$ is an improving multi-switch if and only if $val_\sigma \lhd val_{\sigma[e_1,\ldots,e_k]}$. Policy iteration algorithms are based on the following fundamental theorem. (See, e.g., [How60],[Con92].)

Theorem 3.1 *(i) If σ is not an optimal strategy, then there exists an improving switch $e \notin \sigma$.*
(ii) If $e_1, e_2, \ldots, e_k \notin \sigma$ are improving switches, then $\{e_1, e_2, \ldots, e_k\}$ is an improving multi-switch.

A policy iteration algorithm is an algorithm that starts with some initial strategy σ_0 of player 0 and constructs a sequence of strategies $\sigma_0, \sigma_1, \ldots, \sigma_\ell$, such that $val_{\sigma_0} \lhd val_{\sigma_1} \lhd \cdots \lhd val_{\sigma_\ell}$ and such that σ_ℓ is an optimal strategy for player 0. Strategy σ_{i+1} is obtained form σ_i by performing an improving (multi-)switch. Different policy iteration algorithms differ in the way they choose the improving switches performed in each iteration.

An appealing feature of policy iteration algorithms is that determining whether e constitutes an improving switch with respect to σ can be done *without* evaluating $\sigma[e]$. (Evaluating $\sigma[e]$ is a non-trivial operation as it involves finding an optimal counter-strategy.) An edge $e = (u, v) \notin \sigma$ of a DDPG is an improving switch if and only if $(1 - \lambda)r(u, v) + \lambda\, val_\sigma(v) > val_\sigma(u)$.

Let $e_0 = (u, v_0), e_1 = (u, v_1), \ldots, e_d = (u, v_d)$ be the edges emanating from $u \in V_0$ and suppose that $e_0 \in \sigma$. Note that several of $e_1, e_2, \ldots, e_d$ may be improving switches. The *standard* policy iteration algorithm selects for each vertex u the edge e_i that maximizes $(1 - \lambda)r(u, v_i) + \lambda\, val_\sigma(v_i)$. It then performs the corresponding multi-switch. The standard policy iteration algorithm behaves very well in practice. It was recently shown, however, that its worst-case running time is exponential for parity games (Friedmann [Fri09]) and for MDPs (Fearnley [Fea10]).

For a given set of player 0 edges $F \subseteq E_0(G)$ s.t. for every node $v \in V_0$ there is at least one edge originating from v in F, let $\sigma_{G,F} \subseteq F$ denote an (arbitrary but fixed) optimal strategy w.r.t. F.

We next move from discounted payoff games to mean payoff games. Let σ and τ be strategies of players 0 and 1 in a DMPG. Let $v_0 v_1 \ldots$ be the infinite path that conforms to σ and τ. We define the *value* $\textsc{val}_{\sigma,\tau}(v_0)$ of this play to be $\lim_{n\to\infty} \frac{1}{n} \sum_{j=0}^{n-1} r(v_j, v_{j+1})$. (Note the difference between the *value*, and the *valuation* defined below.) The infinite path $v_0 v_1 \ldots$ is composed of finite path P leading to a cycle $C = u_0 u_1 \ldots u_k$, where $u_k = u_0$, which is repeated an infinite number of times, and $\textsc{val}_{\sigma,\tau}(v_0)$ is simply the average reward $\frac{1}{k} \sum_{j=0}^{k-1} r(u_j, u_{j+1})$ of the cycle C.

We can define improving switches with respect to the values defined above. Unfortunately, in the non-discounted case, it is not the case that if σ is not optimal then there exist at least one switch that *strictly* improves the value. To remedy the situation, we define *potentials* as follows. Let $u = u(C)$ be a fixed vertex on the cycle C, e.g., the vertex with the smallest index. Let $v_0 v_1 \ldots v_\ell = u$ be the finite prefix of the infinite path $v_0 v_1 \ldots$ that ends with the first visit to u. We define the *potential* $\textsc{pot}_{\sigma,\tau}(v_0)$ to be $\sum_{j=0}^{\ell-1} (r(v_j, v_{j-1}) - \textsc{val}_{\sigma,\tau}(v_0))$, i.e., the total reward on the path leading to u, when $\textsc{val}_{\sigma,\tau}(v_0)$ is subtracted from the reward of each edge. Finally, we define the *valuation* $val_{\sigma,\tau}(v_0)$ to be the pair $(\textsc{val}_{\sigma,\tau}(v_0), \textsc{pot}_{\sigma,\tau}(v_0))$. We compare valuations *lexicographically*, i.e., $val_{\sigma,\tau}(v_0) \prec val_{\sigma',\tau'}(v_0)$ if and only if $\textsc{val}_{\sigma,\tau}(v_0) < \textsc{val}_{\sigma',\tau'}(v_0)$, or $\textsc{val}_{\sigma,\tau}(v_0) = \textsc{val}_{\sigma',\tau'}(v_0)$ and $\textsc{pot}_{\sigma,\tau}(v_0) < \textsc{pot}_{\sigma',\tau'}(v_0)$. With these slightly more complicated valuations, Theorem 3.1 becomes valid again, and policy iteration algorithms can therefore be used to solve DMPGs.

Finally, we consider parity games. Let σ and τ be strategies of players 0 and 1 in a DMPG. Let $v_0 v_1 \ldots$ be the infinite path that conforms to σ and τ. Let $C = u_0 u_1 \ldots u_k$, where $u_k = u_0$ be the cycle repeated an infinite number of times, and let $u = u(C)$ be a fixed vertex on the cycle C, e.g., the vertex with the smallest index. Let $v_0 v_1 \ldots v_\ell = u$ be the prefix leading to u. Following Vöge and Jurdziński [VJ00] (see also [Vög00]), we define the valuation $val_{\sigma,\tau}(v_0)$ to be a triplet (p, A, ℓ), where p is the largest priority on C, A is the multiset of priorities greater than p appearing on the path $v_0 v_1 \ldots v_\ell = u$, and ℓ is the

length of this path. We define a total ordering on valuations as follows: $(p_1, A_1, \ell_1) \prec (p_2, A_2, \ell_2)$ if and only if one of the following conditions hold: (i) $(-1)^{p_1} p_1 < (-1)^{p_2} p_2$; or (ii) $p_1 = p_2$, $A_1 \neq A_2$ and the largest element in $A_1 \oplus A_2$, the symmetric difference, is even and belongs to A_2, or is odd and belongs to A_1; or (iii) $p_1 = p_2$, $A_1 = A_2$ and p_1 is even and $\ell_1 > \ell_2$, or p_1 is odd and $\ell_1 < \ell_2$. With these valuations, policy iteration algorithm can be applied directly on parity games.

Let $G = (V_0, V_1, E, \Omega)$ be a parity game. We say that G is an *1-sink parity game* iff there is a node $v \in V$ s.t. $\Omega(v) = 1$, $(v, v) \in E$, $\Omega(w) > 1$ for every other node $w \in V$, v is the only cycle in G that is won by player 1 and player 1 has a winning strategy for the whole game.

Theorem 3.2 ([Fri10]) *Let G be an 1-sink parity game. Discrete policy iteration requires the same number of iterations to solve G as the policy iteration for the induced payoff games as well as turn-based stochastic games to solve the respective game G' induced by applying the standard reduction from G to the respective game class, assuming that the improvement policy solely depends on the ordering of the improving edges.*

4 The Random Facet algorithm

The RandomFacet algorithm of Matoušek, Sharir and Welzl [MSW96] is a very simple randomized algorithm for solving LP-type problems. As parity games, and the other type of games considered in Section 2, are LP-type problems (Halman [Hal07]), the algorithm can be used to solve these games, as was done by Ludwig [Lud95], Petersson and Vorobyov [PV01], and Björklund *et al.* [BSV03],[BV05],[BV07].

For concreteness, we describe the operation of the RandomFacet algorithm on parity games. Let $G = (V_0, V_1, E, \Omega)$ be a parity game. Recall that $E_0(G) = E \cap (V_0 \times V)$. Let σ be an initial strategy for player 0. If $\sigma = E_0(G)$, then σ is the only possible strategy for player 0, and is thus also an optimal strategy. Otherwise, the algorithm chooses, uniformly at random, an edge $e \in E_0(G) \setminus \sigma$ and applies the algorithm recursively on the subgame $G \setminus \{e\} = (V_0, V_1, E \setminus \{e\}, \Omega)$, the game obtained by removing e from G, with the initial strategy σ. The recursive call returns a strategy σ' which is an optimal strategy for player 0 in $G \setminus \{e\}$. If e is not an improving switch for σ, then σ' is also an optimal strategy in G. Otherwise, the algorithm performs the switch $\sigma'[e]$, and recursively calls the algorithm on G, with initial strategy $\sigma'[e]$. For pseudo-code, see the left-hand side of Figure 1. It follows from the analysis of [MSW96] that the *expected number of switches* performed by the RandomFacet algorithm on any parity game is $2^{O(\sqrt{n \log n})}$, where $n = |V_0|$ is the number of vertices controlled by player 0 in G.

We also consider a variant RandomFacet* of the RandomFacet algorithm shown on the right-hand side of Figure 1. This variant receives, as a third argument, an *index function* $\texttt{ind} : E_0(G) \to \mathbb{N}$ that assigns each edge of $E_0(G)$ a *distinct* natural number. Instead of choosing a *random* edge e from $E_0(G) \setminus \sigma$, the algorithm now chooses the edge of $E_0(G) \setminus \sigma$ with the *smallest index*. We show below that the expected running time of this modified algorithm, when $\texttt{ind}$ is taken to be a *random permutation* of $E_0(G)$, is exactly equal to the expected running time of the original algorithm. We find it more convenient to work with the modified algorithm. The fact that the ordering of the edges is selected in advance simplifies the analysis. All our results apply, of course, also to the original version.

We next derive some simple recurrence relations for the expected number of iterations performed by the RandomFacet and RandomFacet* algorithms.

Let $G = (V_0, V_1, E, \Omega)$ be a parity game and let $F \subseteq E_0(G)$. We let $G_F = (V_0, V_1, E \cap F, \Omega)$ be the subgame of G in which only the edges of F are available for player 0. Let $\sigma \subseteq F$ be a strategy of player 0 in G_F. We let $\mathbb{E}_G(F, \sigma)$ be the expected number of iterations performed the call RandomFacet(G_F, σ).

Function RandomFacet(G, σ)

```
if E_0(G) = σ then
    return σ
else
    Choose e ∈ E_0(G) \ σ uniformly at random
    σ' ← RandomFacet(G \ {e}, σ)
    if val_σ' < val_σ'[e] then
        σ'' ← σ'[e]
        return RandomFacet(G, σ'')
```

Function RandomFacet$^*(G, \sigma, \texttt{ind})$

```
if E_0(G) = σ then
    return σ
else
    e ← argmin_{e' ∈ E_0(G)\σ} ind(e')
    σ' ← RandomFacet*(G \ {e}, σ, ind)
    if val_σ' < val_σ'[e] then
        σ'' ← σ'[e]
        return
        RandomFacet*(G, σ'', ind)
```

Figure 1: Two versions of the RandomFacet algorithm.

If $\sigma = F$, then $\mathbb{E}_G(F, \sigma) = 1$. Otherwise, if $\sigma \subsetneq F$, we have

$$\mathbb{E}_G(F, \sigma) := \frac{1}{|F \setminus \sigma|} \sum_{e \in F \setminus \sigma} \mathbb{E}_G(F, e, \sigma)$$

$$\mathbb{E}_G(F, e, \sigma) := \mathbb{E}_G(F \setminus \{e\}, \sigma) + \mathbb{1}_{e \in \sigma_{G,F}} \cdot \mathbb{E}_G(F, \sigma_{G,F \setminus \{e\}}[e])$$

where $\mathbb{1}_{\texttt{pred}} \in \{0, 1\}$ denotes the *indicator function*, i.e., $\mathbb{1}_{\texttt{pred}} = 1$ iff pred holds.

Let $\mathbb{E}^*_G(F, \sigma, \texttt{ind})$ be the expected number of switches performed by a call RandomFacet$^*(G_F, \sigma, \texttt{ind})$. If $\sigma = F$, we have $\mathbb{E}^*_G(F, \sigma, \texttt{ind}) := 1$. Otherwise, if $\sigma \subsetneq F$, with $e = \operatorname{argmin}_{e' \in F \setminus \sigma} \texttt{ind}(e')$ we have

$$\mathbb{E}^*_G(F, \sigma, \texttt{ind}) := \mathbb{E}^*_G(F \setminus \{e\}, \sigma, \texttt{ind}) + \mathbb{1}_{e \in \sigma_{G,F}} \cdot \mathbb{E}^*_G(F, \sigma_{G,F \setminus \{e\}}[e], \texttt{ind}).$$

Finally, we let $\mathbb{E}^*_G(F, \sigma)$ be the expected number of iterations performed by RandomFacet$^*(G_F, \sigma, \texttt{ind})$ with a random index function ind. Let $\mathcal{I}(G)$ be the set of all index functions w.r.t. G with range $\{1, 2, \ldots, |E_0(G)|\}$. Then,

$$\mathbb{E}^*_G(F, \sigma) := \frac{1}{|\mathcal{I}(G)|} \sum_{\texttt{ind} \in \mathcal{I}(G)} \mathbb{E}^*_G(F, \sigma, \texttt{ind}).$$

Lemma 4.1 *Let G be a game, $F \subseteq E_0(G)$ and $\sigma \subseteq F$ be a player 0 strategy. Then $\mathbb{E}_G(F, \sigma) = \mathbb{E}^*_G(F, \sigma)$.*

The proof of Lemma 4.1, and the other lemmas of this section can be found in Appendix A. Basically, the lemma holds as by the linearity of the expectation, the random choices made by the two recursive calls made by RandomFacet are allowed to be dependent.

Suppose that $F \subseteq E_0(G)$ and that $H \subseteq F \setminus \sigma$. It is not necessarily the case that $\mathbb{E}^*_G(F, \sigma, \texttt{ind}) \geq \mathbb{E}^*_G(F \setminus H, \sigma, \texttt{ind})$. (In other words, removing edges does not necessarily reduce the expected number of iterations performed by RandomFacet*.) There is, however, a special case in which this holds. Namely, if all the edges of H appear in the ordering determined by ind before all other edges of $(F \setminus H) \setminus \sigma$. This simple observation simplifies our analysis.

Lemma 4.2 *Let G be a parity game, $F \subseteq E_0(G)$, $\sigma \subseteq F$ be a player 0 strategy, $\texttt{ind} \in \mathcal{I}(G)$ be an index function and $H \subseteq F \setminus \sigma$ s.t. $\texttt{ind}(e) < \texttt{ind}(e')$ for every $e \in H$ and every $e' \in (F \setminus H) \setminus \sigma$. Then $\mathbb{E}^*_G(F, \sigma, \texttt{ind}) \geq \mathbb{E}^*_G(F \setminus H, \sigma, \texttt{ind})$.*

A *choice function* $\texttt{cho} : 2^{E_0(G)} \times \mathcal{I}(G) \to (E_0(G) \cup \{\bot\})$ is a function that assigns each subset $F \setminus \sigma \subseteq E_0(G)$ and every index function $\texttt{ind} \in \mathcal{I}(G)$ an edge $e \in F \setminus \sigma$ or $\bot$; we denote the set of all choice

functions w.r.t. G by $\mathcal{C}(G)$. Intuitively, a choice function allows us to specify which edge e of $F \setminus \sigma$ is the first *interesting* edge according to the order specified by $\texttt{ind}$. Using Lemma 4.2 we can then ignore all edges that precede e.

Each choice function $\texttt{cho} \in \mathcal{C}(G)$ gives a lower bound $\mathbb{E}^{\dagger}_{G}(F, \sigma, \texttt{cho})$ for $\mathbb{E}^{*}_{G}(F, \sigma, \texttt{ind})$ defined as follows.

$$\mathbb{E}^{\dagger}_{G}(F, \sigma, \texttt{cho}) := \frac{1}{|\mathcal{I}(G)|} \sum_{\texttt{ind} \in \mathcal{I}(G)} \mathbb{E}^{\dagger}_{G}(F, \sigma, \texttt{cho}, \texttt{ind})$$

If $\texttt{cho}(F \setminus \sigma, \texttt{ind}) = \bot$, we set $\mathbb{E}^{\dagger}_{G}(F, \sigma, \texttt{cho}, \texttt{ind}) := 1$. Otherwise, if $\texttt{cho}(F \setminus \sigma, \texttt{ind}) \neq \bot$, we set

$$\mathbb{E}^{\dagger}_{G}(F, \sigma, \texttt{cho}, \texttt{ind}) := \mathbb{E}^{\dagger}_{G}(F \setminus H \setminus \{e\}, \sigma, \texttt{cho}, \texttt{ind}) + \mathbb{1}_{e \in \sigma_{G, F \setminus H}} \cdot \mathbb{E}^{\dagger}_{G}(F \setminus H, \sigma_{G, F \setminus H \setminus \{e\}}[e], \texttt{cho}, \texttt{ind})$$

where $e = \texttt{cho}(F \setminus \sigma, \texttt{ind})$ and $H = \{e' \in F \setminus \sigma \mid \texttt{ind}(e') < \texttt{ind}(e)\}$. By Lemma 4.2 we get:

Lemma 4.3 *Let G be a parity game, $F \subseteq E_0(G)$, $\sigma \subseteq F$ be a player 0 strategy and $\texttt{cho} \in \mathcal{C}(G)$ be a choice function. Then $\mathbb{E}^{*}_{G}(F, \sigma) \geq \mathbb{E}^{\dagger}_{G}(F, \sigma, \texttt{cho})$.*

5 High-level description of the construction

The parity games that we construct on which RandomFacet performs an expected subexponential number of iterations simulate a *randomized counter*. Such a counter is composed of n bits $\mathcal{B}_n = (\beta_0, \ldots, \beta_{n-1})$ all initially set to 0. The randomized counter works in a recursive manner, focusing each time on a subset $N \subseteq [n] := \{0, \ldots, n-1\}$ of the bits. Initially $N = [n]$. If $N = \emptyset$, then nothing is done. Otherwise, the counter chooses a random index $i \in [n]$ and recursively performs a randomized count on $N \setminus \{i\}$. When this recursive call is done, we have $\beta_j = 1$, for every $j \in N \setminus \{i\}$, while $\beta_i = 0$. Next, β_i is set to 1 while all bits β_j, where $j \in N \cap [i]$, are set to 0. (Note that this last step is identical to the incementation of a standard counter.) Finally, a recursive randomized count is performed on $N \cap [i]$. A function RandCount that implements a randomized counter is given on the left-hand side of Figure 2.

There is an intentional similarity between the behavior of RandCount and the behavior of the RandomFacet algorithm. A randomized counter corresponds to an "abstract game" with n vertices, each having two outgoing "edges". The two edges emanating from the i-th abstract vertex correspond to the settings "$\beta_i = 0$" and "$\beta_i = 1$". The set N corresponds to the set F of the previous section, or more precisely to the set of states for which the set F still contains the two edges "$\beta_i = 0$" and "$\beta_i = 1$". The state of the counter corresponds to the current strategy σ used by RandomFacet.

Let $g(n)$ be the expected number of steps performed by an n-bit randomized counter. It is easy to see that $g(0) = 1$ and that

$$g(n) := g(n-1) + \frac{1}{n} \sum_{i=0}^{n-1} g(i) \quad , \quad \text{for } n > 0.$$

The asymptotic behavior of $g(n)$ is known quite precisely:

Lemma 5.1 ([FS09], p. 596-597) *It holds that $g(n) \longrightarrow \frac{e^{2 \cdot \sqrt{n} - \frac{1}{2}}}{2 \cdot \sqrt{\pi} \cdot n^{\frac{1}{4}}}$ for $n \to \infty$.*

The expected number of steps performed by a randomized counter is thus just of the right subexponential form. The challenge, of course, is to construct parity games on which the behavior of the RandomFacet algorithm mimics the behavior of RandCount. We do that in the next section.

As was the case with RandomFacet, we can obtain a modified version of RandCount in which all random decisions are made in advance. In the case of RandCount, this corresponds to choosing a random permutation on $[n]$.

Function RANDCOUNT$(N, \mathcal{B}_n)$

if $N \neq \emptyset$ **then**
- Choose $i \in N$ uniformly at random
- RANDCOUNT$(N \setminus \{i\}, \mathcal{B}_n)$
- $\beta_i \leftarrow 1$
- **for** $j \in N \cap [i]$ **do** $\beta_j \leftarrow 0$
- RANDCOUNT$(N \cap [i], \mathcal{B}_n)$

Function RANDCOUNT$^*(N, \mathcal{AB}_n, \varphi)$

if $N \neq \emptyset$ **then**
- Let $i \leftarrow \operatorname{argmin}_{j \in N} \varphi(j)$
- RANDCOUNT$^*(N \setminus \{i\}, \mathcal{AB}_n, \varphi)$
- $\beta_i \leftarrow 1$
- **for** $j \in N \cap [i]$ **do** $\beta_j \leftarrow 0, \alpha_j \leftarrow 0$
- $\alpha_i \leftarrow 1$
- RANDCOUNT$^*(N \cap [i], \mathcal{AB}_n, \varphi)$

Figure 2: Counting with a randomized bit-counter (left), and counting with a modified, augmented, randomized bit-counter (right).

One of the main difficulties in mimicking the behavior of RANDCOUNT is to ensure that the bits β_j, for $j \in N \cap [i]$, are *reset* before the simulation of the second recursive call commences. (Note that we have to 'persuade' the players to perform these resets, i.e., make sure that they all correspond to profitable switches from their point of view.)

To help achieve the reset functionality we augment the randomized counter with *access bits* $(\alpha_0, \ldots, \alpha_{n-1})$. The state of the counter is now $\mathcal{AB}_n = (\alpha_0, \ldots, \alpha_{n-1}, \beta_0, \ldots, \beta_{n-1})$. In the abstract setting considered in this section these access bits do not serve any purpose. It is useful, however, to specify the way in which access bits are set and reset, as this will be crucial in the actual construction of the next section.

The augmented and modified randomized counter RANDCOUNT* is given on the right-hand side of Figure 2. The second argument $\mathcal{AB}_n$ is now composed of both the access and the counter bits. The third argument φ is a permutation on $[n]$. At each stage, instead of choosing a random element $i \in N$, RANDCOUNT* chooses the index $i \in N$ for which $\varphi(i)$ is minimized.

Access bits are eventually assigned the same values as the corresponding counter bits. Note, however, the order in which this is done. First β_i is set. Next α_j and β_j, for $j \in N \setminus \{i\}$, are reset, and only after that α_i is set. As mentioned, this will be important in the next section.

To ensure, with high probability, the desired behavior of the access and counter bits, the actual construction will use many *duplicates* of each 'gadget' that implements an access or a counter bit. (The number of duplicates will be polylogarithmic, so this will not blow up the construction by too much.)

We end this section with an analysis of the number of steps performed by RANDCOUNT* and by the claim that the expected number of steps performed by RANDCOUNT is equal to the number of steps performed by RANDCOUNT*.

Let $\mathcal{S}(n)$ be the set of permutations on $[n]$. Let $f_n(N, \varphi)$ be the expected number of steps performed by a call RANDCOUNT$^*(N, \mathcal{AB}_n, \varphi)$. If $N = \emptyset$, then $f_n(N, \varphi) := 1$. Otherwise, if $N \neq \emptyset$, we have

$$f_n(N, \varphi) := f_n(N \setminus \{i\}, \varphi) + f_n(N \cap [i], \varphi) \quad \text{where} \quad i = \operatorname*{argmin}_{j \in N} \varphi(j).$$

We let $f_n(N)$ be the expected number of steps performed by a call RANDCOUNT$^*(N, \mathcal{AB}_n, \varphi)$, where φ is random. Clearly

$$f_n(N) := \frac{1}{|\mathcal{S}(n)|} \sum_{\varphi \in \mathcal{S}(n)} f_n(N, \varphi)$$

The proof of the following lemma, which states that the expected number of steps performed by the original and modified randomized counters is the same, is similar to the proof of Lemma 4.1 and is given in Appendix A.

Lemma 5.2 *Let $n \in \mathbb{N}$. Then $f_n([n]) = g(n)$.*

6 Complete description of lower bound games

We call a tuple $\zeta = (n, r, l, s, t)$ with $n, r, l, s, t > 0$ a *signature*. The family of *lower bound games* G_ζ w.r.t. a signature $\zeta = (n, r, l, s, t)$ are 1-sink parity games $G_\zeta = (V_\zeta^0, V_\zeta^1, E_\zeta, \Omega_\zeta)$ defined as follows.

$$V_\zeta^0 := \{a_{i,j,k}\}_{[n]\times[l]\times[s]} \cup \{b_{i,j}\}_{[n]\times[t]} \cup \{c_i\}_{[n]} \cup \{d_i\}_{[n]}$$
$$V_\zeta^1 := \{\mathrm{A}_{i,j}\}_{[n]\times[l]} \cup \{\mathrm{B}_i\}_{[n]} \cup \{\mathrm{D}_{i,j}\}_{[n]\times[r]} \cup \{\mathrm{G}_{i,j,k}\}_{[n]\times[l]\times[r]} \cup \{\mathrm{H}_{i,j}\}_{[n]\times[r]} \cup \{\mathrm{T}\}$$

where $\{a_{i,j,k}\}_{[n]\times[l]\times[s]}$ is a shorthand for $\{a_{i,j,k} \mid 0 \le i < n, 0 \le j < l, 0 \le k < s\}$, etc. Figure 3 defines the edge set E_ζ and the priority assignment function Ω, where $\mathrm{D}_{i,*}$ is a shorthand for the set $\{\mathrm{D}_{i,j} \mid 0 \le j < r\}$. (The ordering of the vertices in Figure 3 is related to their function.)

Node V_ζ	Successors in E_ζ	Priority Ω_ζ
$b_{i,j}$	$\mathrm{B}_i, \mathrm{D}_{i,*}$	2
B_i	$c_i, b_{i,*}$	4
$\mathrm{H}_{i,j}$	B_i	5
$a_{i,j,k}$	$\mathrm{A}_{i,j}, \mathrm{D}_{i,*}$	2
$\mathrm{A}_{i,j}$	$\mathrm{H}_{i,0}, b_{i,j,*}$	2
$\mathrm{G}_{i,j,k}$	$\mathrm{A}_{i,j}$	4
c_{n-1}	T	$2n+4$
c_i	$\mathrm{G}_{i+1,*,*}$	$2i+6$
d_i	$\mathrm{D}_{i,*}, \mathrm{H}_{i,*}$	2
$\mathrm{D}_{n-1,j}$	T	2
$\mathrm{D}_{i,j}$	d_{i+1}	2
T	T	1

Figure 3: Edges and Priorities of G_ζ

An example of a lower bound game with three bits, i.e., $n = 3$, is shown in Figure 4. Nodes owned by player 0 are drawn as circles and nodes owned by player 1 are drawn as rectangles. Structures that are to be duplicated k times are encircled by dotted rectangles with label k.

Counting the number of vertices and edges, we get the following lemma.

Lemma 6.1 *The parity game $G_\zeta = (V_\zeta^0, V_\zeta^1, E_\zeta, \Omega_\zeta)$, with $\zeta = (n, r, l, s, t)$, has:*

- $|V_\zeta^0| = n(2 + t + ls)$.
- $|E_0(G_\zeta)| = n(r(2 + t + l(s + 1)) + t + ls)$.
- $|V_\zeta^1| = n(1 + l + (l + 2)r) + 1$.
- $|E_1(G_\zeta)| = n(2r(l + 1) + ls + 1) + 1$.

Edges originating at vertices controlled by player 0 can generally be allocated to one of three categories. *Counting bit edges* of the form $(b_{i,j}, \mathrm{B}_i)$, *access bit edges* of the form $(a_{i,j,k}, \mathrm{A}_{i,j})$, and *reset edges*. It will be convenient to group copies of reset edges together in sets. Thus, we define:

$$(b_{i,j}, \mathrm{D}_{i,*}) := \{(b_{i,j}, \mathrm{D}_{i,m}) \mid m \in [r]\},$$
$$(a_{i,j,k}, \mathrm{D}_{i,*}) := \{(a_{i,j,k}, \mathrm{D}_{i,m}) \mid m \in [r]\},$$
$$(c_i, \mathrm{G}_{i+1,j,*}) := \{(c_i, \mathrm{G}_{i+1,j,m}) \mid m \in [r]\},$$
$$(d_i, \mathrm{D}_{i,*}) := \{(d_i, \mathrm{D}_{i,m}) \mid m \in [r]\},$$
$$(d_i, \mathrm{H}_{i,*}) := \{(d_i, \mathrm{H}_{i,m}) \mid m \in [r]\}.$$

The *set of reset copies* $R_\zeta(F)$, for some set $F \subseteq E_0(G_\zeta)$, is the set of sets of reset edges that remain in F, including possibly empty sets. We define $R_\zeta(F)$ as follows:

$$\begin{aligned} R_\zeta(F) :=& \{F \cap (b_{i,j}, \mathrm{D}_{i,*})\}_{[n]\times[t]} \ \cup \ \{F \cap (a_{i,j,k}, \mathrm{D}_{i,*})\}_{[n]\times[l]\times[s]} \ \cup \\ & \{F \cap (c_i, \mathrm{G}_{i+1,j,*})\}_{[n]\times[l]} \ \cup \ \{F \cap (d_i, \mathrm{D}_{i,*})\}_{[n]} \ \cup \ \{F \cap (d_i, \mathrm{H}_{i,*})\}_{[n]}. \end{aligned}$$

The *reset budget* of F, $\mathtt{res}_\zeta(F)$, which counts the minimal number of copies of reset edges that is contained in F, is defined by $\mathtt{res}_\zeta(F) := \min\{|A| \mid A \in R_\zeta(F)\}$.

The *set of complete bits* of F, $\mathtt{Com}_\zeta(F)$, which collects all counting bits and all copies of access bits that are completely contained in F, is defined by

$$\begin{aligned} \mathtt{Com}_\zeta(F) :=& \{\beta_i \mid i \in [n] \text{ and } (b_{i,j}, \mathrm{B}_i) \in F \text{ for every } j \in [t]\} \cup \\ & \{\alpha_{i,j} \mid i \in [n],\ j \in [l] \text{ and } (a_{i,j,k}, \mathrm{A}_{i,j}) \in F \text{ for every } k \in [s]\} \end{aligned}$$

We are often only interested in whether a counting bit or whether any copy of an access bit is completey contained in F, hence we define the *abstract representation* of F, $\mathtt{Abs}_\zeta(F)$, by

$$\mathtt{Abs}_\zeta(F) := \{\beta_i \in \mathtt{Com}_\zeta(F)\} \cup \{\alpha_i \mid i \in [n],\ \alpha_{i,j} \in \mathtt{Com}_\zeta(F) \text{ for some } j \in [l]\}$$

The *set of bits* of F, $\mathtt{Bit}_\zeta(F)$, is defined by $\mathtt{Bit}_\zeta(F) := \{i < n \mid \beta_i \in \mathtt{Abs}_\zeta(F)\}$.

Recall that our modified, augmented, randomized bit-counter is based on the invariance that lower counting (and access) bits can only be set iff for every higher bit i it holds that if β_i is enabled, some copy of α_i must be enabled as well. Therefore, given a set of counting and access bits, we are interested in the lowest counting bit which can be set, assuming that all higher bits contained in the given set are used. Formally, the *lowest set bit* of F, $\mathtt{low}_\zeta(F)$, is defined by

$$\mathtt{low}_\zeta(F) := \max(\{0\} \cup \{i < n \mid \beta_i \in \mathtt{Abs}_\zeta(F) \text{ and } \alpha_i \notin \mathtt{Abs}_\zeta(F)\})$$

Next, we describe the explicit structure of the optimal player 0 strategy given a subset of edges as well as the corresponding optimal player 1 counterstrategy. We assume that for every reset edge, there is at least one copy contained in the given set of edges.

Lemma 6.2 *Let $\zeta = (n, r, l, s, t)$ be a signature and $F \subseteq E_0(G_\zeta)$ s.t. $\mathtt{res}_\zeta(F) > 0$. The strategies $\sigma_{G,F}$ and $\tau_{\sigma_{G,F}}$ are of the following form (we omit nodes with out-degree 1).*

$$\sigma_{G,F} \equiv \begin{cases} b_{i,j} \mapsto & \begin{cases} \mathrm{B}_i & \text{if } (b_{i,j}, \mathrm{B}_i) \in F \text{ and } \mathtt{low}_\zeta(F) \le i \\ \mathrm{D}_{i,*} & \text{otherwise} \end{cases} \\ a_{i,j,k} \mapsto & \begin{cases} \mathrm{A}_{i,j} & \text{if } (a_{i,j,k}, \mathrm{A}_{i,j}) \in F,\ \mathtt{low}_\zeta(F) \le i \text{ and } \beta_i \in \mathtt{Abs}_\zeta(F) \\ \mathrm{D}_{i,*} & \text{otherwise} \end{cases} \\ d_i \mapsto & \begin{cases} \mathrm{H}_{i,*} & \text{if } \mathtt{low}_\zeta(F) \le i \text{ and } \beta_i \in \mathtt{Abs}_\zeta(F) \\ \mathrm{D}_{i,*} & \text{otherwise} \end{cases} \\ c_{i<n-1} \mapsto & \begin{cases} \mathrm{G}_{i+1,j,*} & \text{if } \beta_{i+1}, \alpha_{i+1} \in \mathtt{Abs}_\zeta(F) \text{ and } \alpha_{i+1,j} \in \mathtt{Com}_\zeta(F) \text{ for some } j < l \\ \mathrm{G}_{i+1,*,*} & \text{otherwise} \end{cases} \end{cases}$$

$$\tau_{\sigma_{G,F}} \equiv \begin{cases} \mathrm{B}_i \mapsto & \begin{cases} b_{i,j} & \text{if } \mathtt{low}_\zeta(F) \le i \text{ and } (b_{i,j}, \mathrm{B}_i) \notin F \text{ for some } j < t \\ c_i & \text{otherwise} \end{cases} \\ \mathrm{A}_{i,j} \mapsto & \begin{cases} a_{i,j,k} & \text{if } \mathtt{low}_\zeta(F) \le i,\ \beta_i \in \mathtt{Abs}_\zeta(F) \text{ and } (a_{i,j,k}, \mathrm{A}_{i,j}) \notin F \text{ for some } k < s \\ \mathrm{H}_{i,0} & \text{otherwise} \end{cases} \end{cases}$$

Additionally, it holds for every $i < n$ that $val_{\sigma_{G,F}}(\mathrm{D}_{i,*}) < val_{\sigma_{G,F}}(c_i)$ iff $\mathtt{low}_\zeta(F) \le i$.

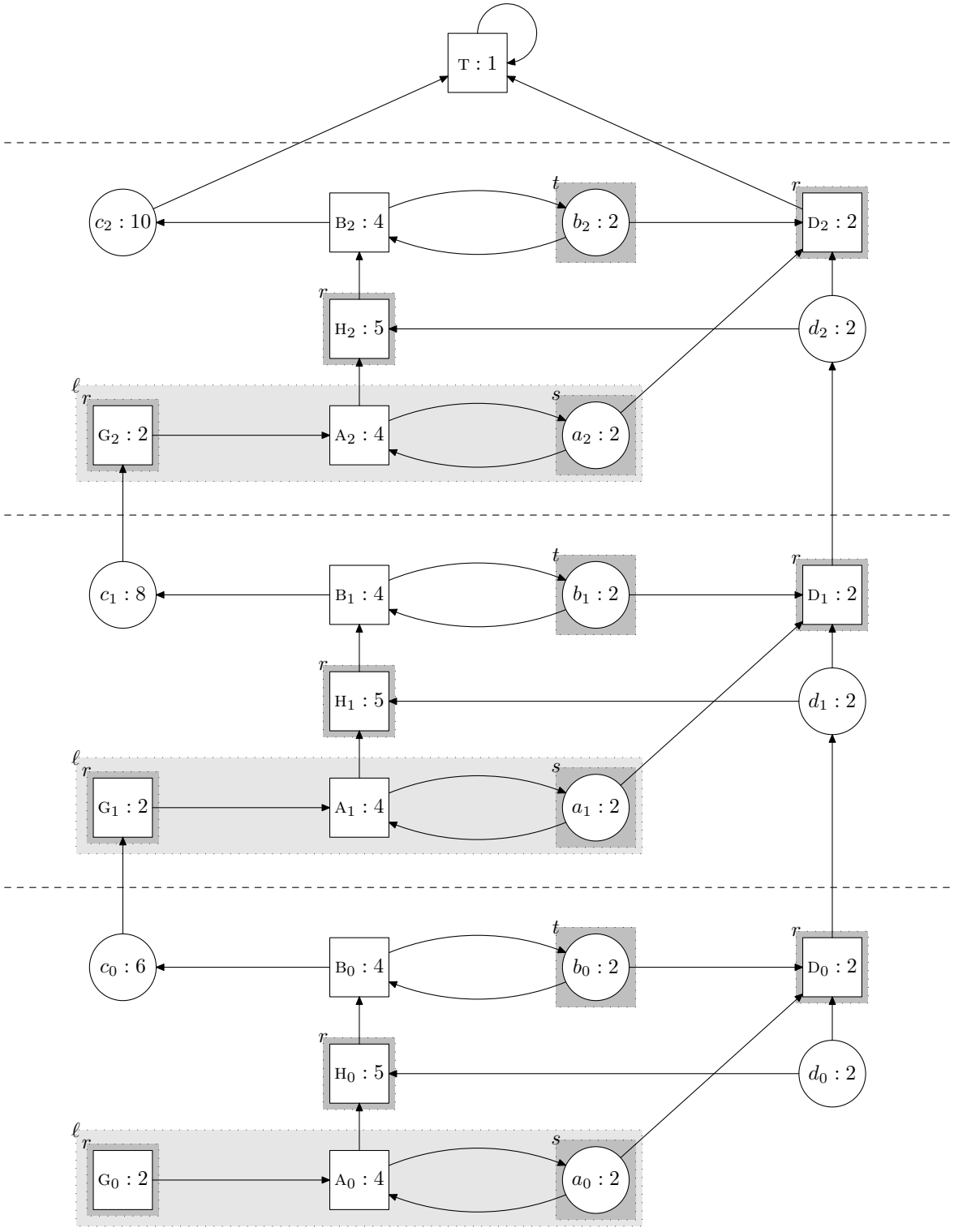

Figure 4: G_ζ with $\zeta = (3, r, \ell, s, t)$. Subgraphs corresponding to separate counting bits and access bits have been separated by dashed lines. Shaded rectangles indicate duplication with the duplication factor shown in the upper-left corner.

7 Lower bound proof

Our general strategy is to show that the modified RandomFacet essentially requires the same number of iterations to solve G_ζ as the modified, augmented, randomized bit-counter. In order to prove this claim, we require a given index function to satisfy some good properties. We start by defining these and show afterwards that a random index functions fulfills these properties with high probability.

Let $\zeta = (n, r, l, s, t)$ be a signature and $\texttt{ind} \in \mathcal{I}(G_\zeta)$ be an index function. Let

$$\begin{aligned}
\texttt{first}_{\texttt{ind}}(b_i) &= \min\{\texttt{ind}(b_{i,j}, \mathrm{B}_i) \mid j \in [t]\} \\
\texttt{first}_{\texttt{ind}}(a_{i,j}) &= \min\{\texttt{ind}(a_{i,j,k}, \mathrm{A}_{i,j}) \mid k \in [s]\} \\
\texttt{last}(\texttt{ind}) &= \max\{\texttt{first}_{\texttt{ind}}(a_{i,j}) \mid i \in [n], j \in [l]\}.
\end{aligned}$$

Let $F \subseteq E_0(G)$ and $\texttt{ind} \in \mathcal{I}(G_\zeta)$. We say that F *is good w.r.t.* ind iff it satisfies the following two properties:

1. $\forall \beta_i \in \texttt{Com}_\zeta(F)\ \exists \alpha_{i,j} \in \texttt{Com}_\zeta(F) :\ \ \texttt{first}_{\texttt{ind}}(b_i) < \texttt{first}_{\texttt{ind}}(a_{i,j})$.
2. $\texttt{res}_\zeta(\{e \in F \mid \texttt{ind}(e) > \texttt{last}(\texttt{ind})\}) > 0$.

Essentially, in order to be a good set, it must be the case that after the first edge of every completely contained counting bit, there is at least one copy of an access bit structure completely contained and also, that after the first edge of every counting and access bit structure, there is at least one copy of every reset edge contained in the given set.

We say that an index function $\texttt{ind} \in \mathcal{I}(G_\zeta)$ is *good* iff $E_0(G)$ is good w.r.t. ind, in which case the first property can also be formulated as $\forall i \in [n]\ \exists j \in [l] :\ \texttt{first}_{\texttt{ind}}(b_i) < \texttt{first}_{\texttt{ind}}(a_{i,j})$.

Note that we are particularly interested in good index functions instead of good sets w.r.t. a given index function. The latter definition is only used in an inductive way: starting with the full set of edges and a good index function, the smaller sets that we consider in a run of the RandomFacet algorithm will be good as well.

Lemma 7.1 *Let $\zeta = (n, r, l, s, t)$. Let $\texttt{ind} \in \mathcal{I}(G_\zeta)$ be chosen uniformly at random. Then with probability $p_\zeta \geq 1 - n\frac{l!\cdot q!}{(l+q)!} - n^2 l(t + ls + l + 2)\frac{s!\cdot r!}{(s+r)!}$,* ind *is good, where $q = \lfloor \frac{t}{s} \rfloor$.*

Let σ be a player 0 strategy. We say that σ is a *good initial strategy* iff $\texttt{Com}_\zeta(E_0(G) \setminus \sigma) = \texttt{Com}_\zeta(E_0(G))$. In other words: an initial strategy is good iff it does not use any access bit or counting bit edges.

The following definitions will be a bit technical; their purpose is to establish a relation between the current state of the RandomFacet algorithm when applied to G_ζ and the current state of the modified, augmented, randomized bit-counter.

First, we define a certain subclass of very well-structured player 0 strategies. Let $F \subseteq E_0(G)$, $\sigma \subseteq F$ be a player 0 strategy and $i \leq n$. We say that σ is an *i-boundary strategy w.r.t. F* iff the following properties hold.

1. $\forall j < n\ \forall k < t\ :\ \ \sigma(b_{j,k}) = \mathrm{B}_j \iff (j \geq i \text{ and } (b_{j,k}, \mathrm{B}_j) \in F)$.
2. $\forall j < n\ \forall m < l\ \forall k < s\ :\ \ \sigma(a_{j,m,k}) = \mathrm{A}_{j,m} \iff (j \geq i \text{ and } (a_{j,m,k}, \mathrm{A}_{j,m}) \in F)$.

In other words, an i-boundary strategy corresponds to the set of the bit-counter in which all counting and all access bits contained in F with an index greater or equal to i are set.

Second, we relate a given index function for G_ζ to an index function for the modified, augmented, randomized bit-counter. Let $\zeta = (n, r, l, s, t)$ be a signature, $F \subseteq E_0(G_\zeta)$, $\sigma \subseteq F$ be a player 0 strategy and $\mathtt{ind} \in \mathcal{I}(G)$ be an index function. The *induced index w.r.t.* $\mathtt{ind} \in \mathcal{I}(G_\zeta)$ *and* ζ is defined by

$$\varphi_\zeta^{\mathtt{ind}} : i \mapsto |\{j < n \mid \mathtt{first}_{\mathtt{ind}}(b_j) < \mathtt{first}_{\mathtt{ind}}(b_i)\}|$$

Third, we define the actual choice function $\mathtt{cho}_\zeta$ that will be used in the analysis of the algorithm.

If possible, $\mathtt{cho}_\zeta$ selects the first counting bit edge, denoted here by $b_{F,\sigma,\mathtt{ind}} \in F \setminus \sigma$, w.r.t. $\mathtt{ind}$ for which all edges of the corresponding counting bit are contained in $F \setminus \sigma$. Let us assume that $b_{F,\sigma,\mathtt{ind}}$ is part of the i'th counting bit. This corresponds to excluding the i'th counting bit, counting recursively, and then enabling it again. After excluding the i'th counting bit, the optimal strategy $\sigma_{G,F\setminus\{b_{F,\sigma,\mathtt{ind}}\}}$ w.r.t. the remaining edges contains all other counting bits as well as all other access bits. Let $\sigma' = \sigma_{G,F\setminus\{b_{F,\sigma,\mathtt{ind}}\}}[b_{F,\sigma,\mathtt{ind}}]$. The only remaining bit structures in $F \setminus \sigma'$, hence, are some copies of the access bit structures of index i. In this case, the choice function $\mathtt{cho}_\zeta$ selects the first access bit edge, denoted here by $a_{F,\sigma',\mathtt{ind}} \in F \setminus \sigma'$, of the last copy of the access bits associated with counting bit i contained in F. Excluding $a_{F,\sigma',\mathtt{ind}}$, the optimal strategy w.r.t. the remaining edges corresponds to the state in which all bits, counting bits as well as access bits, greater or equal to i are set and all lower bits are reset. Hence, when including $a_{F,\sigma',\mathtt{ind}}$ again the resulting strategy is an i-boundary strategy s.t. all lower bits can be set once again.

Formally, if $\mathtt{Bit}_\zeta(F \setminus \sigma) \neq \emptyset$, we define:

$$b_{F,\sigma,\mathtt{ind}} = \operatorname*{argmin}_{e \in \{(b_{i,j},\mathrm{B}_i) \in F\setminus\sigma \;\mid\; \beta_i \in \mathtt{Com}_\zeta(F\setminus\sigma)\}} \mathtt{ind}(e).$$

Also, if $\mathtt{Bit}_\zeta(F \setminus \sigma) = \emptyset$ and $\mathtt{Abs}_\zeta(F \setminus \sigma) \neq \emptyset$, we define:

$$a_{F,\sigma,\mathtt{ind}} = \operatorname*{argmax}_{e \in \{(a_{i,j,k},\mathrm{A}_{i,j}) \in F\setminus\sigma \;\mid\; \mathtt{Com}_\zeta(\{e' \in F\setminus\sigma \mid \mathtt{ind}(a_{i,j,k},\mathrm{A}_{i,j}) \leq \mathtt{ind}(e')\}) \neq \emptyset\}} \mathtt{ind}(e)$$

Finally, the *canonic choice function w.r.t.* ζ is defined as follows. Let $F \setminus \sigma \subseteq E_0(G)$ and $\mathtt{ind}$ be an index function.

$$\mathtt{cho}_\zeta(F \setminus \sigma, \mathtt{ind}) = \begin{cases} b_{F,\sigma,\mathtt{ind}} & \text{if } \mathtt{Bit}_\zeta(F \setminus \sigma) \neq \emptyset \\ a_{F,\sigma,\mathtt{ind}} & \text{if } \mathtt{Bit}_\zeta(F \setminus \sigma) = \emptyset \text{ and } \mathtt{Abs}_\zeta(F \setminus \sigma) \neq \emptyset \\ \bot & \text{otherwise} \end{cases}$$

Next, we establish the formal correspondence between the modified RandomFacet algorithm on G_ζ and the modified, augmented, randomized bit-counter.

Lemma 7.2 *Let $\zeta = (n, r, l, s, t)$ be a signature and $\mathtt{ind} \in \mathcal{I}(G_\zeta)$ be a good index function. Let $F \subseteq E_0(G_\zeta)$ be good w.r.t. $\mathtt{ind}$, $i \in \mathtt{Bit}_\zeta(F) \cup \{n\}$ and $\sigma \subseteq F$ be an i-boundary strategy w.r.t. F. Then*

$$\mathbb{E}^\dagger_{G_\zeta}(F, \sigma, \mathtt{cho}_\zeta, \mathtt{ind}) \geq f_n(\mathtt{Bit}_\zeta(F) \cap [i], \varphi_\zeta^{\mathtt{ind}})$$

It is easy to derive that, starting with a good index function, the full set of edges and a good initial strategy, Lemma 7.2 can be applied.

Corollary 7.3 *Let $\zeta = (n, r, l, s, t)$ be a signature, $\mathtt{ind} \in \mathcal{I}(G)$ be a good index function and σ be a good initial strategy. Then $\mathbb{E}^\dagger_{G_\zeta}(E_0(G_\zeta), \sigma, \mathtt{cho}_\zeta, \mathtt{ind}) \geq f_n([n], \varphi_\zeta^{\mathtt{ind}})$.*

We conclude that, given a good initial strategy, the expected number of iterations that the unmodified RandomFacet algorithm requires to solve G_ζ can be expressed in terms of p_ζ and $g(n)$.

Lemma 7.4 *Let ζ be a signature and σ be a good initial strategy. Then $\mathbb{E}(E_0(G_\zeta), \sigma) \geq p_\zeta \cdot g(n)$.*

By a close analysis of the probability p_ζ, we derive the final theorem of this paper.

Theorem 7.5 *The worst-case expected running time of the* RandomFacet *algorithm for n-state parity games is at least $2^{\Omega(\sqrt{n}/\log n)}$.*

By Theorem 3.2 this result extends to Mean Payoff Games, Discounted Payoff Games and Turn-based Stochastic Games, and we get the following corollary.

Corollary 7.6 *The worst-case expected running time of the* RandomFacet *algorithm for n-state Mean Payoff Games, Discounted Payoff Games and Turn-based Stochastic Games is at least $2^{\Omega(\sqrt{n}/\log n)}$.*

8 Concluding remarks

We constructed explicit parity games on n vertices on which the expected running time of the RandomFacet algorithm is $2^{\tilde{O}(\sqrt{n})}$, almost matching the upper bound of Matoušek, Sharir and Welzl [MSW96]. This dashes the hope that the RandomFacet algorithm could be used to solve parity games, or even deterministic or stochastic mean payoff games, in polynomial time.

Our parity games also supply the first *explicit* instances on which the expected running time of the RandomFacet algorithm is not polynomial. Matoušek [Mat94] previously constructed a family of instances such that the expected running time of the RandomFacet algorithm is not polynomial when run on a *random* instance from this family. One advantage of Matoušek's non-explicit construction is that his instances correspond to *Acyclic Unique Sink Orientations* (AUSOs) of n-cubes, while our explicit instances do not. (For information on AUSOs, see [SW01],[Gär02],[GS06].) It would be interesting to see whether an extension of the method used here could be used to construct an explicit AUSO of an n-cube on which RandomFacet takes more than polynomial time to find the sink.

Another natural randomized algorithm for solving parity games and other games is the RandomEdge algorithm in which a random improving switch is performed at each step. Matoušek and Szabó constructed abstract instances on which the expected running time of this algorithm is $2^{\Omega(n^{1/3})}$. Are there parity games on which the algorithm behaves as badly?

The most interesting open problem is, of course, whether there is a polynomial time algorithm for solving parity games.

References

[AM09] D. Andersson and P.B. Miltersen. The complexity of solving stochastic games on graphs. In *Proc. of 20th ISAAC*, pages 112–121, 2009.

[BSV03] H. Björklund, S. Sandberg, and S. Vorobyov. A discrete subexponential algorithm for parity games. In *Proc. of 20th STACS*, pages 663–674, 2003.

[BV05] H. Björklund and S. Vorobyov. Combinatorial structure and randomized subexponential algorithms for infinite games. *Theoretical Computer Science*, 349(3):347–360, 2005.

[BV07] H. Björklund and S. Vorobyov. A combinatorial strongly subexponential strategy improvement algorithm for mean payoff games. *Discrete Applied Mathematics*, 155(2):210–229, 2007.

[Con92] A. Condon. The complexity of stochastic games. *Information and Computation*, 96:203–224, 1992.

[EJ91] E.A. Emerson and C. Jutla. Tree automata, μ-calculus and determinacy. In *Proceedings of the 32nd FOCS*, pages 368–377. IEEE Computer Society Press, 1991.

[EJS93] E. A. Emerson, C. S. Jutla, and A. P. Sistla. On model-checking for fragments of μ-calculus. In *International Conference on Computer-Aided Verification, CAV'93*, volume 697 of *LNCS*, pages 385–396, 1993.

[EM79] A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 8:109–113, 1979.

[Fea10] J. Fearnley. Exponential lower bounds for policy iteration. *CoRR*, abs/1003.3418, 2010.

[FL09] O. Friedmann and M. Lange. Solving parity games in practice. In *Proceedings of the 7th ATVA*, pages 182–196, 2009.

[Fri09] O. Friedmann. An exponential lower bound for the parity game strategy improvement algorithm as we know it. In *Proc. of 24th LICS*, pages 145–156, 2009.

[Fri10] O. Friedmann. An exponential lower bound for the latest deterministic strategy iteration algorithms. *Selected Papers of the Conference "Logic in Computer Science 2009" (to appear)*, 2010. A preprint available from `http://www.tcs.ifi.lmu.de/~friedman`.

[FS09] P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.

[Gär02] B. Gärtner. The random-facet simplex algorithm on combinatorial cubes. *Random Structures and Algorithms*, 20(3):353–381, 2002.

[GKK88] V.A. Gurvich, A.V. Karzanov, and L.G. Khachiyan. Cyclic games and an algorithm to find minimax cycle means in directed graphs. *USSR Computational Mathematics and Mathematical Physics*, 28:85–91, 1988.

[Gol95] M. Goldwasser. A survey of linear programming in randomized subexponential time. *SIGACT News*, 26(2):96–104, 1995.

[GS06] B. Gärtner and I. Schurr. Linear programming and unique sink orientations. In *Proc. 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 749–757, 2006.

[GTW02] E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logics, and Infinite Games. A Guide to Current Research*, volume 2500 of *LNCS*. Springer, 2002.

[Hal07] N. Halman. Simple stochastic games, parity games, mean payoff games and discounted payoff games are all LP-type problems. *Algorithmica*, 49(1):37–50, 2007.

[HK66] A. Hoffman and R. Karp. On nonterminating stochastic games. *Management Science*, 12:359–370, 1966.

[How60] R.A. Howard. *Dynamic programming and Markov processes*. MIT Press, 1960.

[JPZ08] M. Jurdziński, M. Paterson, and U. Zwick. A deterministic subexponential algorithm for solving parity games. *SIAM Journal on Computing*, 38(4):1519–1532, 2008.

[Jur98] M. Jurdziński. Deciding the winner in parity games is in UP $\cap$ co-UP. *Information Processing Letters*, 68(3):119–124, 1998.

[Jur00] M. Jurdziński. Small progress measures for solving parity games. In *Proc. of 17th STACS*, pages 290–301, 2000.

[Kal92] G. Kalai. A subexponential randomized simplex algorithm (extended abstract). In *Proc. of 24th STOC*, pages 475–482, 1992.

[Kal97] G. Kalai. Linear programming, the simplex algorithm and simple polytopes. *Mathematical Programming*, 79:217–233, 1997.

[Lud95] W. Ludwig. A subexponential randomized algorithm for the simple stochastic game problem. *Information and Computation*, 117(1):151–155, 1995.

[Mat94] J. Matoušek. Lower bounds for a subexponential optimization algorithm. *Random Structures and Algorithms*, 5(4):591–608, 1994.

[MS06] J. Matoušek and T. Szabó. RANDOM EDGE can be exponential on abstract cubes. *Advances in Mathematics*, 204(1):262–277, 2006.

[MSW96] J. Matoušek, M. Sharir, and E. Welzl. A subexponential bound for linear programming. *Algorithmica*, 16(4-5):498–516, 1996.

[MTZ10] O. Madani, M. Thorup, and U. Zwick. Discounted deterministic Markov decision processes and discounted all-pairs shortest paths. *ACM Transcations on Algorithms*, 6(2):1–25, 2010.

[Pur95] A. Puri. *Theory of Hybrid Systems and Discrete Event Simulation*. PhD thesis, University of California, Berkeley, 1995.

[PV01] V. Petersson and S.G. Vorobyov. A randomized subexponential algorithm for parity games. *Nord. J. Comput.*, 8(3):324–345, 2001.

[Sch08] S. Schewe. An optimal strategy improvement algorithm for solving parity and payoff games. In *Proc. of 17th CSL*, pages 369–384, 2008.

[Sti95] C. Stirling. Local model checking games. In *Proceedings of the 6th International Conference on Concurrency Theory (CONCUR'95)*, volume 962 of *LNCS*, pages 1–11. Springer, 1995.

[SW01] T. Szabó and E. Welzl. Unique sink orientations of cubes. In *Proc. of 42th FOCS*, pages 547–555, 2001.

[VJ00] J. Vöge and M. Jurdziński. A discrete strategy improvement algorithm for solving parity games (Extended abstract). In *International Conference on Computer-Aided Verification, CAV 2000*, volume 1855 of *LNCS*, pages 202–215, 2000.

[Vög00] J. Vöge. *Strategiesynthese für Paritätsspiele auf endlichen Graphen*. PhD thesis, University of Aachen, 2000.

[Zie98] W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200:135–183, 1998.

[ZP96] U. Zwick and M.S. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158(1–2):343–359, 1996.

A Appendix

Lemma 4.1. Proof: By lexicographic induction on $(|F|, val_\sigma)$. For $|F| = |\sigma|$ this clearly holds true. For the induction step, let now $|F| > |\sigma|$ and $e_i := \text{argmin}_{e' \in F \setminus \sigma} i(e')$ for an index function i. Then

$$\begin{aligned}
\mathbb{E}^*_G(F,\sigma) =& \frac{1}{|\mathcal{I}(G)|} \sum_{i \in \mathcal{I}(G)} \mathbb{E}^*_G(F,\sigma,i) \\
=& \frac{1}{|\mathcal{I}(G)|} \sum_{i \in \mathcal{I}(G)} \mathbb{E}^*_G(F \setminus \{e_i\},\sigma,i) + \mathbb{1}_{e_i \in \sigma_{G,F}} \cdot \mathbb{E}^*_G(F, \sigma_{G,F\setminus\{e_i\}}[e_i], i) \\
=& \frac{1}{|F \setminus \sigma|} \sum_{e \in F \setminus \sigma} \frac{1}{|\mathcal{I}(G \setminus \{e\})|} \sum_{i \in \mathcal{I}(G\setminus\{e\})} \mathbb{E}^*_{G\setminus\{e\}}(F \setminus \{e\},\sigma,i) + \\
& \frac{1}{|F \setminus \sigma|} \sum_{e \in F \setminus \sigma} \mathbb{1}_{e \in \sigma_{G,F}} \cdot \frac{1}{|\mathcal{I}(G)|} \sum_{i \in \mathcal{I}(G)} \mathbb{E}^*_G(F, \sigma_{G,F\setminus\{e\}}[e], i) \\
\overset{IH}{=}& \frac{1}{|F \setminus \sigma|} \sum_{e \in F \setminus \sigma} \mathbb{E}_{G\setminus\{e\}}(F \setminus \{e\},\sigma) + \\
& \frac{1}{|F \setminus \sigma|} \sum_{e \in F \setminus \sigma} \mathbb{1}_{e \in \sigma_{G,F}} \cdot \mathbb{E}_G(F, \sigma_{G,F\setminus\{e\}}[e]) \\
=& \mathbb{E}_G(F,\sigma)
\end{aligned}$$

□

Lemma 4.2. Proof: By induction on $|H|$. The induction basis $|H| = 0$ obviously holds true. For the induction step let now $|H| > 0$. Let $e = \text{argmin}_{e' \in F \setminus \sigma} \texttt{ind}(e')$ and note that $e \in H$. By definition, it holds that

$$\mathbb{E}^*_G(F,\sigma,\texttt{ind}) \geq \mathbb{E}^*_G(F \setminus \{e\},\sigma,\texttt{ind})$$

Obviously, the induction hypothesis is applicable to $F \setminus \{e\}$ and $H \setminus \{e\}$, hence we get

$$\mathbb{E}^*_G(F \setminus \{e\},\sigma,\texttt{ind}) \geq \mathbb{E}^*_G(F \setminus \{e\} \setminus (H \setminus \{e\}),\sigma,\texttt{ind})$$

Since $F \setminus \{e\} \setminus (H \setminus \{e\}) = F \setminus H$, the claim follows. □

Lemma 4.3. Proof: Let $\texttt{ind} \in \mathcal{I}(G)$ and $\texttt{cho} \in \mathcal{C}(G)$. It suffices to show that $\mathbb{E}^*_G(F,\sigma,\texttt{ind}) \geq \mathbb{E}^\dagger_G(F,\sigma,\texttt{cho},\texttt{ind})$ by lexicographic induction on $(|F|, \mathbb{E}^*_G(F,\sigma))$ with $|F| \geq |\sigma|$.
The basis $|F| = |\sigma|$ obviously holds true. For the induction step let $|F| > |\sigma|$. Let $e = \texttt{cho}(F \setminus \sigma, \texttt{ind})$. If $e = \perp$ the claim follows as well; assume therefore that $e \neq \perp$ and let $H = \{e' \in F \setminus \sigma \mid \texttt{ind}(e') < \texttt{ind}(e)\}$. Then Lemma 4.2 tells us that $\mathbb{E}^*_G(F,\sigma,\texttt{ind}) \geq \mathbb{E}^*_G(F \setminus H,\sigma,\texttt{ind})$. By definition it follows that

$$\mathbb{E}^*_G(F \setminus H,\sigma,\texttt{ind}) \geq \mathbb{E}^*_G(F \setminus H \setminus \{e\},\sigma,\texttt{ind}) + \mathbb{1}_{e \in \sigma_{G,F\setminus H}} \cdot \mathbb{E}^*_G(F \setminus H, \sigma_{G,F\setminus H\setminus\{e\}}[e], \texttt{ind})$$

If follows by induction hypothesis that $\mathbb{E}^*_G(F \setminus H \setminus \{e\},\sigma,\texttt{ind}) \geq \mathbb{E}^\dagger_G(F \setminus H \setminus \{e\},\sigma,\texttt{cho},\texttt{ind})$ as well as $\mathbb{E}^*_G(F \setminus H, \sigma_{G,F\setminus H\setminus\{e\}}[e], \texttt{ind}) \geq \mathbb{E}^\dagger_G(F \setminus H, \sigma_{G,F\setminus H\setminus\{e\}}[e], \texttt{cho}, \texttt{ind})$ (assuming $e \in \sigma_{G,F\setminus H}$). Hence, the claim follows. □

Lemma 5.2. Proof: By induction on n. The claim obviously holds true for $n = 0$. Let now $n > 0$. The following holds.

$$
\begin{aligned}
f_n([n]) =& \frac{1}{|\mathcal{S}(n)|} \sum_{\varphi \in \mathcal{S}(n)} f_n([n], \varphi) \\
=& \frac{1}{|\mathcal{S}(n)|} \sum_{\varphi \in \mathcal{S}(n)} f_n([n] \setminus \{\varphi^{-1}(0)\}, \varphi) + f_n([\varphi^{-1}(0)], \varphi) \\
=& \frac{1}{|\mathcal{S}(n-1)|} \sum_{\varphi \in \mathcal{S}(n-1)} f_{n-1}([n-1], \varphi) + \\
& \frac{1}{|\mathcal{S}(n)|} \sum_{\varphi \in \mathcal{S}(n)} f_{\varphi^{-1}(0)}([\varphi^{-1}(0)], \varphi|_{\varphi([\varphi^{-1}(0)])}) \\
=& g(n-1) + \frac{1}{n} \sum_{i=0}^{n-1} \frac{1}{|\mathcal{S}(i)|} \sum_{\varphi \in \mathcal{S}(i)} f_i([i], \varphi) \\
=& g(n-1) + \frac{1}{n} \sum_{i=0}^{n-1} g(i) = g(n)
\end{aligned}
$$

□

If $G = (V_0, V_1, E, \Omega)$ is a parity game and $F \subseteq E_0(G)$, we let $\sigma_{G,F}$ be an optimal strategy of player 0 in the subgame $G_F = (V_0, V_1, F \cup E_1(G), \Omega)$. For strategies σ and τ for players 0 and 1, we let $\pi_{v,\sigma,\tau}$ denote the uniquely determined play that starts in v and conforms to σ and τ. We define the *σ-τ-reachability* relation $\rightsquigarrow_{G,\sigma,\tau}$ as follows. For two nodes $u, v \in V$ let $u \rightsquigarrow_{G,\sigma,\tau} v$ iff v occurs in the play $\pi_{v,\sigma,\tau}$.
The *abstract state* of F, $\mathtt{Sta}_\zeta(F)$, is defined as follows.

$$\mathtt{Sta}_\zeta(F) := \{\beta_i \in \mathtt{Abs}_\zeta(F) \mid i \geq \mathtt{low}_\zeta(F)\} \cup \{\alpha_i \in \mathtt{Abs}_\zeta(F) \mid i \geq \mathtt{low}_\zeta(F) \text{ and } \beta_i \in \mathtt{Abs}_\zeta(F)\}$$

Lemma 6.2. Proof: Let $\zeta = (n, r, l, s, t)$ be a signature and $F \subseteq E_0(G_\zeta)$ s.t. $\mathtt{res}_\zeta(F) > 0$. Let $\sigma := \sigma_{G_\zeta,F}$, $val := val_{G,\sigma}$, $\tau := \tau_{G_\zeta,\sigma}$ and $\rightsquigarrow := \rightsquigarrow_{G_\zeta,\sigma,\tau}$. We prove that the following statements hold for all $i < n$.

(1) If $i < n-1$ it holds, $c_i \rightsquigarrow \mathrm{B}_{i+1}$ iff $\alpha_{i+1} \in \mathtt{Sta}_\zeta(F)$ or $\beta_{i+1} \notin \mathtt{Sta}_\zeta(F)$.

(2) $val(\mathrm{D}_{i,*}) < val(c_i)$ iff $\mathtt{low}_\zeta(F) \leq i$.

(3) For every $j < t$ with $(b_{i,j}, \mathrm{B}_i) \in F$ it holds: $b_{i,j} \rightsquigarrow \mathrm{B}_i$ iff $\mathtt{low}_\zeta(F) \leq i$.

(4) $\mathrm{B}_i \rightsquigarrow c_i$ iff $\beta_i \in \mathtt{Sta}_\zeta(F)$ or $\mathtt{low}_\zeta(F) > i$.

(5) $d_i \rightsquigarrow c_i$ iff $\beta_i \in \mathtt{Sta}_\zeta(F)$.

(6) For every $k < l$ and every $j < s$ with $(a_{i,k,j}, \mathrm{A}_{i,k}) \in F$ it holds: $a_{i,k,j} \rightsquigarrow \mathrm{A}_{i,k}$ iff $\beta_i \in \mathtt{Sta}_\zeta(F)$.

(7) For every $k < l$ it holds: $\mathrm{A}_{i,k} \rightsquigarrow \mathrm{B}_i$ iff $\beta_i \notin \mathtt{Sta}_\zeta(F)$ or $(a_{i,k,j}, \mathrm{A}_{i,k}) \in F$ for every $j < s$.

These statements directly imply that the strategies are of the form as claimed in the lemma. We prove all statements simultaneously by backward induction on $i < n$. Let $i < n$ and assume that all statements hold for all $j > i$ with $j < n$.

(1) Assume that $i < n-1$.

If $c_i \rightsquigarrow \mathrm{B}_{i+1}$, assume that $\beta_{i+1} \in \mathtt{Sta}_\zeta(F)$. We need to show that $\alpha_{i+1} \in \mathtt{Sta}_\zeta(F)$ as well. It follows that there must be some $k < l$ s.t. $\mathrm{A}_{i+1,k} \rightsquigarrow \mathrm{B}_{i+1}$. By IH (7) it holds that $(a_{i+1,k,j}, \mathrm{A}_{i+1,k}) \in F$ for every $j < s$, hence $\alpha_{i+1} \in \mathtt{Sta}_\zeta(F)$.

For the conversion, we assume first that $\beta_{i+1} \notin \mathtt{Sta}_\zeta(F)$. By IH (7) we derive that $\mathrm{A}_{i+1,k} \rightsquigarrow \mathrm{B}_{i+1}$ for every $k < l$, and hence $c_i \rightsquigarrow \mathrm{B}_{i+1}$.

Finally we assume instead that $\alpha_{i+1} \in \mathtt{Sta}_\zeta(F)$, immediately implying that also $\beta_{i+1} \in \mathtt{Sta}_\zeta(F)$. By IH (2) and IH (4) we derive that $val(\mathrm{D}_{i+1,0}) < val(\mathrm{B}_{i+1})$. By IH (7) we conclude that there is a $k < l$ s.t. $\mathrm{A}_{i+1,k} \rightsquigarrow \mathrm{B}_{i+1}$. Hence, $c_{i+1} \rightsquigarrow \mathrm{B}_{i+1}$.

(2) If $i = n - 1$ this claim immediately follows. Let now $i < n - 1$.

If $val(\mathrm{D}_{i,0}) < val(c_i)$, assume by contradiction that $\mathtt{low}_\zeta(F) > i$. If $\mathtt{low}_\zeta(F) = i + 1$, it follows that $\beta_{i+1} \in \mathtt{Sta}_\zeta(F)$ and by IH (7) that $\mathrm{A}_{i+1,k} \not\rightsquigarrow \mathrm{B}_{i+1}$ for every $k < l$. By IH (2) it follows that $val(\mathrm{D}_{i+1,0}) < val(c_{i+1})$, but this now implies that $val(\mathrm{D}_{i,0}) > val(c_i)$ which cannot be the case. Otherwise, if $\mathtt{low}_\zeta(F) > i+1$, it follows by IH (2) that $val(\mathrm{D}_{i+1,0}) > val(c_{i+1})$. As $\beta_{i+1} \notin \mathtt{Sta}_\zeta(F)$, it follows by IH (7) that $\mathrm{A}_{i+1,k} \rightsquigarrow \mathrm{B}_{i+1}$ for every $k < l$. By IH (4) it follows that $\mathrm{B}_{i+1} \rightsquigarrow c_{i+1}$. Concludingly it holds that $val(\mathrm{D}_{i,0}) > val(c_i)$ which cannot be the case.

For the conversion, we assume that $\mathtt{low}_\zeta(F) \leq i$. By IH (2) this implies that $val(\mathrm{D}_{i+1,0}) < val(c_{i+1})$. If $\alpha_{i+1} \in \mathtt{Sta}_\zeta(F)$ it follows by IH (7) that there is some $k < l$ s.t. $\mathrm{A}_{i+1,k} \rightsquigarrow \mathrm{B}_{i+1}$. By IH (4) we conclude that $\mathrm{A}_{i+1,k} \rightsquigarrow c_{i+1}$. By IH (5) we derive that $d_{i+1} \rightsquigarrow c_{i+1}$ as well. Hence, $val(\mathrm{D}_{i,0}) < val(c_i)$. Otherwise, if $\alpha_{i+1} \notin \mathtt{Sta}_\zeta(F)$, implying that also $\beta_{i+1} \notin \mathtt{Sta}_\zeta(F)$, it follows by IH (4) that $\mathrm{B}_{i+1} \not\rightsquigarrow c_{i+1}$. Hence, $val(\mathrm{D}_{i,0}) < val(c_i)$.

(3) Let $j < t$ with $(b_{i,j}, \mathrm{B}_i) \in F$.

If $\tau(\mathrm{B}_i) = c_i$, the following holds. If also $val(\mathrm{D}_{i,0}) < val(c_i)$, it is clearly the case that $val(\mathrm{D}_{i,j}) < val(\mathrm{B}_i)$ for every $j < r$ and hence $\sigma(b_{i,j}) = \mathrm{B}_i$. For the conversion assume that $val(\mathrm{D}_{i,0}) > val(c_i)$, and hence $\sigma(b_{i,j}) \neq \mathrm{B}_i$.

Otherwise, if $\tau(\mathrm{B}_i) \neq c_i$, it must be the case that $val(\mathrm{D}_{i,0}) < val(c_i)$. It follows that $val(\mathrm{D}_{i,j}) < val(\mathrm{B}_i)$ for every $j < r$ and hence $\sigma(b_{i,j}) = \mathrm{B}_i$.

(4) If $val(c_i) < val(\mathrm{D}_{i,0})$, it is obviously the case that $b_{i,j} \rightsquigarrow \mathrm{B}_i$. Otherwise, if $val(c_i) > val(\mathrm{D}_{i,0})$ the following holds. By (3), we know that $b_{i,j} \rightsquigarrow \mathrm{B}_i$ iff $(b_{i,j}, \mathrm{B}_i) \in F$ for every $j < t$. If $\beta_i \in \mathtt{Sta}_\zeta(F)$, the claim immediately follows. For the conversion, assume that $(b_{i,j}, \mathrm{B}_i) \in F$ for every $j < t$. By (2) this implies $\beta_i \in \mathtt{Sta}_\zeta(F)$.

(5) If $d_i \rightsquigarrow c_i$, it follows immediately that $val(c_i) > val(\mathrm{D}_{i,0})$ and $b_{i,j} \rightsquigarrow \mathrm{B}_i$, hence by (4) we conclude that $\beta_i \in \mathtt{Sta}_\zeta(F)$.

For the conversion assume that $\beta_i \in \mathtt{Sta}_\zeta(F)$, hence again by (4) we know that $b_{i,j} \rightsquigarrow \mathrm{B}_i$. Also, $\beta_i \in \mathtt{Sta}_\zeta(F)$ implies that $\mathtt{low}_\zeta(F) \leq i$ and hence we derive by (2) that $val(\mathrm{D}_{i,0}) < val(c_i)$, immediately proving the claim.

(6) Let $k < l$ and $j < s$ s.t. $(a_{i,k,j}, \mathrm{A}_{i,k}) \in F$.

If $a_{i,k,j} \rightsquigarrow \mathrm{A}_{i,k}$, it must be the case that $val(\mathrm{H}_{i,0}) > val(\mathrm{D}_{i,0})$, which particularly implies that $val(c_i) > val(\mathrm{D}_{i,0})$ and $b_{i,j} \rightsquigarrow \mathrm{B}_i$, hence by (4) we conclude that $\beta_i \in \mathtt{Sta}_\zeta(F)$.

For the conversion assume that $\beta_i \in \mathtt{Sta}_\zeta(F)$. Hence by (2) we know that $val(\mathrm{D}_{i,0}) < val(c_i)$. Also by (4) we derive that $\mathrm{B}_i \rightsquigarrow c_i$. Concludingly it follows that $val(\mathrm{A}_{i,k}) > val(\mathrm{D}_{i,0})$ and hence $a_{i,k,j} \rightsquigarrow \mathrm{A}_{i,k}$.

(7) Let $k < l$.

If $\mathrm{A}_{i,k} \rightsquigarrow \mathrm{B}_i$, assume that $\beta_i \in \mathtt{Sta}_\zeta(F)$. We need to show that $(a_{i,k,j}, \mathrm{A}_{i,k}) \in F$ for every $j < s$. It follows by (2) that $val(\mathrm{D}_{i,0}) < val(c_i)$ and by (4) that $\mathrm{B}_i \rightsquigarrow c_i$. We conclude that $val(\mathrm{H}_i) > val(\mathrm{D}_{i,0})$ which yields that $\mathrm{A}_{i,k} \rightsquigarrow \mathrm{B}_i$ iff $(a_{i,k,j}, \mathrm{A}_{i,k}) \in F$ for every $j < s$.

For the conversion, we assume first that $\beta_i \notin \mathtt{Sta}_\zeta(F)$. If $val(\mathrm{D}_{i,0}) > val(c_i)$ it follows by (4) that $\mathrm{B}_i \rightsquigarrow c_i$ and hence $val(\mathrm{H}_i) < val(\mathrm{D}_{i,0})$. If otherwise $val(\mathrm{D}_{i,0}) < val(c_i)$ it follows by (2) that there must be an edge $(b_{i,j}, \mathrm{B}_i) \notin F$ with $j < t$ and hence $\mathrm{B}_i \rightsquigarrow \mathrm{D}_{i,k}$ for some $k < r$. Hence $\mathrm{A}_{i,k} \rightsquigarrow \mathrm{B}_i$.

Finally, we assume instead that $\beta_i \in \mathtt{Sta}_\zeta(F)$ and $(a_{i,k,j}, \mathrm{A}_{i,k}) \in F$ for every $j < s$. By (6) we know that $a_{i,k,j} \rightsquigarrow \mathrm{A}_{i,k}$ for every $j < s$. Hence, $\mathrm{A}_{i,k} \rightsquigarrow \mathrm{B}_i$.

□

Lemma 7.1. Proof: We first consider the probability of $\mathtt{ind}$ satisfying the first property. Let

$$\mathtt{first}_{\mathtt{ind}}(b_{i,j}) = \min\{\mathtt{ind}(b_{i,m}, \mathrm{B}_i) \mid m \in [(j+1)s] \setminus [js]\}$$

Thus, we get for all $i \in [n]$:

$$\begin{aligned}\exists j' \in [q]\ \exists j \in [l] : \mathtt{first}_{\mathtt{ind}}(b_{i,j'}) < \mathtt{first}_{\mathtt{ind}}(a_{i,j}) \quad \Rightarrow \\ \exists j \in [l] : \mathtt{first}_{\mathtt{ind}}(b_i) < \mathtt{first}_{\mathtt{ind}}(a_{i,j})\end{aligned}$$

Let $A_i = \{b_{i,0}, \ldots, b_{i,q-1}, a_{i,0}, \ldots, a_{i,l-1}\}$. Then

$$\forall x, y \in A_i : \ Pr[\mathtt{first}_{\mathtt{ind}}(x) < \mathtt{first}_{\mathtt{ind}}(y)] = Pr[\mathtt{first}_{\mathtt{ind}}(x) > \mathtt{first}_{\mathtt{ind}}(y)].$$

That is, choosing $\mathtt{ind} \in \mathcal{I}(G_\zeta)$ uniformly at random gives us a uniformly random permutation of A_i. It follows that:

$$Pr\Big[\forall j' \in [q]\ \forall j \in [l] : \mathtt{first}_{\mathtt{ind}}(b_{i,j'}) > \mathtt{first}_{\mathtt{ind}}(a_{i,j})\Big] = \frac{l! \cdot q!}{(l+q)!}$$

In general, we get that the probability of the first property not being satisfied is:

$$\begin{aligned}Pr\Big[\exists i \in [n]\ \forall j \in [l] : \ \mathtt{first}_{\mathtt{ind}}(b_i) > \mathtt{first}_{\mathtt{ind}}(a_{i,j})\Big] \\ \leq \sum_{i \in [n]} Pr\Big[\forall j \in [l] : \ \mathtt{first}_{\mathtt{ind}}(b_i) > \mathtt{first}_{\mathtt{ind}}(a_{i,j})\Big] \leq \\ \sum_{i \in [n]} Pr\Big[\forall j' \in [q]\ \forall j \in [l] : \ \mathtt{first}_{\mathtt{ind}}(b_{i,j'}) > \mathtt{first}_{\mathtt{ind}}(a_{i,j})\Big] = n \cdot \frac{l! \cdot q!}{(l+q)!} \quad (1)\end{aligned}$$

Next, we consider the probability of $\mathtt{ind}$ satisfying the second property.

We see that $|R_\zeta(E_0(G_\zeta))| = n(t + ls + l + 2)$, and that every element $A \in R_\zeta(E_0(G_\zeta))$ is a set of r edges. For all $A \in R_\zeta(E_0(G_\zeta))$ let

$$\mathtt{last}_{\mathtt{ind}}(A) = max\{\mathtt{ind}(e) \mid e \in A\}$$

Now, the second property can be slightly strengthened as follows:

$$\begin{aligned}\forall i \in [n]\ \forall k \in [l]\ \forall A \in R_\zeta(E_0(G_\zeta)) : \ \mathtt{first}_{\mathtt{ind}}(a_{i,k}) < \mathtt{last}_{\mathtt{ind}}(A) \quad \Rightarrow \\ \mathtt{res}_\zeta(\{e \in E_0(G_\zeta) \mid \mathtt{ind}(e) > \mathtt{last}(\mathtt{ind})\}) > 0\end{aligned}$$

We note that for some $i \in [n], k \in [l], A \in R_\zeta(E_0(G_\zeta))$, $\mathtt{first}_{\mathtt{ind}}(a_{i,k}) > \mathtt{last}_{\mathtt{ind}}(A)$ means that for all $m \in [s], e \in A$, $ind(a_{i,k,m}, A_{i,k}) > ind(e)$. Thus, we get:

$$Pr\Big[\mathtt{first}_{\mathtt{ind}}(a_{i,k}) > \mathtt{last}_{\mathtt{ind}}(A)\Big] = \frac{s! \cdot r!}{(s+r)!}$$

Now, the probability of $\texttt{ind}$ not satisfying the second property can be bounded as follows.

$$\begin{aligned} &Pr\Big[\texttt{res}_\zeta(\{e \in E_0(G_\zeta) \mid \texttt{ind}(e) > \texttt{last}(\texttt{ind})\}) = 0\Big] \\ &\qquad \leq Pr\Big[\exists i \in [n]\ \exists k \in [l]\ \exists A \in R_\zeta(E_0(G_\zeta)) :\ \texttt{first}_{\texttt{ind}}(a_{i,k}) > \texttt{last}_{\texttt{ind}}(A)\Big] \leq \\ &\qquad \sum_{i\in[n]} \sum_{k\in[l]} \sum_{A\in R_\zeta(E_0(G_\zeta))} Pr\Big[\texttt{first}_{\texttt{ind}}(a_{i,k}) > \texttt{last}_{\texttt{ind}}(A)\Big] = n^2 l(t+ls+l+2)\frac{s!\cdot r!}{(s+r)!} \quad (2) \end{aligned}$$

From (1) and (2) it now follows that the probability of $\texttt{ind}$ being good is:

$$p_\zeta \geq 1 - n\frac{l!\cdot q!}{(l+q)!} - n^2 l(t+ls+l+2)\frac{s!\cdot r!}{(s+r)!}.$$

□

Lemma 7.2. Proof: By induction on $f_n(\texttt{Bit}_\zeta(F) \cap [i], \varphi_\zeta^{\texttt{ind}})$. The claim obviously holds true for $f_n(\texttt{Bit}_\zeta(F) \cap [i], \varphi_\zeta^{\texttt{ind}}) = 1$. Let now $f_n(\texttt{Bit}_\zeta(F) \cap [i], \varphi_\zeta^{\texttt{ind}}) > 1$, i.e. particularly $\texttt{Bit}_\zeta(F) \cap [i] \neq \emptyset$. Let $e = \text{argmin}_{e' \in \texttt{Bit}_\zeta(F)\cap[i]} \varphi_\zeta^{\texttt{ind}}(e')$ and $e^* = \texttt{cho}_\zeta(F \setminus \sigma, \texttt{ind})$.

We show that $\texttt{Bit}_\zeta(F) \cap [i] = \texttt{Bit}_\zeta(F \setminus \sigma)$ as well as $\texttt{cho}_\zeta(F \setminus \sigma, \texttt{ind}) = (b_{e,j}, \text{B}_e)$ for some $j < t$ where $e = \text{argmin}_{e' \in \texttt{Bit}_\zeta(F)\cap[i]} \varphi_\zeta^{\texttt{ind}}(e')$.

It holds that $j \in \texttt{Bit}_\zeta(F) \cap [i]$ iff $j < i$ and $\beta_j \in \texttt{Abs}_\zeta(F)$ iff $\beta_j \in \texttt{Abs}_\zeta(F \setminus \sigma)$ iff $j \in \texttt{Bit}_\zeta(F \setminus \sigma)$. As $\texttt{Bit}_\zeta(F) \cap [i] \neq \emptyset$, $\texttt{cho}_\zeta(F \setminus \sigma, \texttt{ind}) = (b_{y,j}, \text{B}_y)$ for some $j < t$ and some $y \in \texttt{Bit}_\zeta(F \setminus \sigma)$. It remains to show that $y = e$. Considering y, it holds that $\texttt{first}_{\texttt{ind}}(b_y) < \texttt{first}_{\texttt{ind}}(b_z)$ for every other $z \in \texttt{Bit}_\zeta(F \setminus \sigma)$. Regarding e, it holds that $\varphi_\zeta^{\texttt{ind}}(e) < \varphi_\zeta^{\texttt{ind}}(f)$ for every other $f \in \texttt{Bit}_\zeta(F \setminus \sigma)$, implying that $\texttt{first}_{\texttt{ind}}(b_e) < \texttt{first}_{\texttt{ind}}(b_f)$ for every other $f \in \texttt{Bit}_\zeta(F \setminus \sigma)$.

Expanding the right side yields

$$f_n(\texttt{Bit}_\zeta(F) \cap [i], \varphi_\zeta^{\texttt{ind}}) = f_n((\texttt{Bit}_\zeta(F) \setminus \{e\}) \cap [i], \varphi_\zeta^{\texttt{ind}}) + f_n(\texttt{Bit}_\zeta(F) \cap [e], \varphi_\zeta^{\texttt{ind}}) \quad (3)$$

Expanding the left side with $H = \{e' \in F \setminus \sigma \mid \texttt{ind}(e') < \texttt{ind}(e^*)\}$ yields

$$\mathbb{E}^\dagger_{G_\zeta}(F, \sigma, \texttt{cho}_\zeta, \texttt{ind}) = \mathbb{E}^\dagger_{G_\zeta}(F \setminus H \setminus \{e^*\}, \sigma, \texttt{cho}_\zeta, \texttt{ind}) + \mathbb{1}_{e^* \in \sigma_{F\setminus H}} \cdot \mathbb{E}^\dagger_{G_\zeta}(F \setminus H, \sigma_{F\setminus H\setminus\{e^*\}}[e^*], \texttt{cho}_\zeta, \texttt{ind}) \quad (4)$$

First, we show that $F \setminus H$ is good w.r.t. $\texttt{ind}$ and $\texttt{Abs}_\zeta(F) = \texttt{Abs}_\zeta(F \setminus H)$. Obviously $\texttt{Bit}_\zeta(F) = \texttt{Bit}_\zeta(F \setminus H)$ as $H \cap \sigma = \emptyset$ and for every $k < i$ it holds that $k \in \texttt{Bit}_\zeta(F \setminus H)$ iff $k \in \texttt{Bit}_\zeta(F)$ by definition of the canonic choice function and by σ being an i-boundary strategy w.r.t. F. Similarly, it follows that $\texttt{Abs}_\zeta(F) = \texttt{Abs}_\zeta(F \setminus H)$. Finally F being good implies that $\texttt{res}_\zeta(\{e' \in F \mid \texttt{ind}(e') > \texttt{last}(\texttt{ind})\}) > 0$. As $\{e' \in F \mid \texttt{ind}(e') > \texttt{last}(\texttt{ind})\} \cap H = \emptyset$, it follows that $F \setminus H$ is good.

Second, we show that $e^* \in \sigma_{F\setminus H}$. Since $F \setminus H$ is good, it follows that $\texttt{res}_\zeta(F \setminus H) > 0$. Hence, we can apply Lemma 6.2 and get that $e^* \in \sigma_{F\setminus H}$ iff $\texttt{low}_\zeta(F \setminus H) \leq e$. Since $F \setminus H$ is good, it immediately follows that $\texttt{low}_\zeta(F \setminus H) = 0$.

We conclude (4) is equivalent to

$$\mathbb{E}^\dagger_{G_\zeta}(F, \sigma, \texttt{cho}_\zeta, \texttt{ind}) = \mathbb{E}^\dagger_{G_\zeta}(F \setminus H \setminus \{e^*\}, \sigma, \texttt{cho}_\zeta, \texttt{ind}) + \mathbb{E}^\dagger_{G_\zeta}(F \setminus H, \sigma_{F\setminus H\setminus\{e^*\}}[e^*], \texttt{cho}_\zeta, \texttt{ind}) \quad (5)$$

and hence by combining (5) and (3), it suffices to show the following two statements.

$$\mathbb{E}^\dagger_{G_\zeta}(F \setminus H \setminus \{e^*\}, \sigma, \texttt{cho}_\zeta, \texttt{ind}) \geq f_n(\texttt{Bit}_\zeta(F \setminus \{e\}) \cap [i], \varphi_\zeta^{\texttt{ind}}) \quad (6)$$

$$\mathbb{E}^\dagger_{G_\zeta}(F \setminus H, \sigma_{F\setminus H\setminus\{e^*\}}[e^*], \texttt{cho}_\zeta, \texttt{ind}) \geq f_n(\texttt{Bit}_\zeta(F) \cap [e], \varphi_\zeta^{\texttt{ind}}) \quad (7)$$

First, we show (6). As $F \setminus H$ is good, it immediately follows that $F \setminus H \setminus \{e^*\}$ is good as well. Also, σ is still an i-boundary strategy w.r.t. $F \setminus H \setminus \{e^*\}$. By induction hypothesis, we conclude that in order to show (6) it suffices to prove

$$f_n(\mathtt{Bit}_\zeta(F \setminus H \setminus \{e^*\}) \cap [i], \varphi_\zeta^{\mathtt{ind}}) \geq f_n(\mathtt{Bit}_\zeta(F \setminus \{e\}) \cap [i], \varphi_\zeta^{\mathtt{ind}}) \tag{8}$$

Hence, it remains to show that $\mathtt{Bit}_\zeta(F \setminus H \setminus \{e^*\}) = \mathtt{Bit}_\zeta(F \setminus \{e\})$. As $\mathtt{Abs}_\zeta(F) = \mathtt{Abs}_\zeta(F \setminus H)$, this directly follows.

Second, we show (7). As $F \setminus H \setminus \{e^*\}$ is good, we know that $\mathtt{res}_\zeta(F \setminus H \setminus \{e^*\}) > 0$ and $\mathtt{low}_\zeta(F \setminus H \setminus \{e^*\}) = 0$, and hence we get by Lemma 6.2 that $\mathtt{Abs}_\zeta(\sigma_{F \setminus H \setminus \{e^*\}}) = \mathtt{Abs}_\zeta(F \setminus H) \setminus \{\beta_e, \alpha_e\}$. Let $\sigma' = \sigma_{F \setminus H \setminus \{e^*\}}[e^*]$. By the same Lemma it follows that $\mathtt{Abs}_\zeta(\sigma') = \mathtt{Abs}_\zeta(F \setminus H) \setminus \{\alpha_e\}$.

As $\mathtt{Abs}_\zeta(F \setminus H \setminus \sigma') = \{\alpha_e\}$, it follows that $h := \mathtt{cho}_\zeta(F \setminus H, \sigma') = (a_{e,m,k}, \mathrm{A}_{e,m})$ for some $m < l$ and some $k < s$. Let $H' = \{e' \in F \setminus H \setminus \sigma' \mid \mathtt{ind}(e') < \mathtt{ind}(h)\}$. Under-approximating (7) therefore yields

$$\mathbb{E}^\dagger_{G_\zeta}(F \setminus H, \sigma', \mathtt{cho}_\zeta, \mathtt{ind}) \geq \mathbb{1}_{h \in \sigma_{F \setminus H \setminus H'}} \cdot \mathbb{E}^\dagger_{G_\zeta}(F \setminus H \setminus H', \sigma_{F \setminus H \setminus H' \setminus \{h\}}[h], \mathtt{cho}_\zeta, \mathtt{ind}) \tag{9}$$

Consider $F \setminus H \setminus H'$. We show that $F \setminus H \setminus H'$ is good and that $\mathtt{Abs}_\zeta(F) = \mathtt{Abs}_\zeta(F \setminus H \setminus H')$. We already know that $\mathtt{Abs}_\zeta(F) = \mathtt{Abs}_\zeta(F \setminus H)$ and also that $\mathtt{Abs}_\zeta(\sigma') = \mathtt{Abs}_\zeta(F \setminus H) \setminus \{\alpha_e\}$. As $\sigma' \cap H' = \emptyset$ and also $\alpha_e \in \mathtt{Abs}_\zeta(F \setminus H \setminus H')$ by definition of the canonic choice function, it follows that $\mathtt{Abs}_\zeta(F) = \mathtt{Abs}_\zeta(F \setminus H \setminus H')$. As $\{e' \in F \mid \mathtt{ind}(e') > \mathtt{last}(\mathtt{ind})\} \cap (H \cup H') = \emptyset$, it follows that $F \setminus H \setminus H'$ fulfills the reset property. We still need to show that $F \setminus H \setminus H'$ satisfies the first property of good sets. Note that every counting edge and every access bit edge disregarding access bit e is already included in $\sigma' \subseteq F \setminus H \setminus H'$. By definition of the canonic choice function, it follows that F contains a complete e-access bit coming after the first counting edge of bit e. Hence, $F \setminus H \setminus H'$ is good.

Now, we show that $h \in \sigma_{F \setminus H \setminus H'}$. Since $\beta_e \in \mathtt{Sta}_\zeta(F \setminus H \setminus H')$ and since $F \setminus H \setminus H'$ is good, it follows by Lemma 6.2 that $h \in \sigma_{F \setminus H \setminus H'}$.

We conclude that in order to show (7) it suffices to prove

$$\mathbb{E}^\dagger_{G_\zeta}(F \setminus H \setminus H', \sigma_{F \setminus H \setminus H' \setminus \{h\}}[h], \mathtt{cho}_\zeta, \mathtt{ind}) \geq f_n(\mathtt{Bit}_\zeta(F) \cap [e], \varphi_\zeta^{\mathtt{ind}}) \tag{10}$$

Next, we show that $\sigma_{F \setminus H \setminus H' \setminus \{h\}}[h]$ is an e-boundary strategy w.r.t. $F \setminus H \setminus H'$. First note that $\mathtt{res}_\zeta(F \setminus H \setminus H' \setminus \{h\}) > 0$, hence by Lemma 6.2 it follows that $\mathtt{Sta}_\zeta(F \setminus H \setminus H' \setminus \{h\}) = \mathtt{Sta}_\zeta(\sigma_{F \setminus H \setminus H' \setminus \{h\}})$. As particularly $\mathtt{low}_\zeta(F \setminus H \setminus H' \setminus \{h\}) = e$, it follows that $\sigma_{F \setminus H \setminus H' \setminus \{h\}}$ is an e-boundary strategy w.r.t. $F \setminus H \setminus H' \setminus \{h\}$. It is easy to see that then $\sigma_{F \setminus H \setminus H' \setminus \{h\}}[h]$ is an e-boundary strategy w.r.t. $F \setminus H \setminus H'$

We conlude that in order to show (10) is suffices to prove by induction hypothesis that

$$\mathbb{E}^\dagger_{G_\zeta}(F \setminus H \setminus H', \sigma_{F \setminus H \setminus H' \setminus \{h\}}[h], \mathtt{cho}_\zeta, \mathtt{ind}) \geq f_n(\mathtt{Bit}_\zeta(F \setminus H \setminus H') \cap [e], \varphi_\zeta^{\mathtt{ind}}) \tag{11}$$

Hence, it remains to show that $\mathtt{Bit}_\zeta(F \setminus H \setminus H') = \mathtt{Bit}_\zeta(F)$ which directly follows from $\mathtt{Abs}_\zeta(F) = \mathtt{Abs}_\zeta(F \setminus H \setminus H')$.

□

Lemma 7.4. Proof:

$$\begin{aligned}
\mathbb{E}_G(E_0(G_\zeta),\sigma) &\overset{4.1}{=} \mathbb{E}^*_G(E_0(G_\zeta),\sigma)\\
&\overset{4.3}{\geq} \mathbb{E}^\dagger_G(E_0(G_\zeta),\sigma,\mathtt{cho}_\zeta)\\
&= \frac{1}{|\mathcal{I}(G_\zeta)|}\sum_{\mathtt{ind}\in\mathcal{I}(G_\zeta)} \mathbb{E}^\dagger_{G_\zeta}(E_0(G_\zeta),\sigma,\mathtt{cho}_\zeta,\mathtt{ind})\\
&\overset{7.1}{\geq} p_\zeta\cdot\frac{1}{|\mathcal{I}^\oplus_\zeta|}\sum_{\mathtt{ind}\in\mathcal{I}^\oplus_\zeta} \mathbb{E}^\dagger_{G_\zeta}(E_0(G_\zeta),\sigma,\mathtt{cho}_\zeta,\mathtt{ind})\\
&\overset{7.3}{\geq} p_\zeta\cdot\frac{1}{|\mathcal{I}^\oplus_\zeta|}\sum_{\mathtt{ind}\in\mathcal{I}^\oplus_\zeta} f_n([n],\varphi^{\mathtt{ind}}_\zeta)\\
&= p_\zeta\cdot\frac{1}{|\mathcal{S}(n)|}\sum_{\varphi\in\mathcal{S}(n)} f_n([n],\varphi)\\
&\overset{5.2}{=} p_\zeta\cdot g(n)
\end{aligned}$$

□

Theorem 7.5. Proof: First, we pick parameters for $\zeta=(n,r,l,s,t)$ such that p_ζ is at least $\frac{1}{2}$. Let $t=ls$ and $r=s$. Then we get from Lemma 7.1 that

$$p_\zeta \geq 1-n\frac{(l!)^2}{(2l)!}-n^2l(2l(s+1)+2)\frac{(s!)^2}{(2s)!}. \tag{12}$$

It is easy to prove, by induction, that

$$\frac{(k!)^2}{(2k)!}\leq\frac{1}{2^k}.$$

Thus, setting $l=s=3\log n$ we get from (12) that for n sufficiently large:

$$p_\zeta \geq 1-\frac{n}{2^l}-\frac{n^2l(2l(s+1)+2)}{2^s}\geq 1-\frac{n^2(3\log n)^4}{2^{3\log n}}=1-\frac{(3\log n)^4}{n}\geq\frac{1}{2}.$$

From Lemma 7.4 we know that $\mathbb{E}(E_0(G_\zeta),\sigma)\geq p_\zeta\cdot g(n)$, and from Lemma 5.1 we know that $g(n)=2^{\Omega(\sqrt{n})}$. It follows that for G_ζ, with $\zeta=(n,3\log n,3\log n,3\log n,(3\log n)^2)$, $\mathbb{E}(E_0(G_\zeta),\sigma)=2^{\Omega(\sqrt{n})}$. Note that, by Lemma 6.1, the number of vertices and edges of G_ζ are

$$\begin{aligned}
|V_\zeta| &= |V^0_\zeta|+|V^1_\zeta| = n(2+t+ls)+n(1+l+(l+2)r)+1\\
&= 27n\log^2 n+9n\log n+3n+1\leq 40n\log^2 n,\\
|E_\zeta| &= |E_0(G_\zeta)|+|E_1(G_\zeta)| = n(r(2+t+l(s+1))+t+ls)+n(2r(l+1)+ls+1)+1\\
&= 54n\log^3 n+54n\log^2 n+12n\log n+n+1\leq 122n\log^3 n.
\end{aligned}$$

If n' is the number of vertices of G_ζ, then the number of bits is at least $n\geq n'/(40\log^2 n')$, and the expected number of switches performed by the RandomFacet algorithm is $2^{\Omega(\sqrt{n'}/\log n')}=2^{\tilde{\Omega}(\sqrt{n'})}$.

□