Easy to Win, Hard to Master: Optimal Strategies in Parity Games with Costs*

Alexander Weinert[1] and Martin Zimmermann[2]

1 **Reactive Systems Group, Saarland University, 66123 Saarbrücken, Germany**
`weinert@react.uni-saarland.de`
2 **Reactive Systems Group, Saarland University, 66123 Saarbrücken, Germany**
`zimmermann@react.uni-saarland.de`

Abstract

The winning condition of a parity game with costs requires an arbitrary, but fixed bound on the distance between occurrences of odd colors and the next occurrence of a larger even one. Such games quantitatively extend parity games while retaining most of their attractive properties, i.e, determining the winner is in NP and co-NP and one player has positional winning strategies.

We show that the characteristics of parity games with costs are vastly different when asking for strategies realizing the minimal such bound: the solution problem becomes PSPACE-complete and exponential memory is both necessary in general and always sufficient. Thus, playing parity games with costs optimally is harder than just winning them. Moreover, we show that the tradeoff between the memory size and the realized bound is gradual in general.

1998 ACM Subject Classification D.2.4 Software/Program Verification.

Keywords and phrases Parity Games with Costs, Optimal Strategies, Memory Requirements, Tradeoffs

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

1 Introduction

Recently, the focus of research into infinite games for the synthesis of reactive systems moved from studying qualitative winning conditions to quantitative ones. This paradigm shift entails novel research questions, as quantitative conditions induce a (partial) ordering of winning strategies. In particular, there is a notion of semantic optimality for strategies which does not appear in the qualitative setting. Thus, in the quantitative setting, one can ask whether computing optimal strategies is harder than computing arbitrary ones, whether optimal strategies are necessarily larger than arbitrary ones, and whether there are tradeoffs between different quality measures for strategies, e.g., between the size of the strategy and its semantic quality (in terms of satisfaction of the winning condition).

As an introductory example consider the classical (max)-parity condition, which is defined for an infinite sequence drawn from a finite subset of the natural numbers, so-called colors. The parity condition is satisfied if almost all occurrences of an odd color are *answered* by a later occurrence of a larger even color, e.g., the sequence

$$\pi = 1\,0\,2\;1\,0\,0\,2\;1\,0\,0\,0\,2\;1\,0\,0\,0\,0\,2\;1\,0\,0\,0\,0\,0\,2\;1\,0\,0\,0\,0\,0\,0\,2\;1\,0\,0\,0\,0\,0\,0\,0\,2\cdots$$

satisfies the parity condition, as every 1 is eventually answered by a 2.

* Supported by the project "TriCS" (ZI 1516/1-1) of the German Research Foundation (DFG).

The finitary parity condition [9] is obtained by additionally requiring the existence of a bound b such that almost every odd color is answered within at most b steps, i.e., π does not satisfy the finitary parity condition, as the length of the zero-blocks is unbounded. Thus, solving a finitary parity game is a boundedness problem: in order to satisfy the condition, an arbitrary, but fixed bound has to be met. In particular, winning strategies for finitary parity games are naturally ordered by the minimal bound they realize along all consistent plays. Thus, finitary parity games induce an optimization problem: compute an optimal winning strategy, i.e., one that guarantees the smallest possible bound.

Other examples for such quantitative winning conditions include mean payoff [12, 34] and energy [3, 5] conditions and their combinations and extensions, request-response conditions [19, 31], finitary parity [9] and parity with costs [16], and parameterized extensions of Linear Temporal Logic (LTL) [1, 14, 21, 32, 33]. Often, these conditions are obtained by interpreting a classical qualitative winning condition quantitatively, e.g., the finitary parity condition.

Often, the best known algorithms for solving such boundedness conditions are as fast as the best ones for their respective qualitative variant, while the fastest known algorithms for the optimization problem are worse. For example, solving games with winning conditions in Prompt-LTL, a quantitative variant of LTL, is 2ExpTime-complete [21] (i.e., as hard as solving classical LTL games [26]), while computing optimal strategies is only known to be in 3ExpTime [32]. The same is true for the sizes of strategies, which jumps from tight doubly-exponential bounds to triply-exponential upper bounds. The situation is similar for other winning conditions as well, e.g., request-response conditions [19]. These examples all have in common that there are no known lower bounds on the complexity and the memory requirements in the optimization variant, except for the trivial ones for the qualitative case. A notable exception are finitary parity games, which are solvable in polynomial time [9] and thus simpler than parity games (according to the state-of-the-art).

In this work, we study optimal strategies in parity games with costs, a generalization of finitary parity games. In this setting, we are able to show that computing optimal strategies is indeed harder than computing arbitrary strategies, and that optimal strategies have exponentially larger memory requirements in general. A parity game with costs is played in a finite directed graph whose vertices are partitioned into the positions of Player 0 and the positions of Player 1. Starting at an initial vertex, the players move a token through the graph: if it is placed at a vertex of Player i, then this player has to move it to some successor. Thus, after ω rounds, the players have produced an infinite path through the graph, a so-called play. The vertices of the graph are colored by natural numbers and the edges are labeled by a cost (encoded in unary). These two labelings induce the parity condition with costs: there has to be a bound b such that almost all odd colors are followed by a larger even color such that the cost between these two positions is at most b. Thus, the sequence π from above satisfies the parity condition with costs, if the cost of the zero-blocks is bounded. Note that the finitary parity condition is the special case where every edge has cost one and the parity condition is the special case where every edge has cost zero.

Thus, to win a parity game with costs, Player 0 has to bound the cost between requests and their responses along all plays. If Player 0 has any such strategy, then she has a positional strategy [16], i.e., a strategy that determines the next move based only on the vertex the token is currently at, oblivious to the history of the play. If we let n denote the number of vertices of the graph the game is played in, then such a strategy uniformly bounds the costs to some bound $b \leq n$ [16], which we refer to as the cost of the strategy. Furthermore, Mogavero et al. showed that the winner of a parity game with costs can be determined in UP $\cap$ co-UP [23]. All previous work on parity games with costs was concerned with the

boundedness variant, i.e., the problems ask to find some bound, not necessarily the best one. Here, in contrast, we study optimal strategies in parity games with costs.

1.1 Our Contribution

Our first result shows that determining whether Player 0 has a strategy whose cost is smaller than a given bound b is PSpace-complete. Thus, computing the bound of an optimal strategy is strictly harder than just deciding whether or not some bound exists (unless PSpace $\subseteq$ UP $\cap$ co-UP). The hardness result is shown by a reduction from QBF and uses the bound b to require Player 0's strategy to implement a satisfying Skolem function for the formula, where picking truth values is encoded by requests of odd colors. The lower bound is complemented by a polynomial space algorithm that is obtained from an alternating polynomial time Turing machine that simulates a finite-duration variant of parity games with costs that is won by Player 0 if and only if she can enforce a cost of at most b in the original game. To obtain the necessary polynomial upper bound on the play length we rely on the upper bound n on the optimal bound and on pumping arguments to deal with the costs along the play, and on a first-cycle variant of parity games [2] to capture the parity condition in parts of the graph where all edges have cost zero.

Our second result concerns memory requirements of optimal strategies. A corollary of the correctness of the finite-duration game yields exponential upper bounds: if Player 0 has a strategy of cost b, then also one of cost b and of size $(b+2)^d \cdot (n+1) = 2^{d \log(b+2)} \cdot (n+1)$, where d is the number of odd colors in the game.

As a third result, we show that this bound is in general tight: we present a family $\mathcal{G}_d$ of parity games with costs such that $\mathcal{G}_d$ has d odd colors and Player 0 requires strategies of size $2^d - 2$ to play optimally in each $\mathcal{G}_d$. This result is based on using the bound b to require Player 0 to store which odd colors have an open request and in which order they were posed. Our result improves a linear bound presented by Chatterjee and Fijalkow [8].

Finally, we study the tradeoff between memory size and cost of a strategy witnessed by the results above: arbitrary winning strategies are as small as possible, i.e., positional, but in general have cost n. In contrast, optimal strategies realize a smaller bound, but might have exponential size. Hence, one can trade cost for memory and vice versa.

Our fourth result shows that this tradeoff is gradual in the games $\mathcal{G}_d$: there are strategies $\sigma_1, \sigma_2, \ldots, \sigma_d$ such that $1 = |\sigma_1| < |\sigma_2| < \cdots < |\sigma_d| = 2^d - 2$ and $b_1 > b_2 > \cdots > b_d$, where b_j is the cost of σ_j. Furthermore, we show that the strategy σ_j has minimal size among all strategies of cost b_j. Equivalently, the strategy σ_j has minimal cost among all strategies whose size is not larger than σ_j's size.

Both lower bounds we prove and the tradeoff result already hold for the special case of finitary parity games, which can even be solved in polynomial time [9]. Hence, in this case, the gap between just winning and playing optimally is even larger. Also, our results are straightforwardly extendable to both bounded variants, i.e., bounded parity games [9] and bounded parity games with costs [16].

All proofs omitted due to space restrictions can be found in the full version [29].

1.2 Related Work

Tradeoffs in infinite games have been studied before, e.g., in stochastic and timed games, one can trade memory for randomness, i.e., randomized strategies are smaller than deterministic ones [7, 11]. A detailed overview of more recent results in this direction and of tradeoffs in

multi-dimensional winning conditions is given in the thesis of Randour [27]. The nature of these results is quite different from ours.

Lang investigated optimal strategies in the resource reachability problem on pushdown graphs [22], where there exist a finite number of counters, which may be increased and reset, but not read during a play. He shows that in order to keep the values of the counters minimal during the play, exponential memory in the number of counters is both necessary and sufficient for Player 0. While the author shows the corresponding decision problem to be decidable, he does not provide a complexity analysis of the problem. Furthermore, the setting of the problem is quite different to the model considered in this work: he considers infinite graphs and multiple counters, but only reachability conditions, while we consider finite graphs and implicit counters tied to the acceptance condition, which is a general parity condition.

Also, Fijalkow et al. proved the non-existence of a certain tradeoff between size and quality of strategies in boundedness games [15], which refuted a conjecture with important implications for automata theory and logics. Such games are similar to those considered by Lang in that they are played in potentially infinite arenas and have multiple counters.

Finally, our results have been recently rephrased in terms of window-parity games [4], another quantitative variant of parity games.

2 Preliminaries

We denote the non-negative integers by $\mathbb{N}$ and define $[n] = \{0, 1, \dots, n-1\}$ for every $n \geq 1$.

An *arena* $\mathcal{A} = (V, V_0, V_1, E, v_I)$ consists of a finite, directed graph (V, E), a partition $\{V_0, V_1\}$ of V into the positions of Player 0 (drawn as circles) and Player 1 (drawn as rectangles), and an initial vertex $v_I \in V$. A *play* in $\mathcal{A}$ is an infinite path $\rho = v_0 v_1 v_2 \cdots$ through (V, E) starting in v_I. To rule out finite plays, we require every vertex to be non-terminal. A *game* $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$ consists of an arena $\mathcal{A}$ with vertex set V and a set $\mathrm{Win} \subseteq V^\omega$ of winning plays for Player 0. The set of winning plays for Player 1 is $V^\omega \setminus \mathrm{Win}$.

A *strategy* for Player i is a mapping $\sigma \colon V^* V_i \to V$ where $(v, \sigma(wv)) \in E$ for all $wv \in V^* V_i$. We say that σ is *positional* if $\sigma(wv) = \sigma(v)$ for every $wv \in V^* V_i$. We often view positional strategies as a mapping $\sigma \colon V_i \to V$. A play $v_0 v_1 v_2 \cdots$ is *consistent* with a strategy σ for Player i, if $v_{j+1} = \sigma(v_0 \cdots v_j)$ for every j with $v_j \in V_i$. A strategy σ for Player i is a *winning strategy* for $\mathcal{G}$ if every play that is consistent with σ is won by Player i. If Player i has a winning strategy, then we say she wins $\mathcal{G}$. *Solving* a game amounts to determining its winner.

A *memory structure* $\mathcal{M} = (M, m_I, \mathrm{Upd})$ for an arena (V, V_0, V_1, E, v_I) consists of a finite set M of memory states, an initial memory state $m_I \in M$, and an update function $\mathrm{Upd} \colon M \times E \to M$. The update function can be extended to finite play prefixes in the usual way: $\mathrm{Upd}^+(m, v) = m$ and $\mathrm{Upd}^+(m, wvv') = \mathrm{Upd}(\mathrm{Upd}^+(m, wv), (v, v'))$ for $w \in V^*$ and $(v, v') \in E$. A next-move function for Player i $\mathrm{Nxt} \colon V_i \times M \to V$ has to satisfy $(v, \mathrm{Nxt}(v, m)) \in E$ for all $v \in V_i$ and all $m \in M$. It induces a strategy σ for Player i with memory $\mathcal{M}$ via $\sigma(v_0 \cdots v_n) = \mathrm{Nxt}(v_n, \mathrm{Upd}^+(m_I, v_0 \cdots v_n))$. A strategy is called *finite-state* if it can be implemented by a memory structure. We define $|\mathcal{M}| = |M|$. The size of a finite-state strategy is the size of a smallest memory structure implementing it.

An arena $\mathcal{A} = (V, V_0, V_1, E, v_I)$ and a memory structure $\mathcal{M} = (M, m_I, \mathrm{Upd})$ for $\mathcal{A}$ induce the expanded arena $\mathcal{A} \times \mathcal{M} = (V \times M, V_0 \times M, V_1 \times M, E', (v_I, m_I))$ where $((v, m), (v', m')) \in E'$ if and only if $(v, v') \in E$ and $\mathrm{Upd}(m, (v, v')) = m'$. Every play $\rho = v_0 v_1 v_2 \cdots$ in $\mathcal{A}$ has a unique extended play $\mathrm{ext}(\rho) = (v_0, m_0)(v_1, m_1)(v_2, m_2) \cdots$ in $\mathcal{A} \times \mathcal{M}$ defined by $m_0 = m_I$ and $m_{n+1} = \mathrm{Upd}(m_n, (v_n, v_{n+1}))$, i.e., $m_n = \mathrm{Upd}^+(m_I, v_0 \cdots v_n)$. The extended play of a finite play prefix in $\mathcal{A}$ is defined similarly.

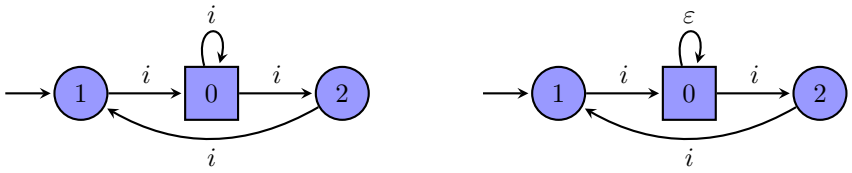

Figure 1 Two parity games with costs. Player 1 only has a winning strategy in the left game.

3 Parity Games with Costs

In this section, we introduce the parity condition with costs [16]. Fix an arena $\mathcal{A} = (V, V_0, V_1, E, v_I)$. A cost function for $\mathcal{A}$ is an edge-labeling $\mathrm{Cst}\colon E \to \{\varepsilon, i\}$. [1] Edges labeled with i are called increment-edges while edges labeled with ε are called ε-edges. We extend the edge-labeling to a cost function over plays obtained by counting the number of increment-edges traversed during the play, i.e., $\mathrm{Cst}(\rho) \in \mathbb{N} \cup \{\infty\}$. The cost of a finite path is defined analogously. Also, fix a coloring $\Omega\colon V \to \mathbb{N}$ of $\mathcal{A}$'s vertices and let $\mathrm{Ans}(c) = \{c' \in \mathbb{N} \mid c' \geq c \text{ and } c' \text{ is even}\}$ be the set of colors that answer a request of color c.

Let $\rho = v_0 v_1 v_2 \cdots$ be a play. We define the cost-of-response at position $k \in \mathbb{N}$ of ρ by

$$\mathrm{Cor}(\rho, k) = \min\{\mathrm{Cst}(v_k \cdots v_{k'}) \mid k' \geq k \text{ and } \Omega(v_{k'}) \in \mathrm{Ans}(\Omega(v_k))\} \ ,$$

where we use $\min \emptyset = \infty$, i.e., $\mathrm{Cor}(\rho, k)$ is the cost of the infix of ρ from position k to its first answer, and ∞ if there is no answer.

We say that a request at position k is answered with cost b, if $\mathrm{Cor}(\rho, k) = b$. Consequently, a request at a position k with an even color is answered with cost zero. Furthermore, we say that a request at position k is unanswered with cost ∞, if there is no position $k' \geq k$ such that $\Omega(v_{k'}) \in \mathrm{Ans}(\Omega(v_k))$ and we have $\mathrm{Cst}(v_k v_{k+1} v_{k+2} \cdots) = \infty$, i.e., there are infinitely many increment-edges after position k, but no answer. There is a third alternative: a request can be unanswered with finite cost, i.e., in case it is not answered, but the play ρ contains only finitely many increment-edges. Still, the cost-of-response is infinite in this case.

The parity condition with costs is defined as

$$\mathrm{CostParity}(\Omega, \mathrm{Cst}) = \{\rho \in V^\omega \mid \limsup_{k \to \infty} \mathrm{Cor}(\rho, k) < \infty\} \ ,$$

i.e., ρ satisfies the condition, if there exists a bound $b \in \mathbb{N}$ such that all but finitely many requests are answered with cost less than b. In particular, only finitely many requests may be unanswered, even with finite cost. Note that the bound b depends on the play ρ.

A game $\mathcal{G} = (\mathcal{A}, \mathrm{CostParity}(\Omega, \mathrm{Cst}))$ is called a parity game with costs. If Cst assigns ε to every edge, then $\mathrm{CostParity}(\Omega, \mathrm{Cst})$ is a classical (max-) parity condition, denoted by $\mathrm{Parity}(\Omega)$. Dually, if Cst assigns i to every edge, then $\mathrm{CostParity}(\Omega, \mathrm{Cst})$ is equal to the finitary parity condition over Ω, as introduced by Chatterjee et al. [9] and denoted by $\mathrm{FinParity}(\Omega)$. In these cases, we refer to $\mathcal{G}$ as a parity or a finitary parity game, respectively.

Player 1 has two ways of winning a parity game with costs: Either he violates the classical parity condition, or he delays answers to requests arbitrarily. Consider the two parity games with costs shown in Figure 1. In the game on the left-hand side, Player 1 has a winning strategy, by taking the self-loop of the node labeled with 0 n times upon the n-th visit to it.

[1] Note that using the abstract costs ε and i essentially entails a unary encoding of weights. We discuss the case of a binary encoding of arbitrary weights in Section 7.

Thus, he delays answers to the request for 1 arbitrarily and wins by the second condition. In the game on the right-hand side, however, Player 1 does not have a winning strategy. If he eventually remains in the node labeled with 0, then there are only finitely many requests, only one of which is unanswered. Thus, the cost of the play is 0, i.e., it is won by Player 0. If he, on the other hand, always leaves the node labeled with 0 eventually, then each request is answered with cost 2, hence Player 0 wins as well.

▶ **Theorem 1.**

1. *Solving parity games is in* UP ∩ CO-UP. *The winner has a positional winning strategy [13, 20, 24].*
2. *Solving finitary parity games is in* PTIME. *If Player* 0 *wins, then she has a positional winning strategy, but Player* 1 *has in general no finite-state winning strategy [9].*
3. *Solving parity games with costs is in* UP ∩ CO-UP. *If Player* 0 *wins, then she has a positional winning strategy, but Player* 1 *has in general no finite-state winning strategy [16].*

A winning strategy for Player 0 in a parity game with costs does not have to realize a uniform bound b on the value $\limsup_{k\to\infty} \mathrm{Cor}(\rho, k)$ among all plays ρ that are consistent with σ, but the bound may depend on the play. To capture the cost of a strategy, we first define the cost of a play ρ as $\mathrm{Cst}(\rho) = \limsup_{k\to\infty} \mathrm{Cor}(\rho, k)$ and the cost of a strategy σ as $\mathrm{Cst}(\sigma) = \sup_\rho \mathrm{Cst}(\rho)$, where the supremum ranges over all plays ρ that are consistent with σ. A strategy is optimal for $\mathcal{G}$ if it has minimal cost among all strategies for $\mathcal{G}$.

A corollary of Theorem 1.3 yields an upper bound on the cost of an optimal strategy: a straightforward pumping argument shows that a positional winning strategy, which always exists if there exists any winning strategy, realizes a uniform bound $b \leq n$ for every play, where n is the number of vertices of the game [16].

▶ **Corollary 2.** *Let $\mathcal{G}$ be a parity game with costs with n vertices. If Player* 0 *wins $\mathcal{G}$, then she has a strategy σ with $\mathrm{Cst}(\sigma) \leq n$, i.e., an optimal strategy has cost at most n.*

4 The Complexity of Solving Parity Games with Costs Optimally

In this section we study the complexity of determining the cost of an optimal strategy for a parity game with costs. Recall that solving such games is in UP ∩ CO-UP (and therefore unlikely to be NP-complete or CO-NP-complete) while solving the special case of finitary parity games is in PTIME. Our main result of this section shows that checking whether a strategy of cost at most b exists is PSPACE-complete, where hardness already holds for finitary parity games. Therefore, this decision problem is harder than just solving the game (unless PSPACE ⊆ UP ∩ CO-UP, respectively PSPACE ⊆ PTIME).

▶ **Theorem 3.** *The following problem is* PSPACE*-complete: "Given a parity game with costs $\mathcal{G}$ and a bound $b \in \mathbb{N}$, does Player* 0 *have a strategy σ for $\mathcal{G}$ with $\mathrm{Cst}(\sigma) \leq b$?"*

The proof of this theorem is split into two lemmas, Lemma 4 showing membership and Lemma 7 showing hardness, which are presented in Section 4.1 and Section 4.2, respectively.

4.1 Playing Parity Games with Costs Optimally is in PSPACE

In this section we establish PSPACE-membership of the previously defined decision problem.

▶ **Lemma 4.** *The following problem is in* PSPACE*: "Given a parity game with costs $\mathcal{G}$ and a bound $b \in \mathbb{N}$, does Player* 0 *have a strategy σ for $\mathcal{G}$ with $\mathrm{Cst}(\sigma) \leq b$?"*

The remainder of this section is dedicated to the proof of Lemma 4. To this end, we fix a parity game with costs $\mathcal{G} = (\mathcal{A}, \text{CostParity}(\Omega, \text{Cst}))$ with $\mathcal{A} = (V, V_0, V_1, E, v_I)$ and a bound b. Let $n = |V|$. First, let us remark that we can assume w.l.o.g. $b \leq n$: If $b > n$, then, due to Corollary 2, we just have to check whether Player 0 wins $\mathcal{G}$. This is possible in PSpace due to Theorem 1.3.

To obtain a polynomial space algorithm, we first turn the quantitative game $\mathcal{G}$ into a qualitative parity game $\mathcal{G}'$ in which the cost of open requests is explicitly tracked up to the bound b . To this end, we use functions r mapping odd colors to $\{\bot\} \cup [b+1]$, where $\bot$ encodes that no open request of this color is pending. Additionally, whenever the bound b is exceeded for some request, all open requests are reset and a so-called overflow counter is increased, up to value n. This accounts for a bounded number of unanswered requests, which are allowed by the parity condition with costs. Intuitively, Player 1 wins $\mathcal{G}'$ if he either exceeds the upper bound b at least n times, or if he enforces an infinite play of finite cost with infinitely many unanswered requests. If he wins by the former condition, this implies that he can also enforce infinitely many excesses of b via a pumping argument. The latter condition accounts for plays in which Player 1 wins without violating the bound b repeatedly, but by violating the classical parity condition. We show that Player 0 has a strategy σ in $\mathcal{G}$ with $\text{Cst}(\sigma) \leq b$ if and only if Player 0 wins $\mathcal{G}'$ from its initial vertex $v_I' = (v_I, r_0, 0)$. Here, r_0 encodes the open requests of the play prefix v_I.

The resulting game $\mathcal{G}'$ is of exponential size in the number of odd colors and can therefore in general not be solved in polynomial space in n. Thus, in a second step, we construct a finite-duration variant $\mathcal{G}_f'$ of $\mathcal{G}'$, which is played on the same arena as $\mathcal{G}'$, but each play is stopped after a polynomial number of moves. We show that Player 0 wins $\mathcal{G}'$ if and only if she wins $\mathcal{G}_f'$.

To conclude, we show how to simulate $\mathcal{G}_f'$ on the fly on an alternating Turing machine in polynomial time in n, which yields a polynomial space algorithm by removing the alternation [6].

We begin by defining $\mathcal{G}'$. Let $R = (\{\bot\} \cup [b+1])^D$ be the set of request functions, where D is the set of odd colors occurring in $\mathcal{G}$. Here, $r(c) = \bot$ denotes that there is no open request for the color c, while $r(c) \neq \bot$ encodes that the oldest open request of c has incurred cost $r(c)$. Using these functions, we define the memory structure $\mathcal{M} = (R \times [n+1], m_I, \text{Upd})$, where the second component $[n+1] = \{0, \ldots, n\}$ implements the overflow counter. It suffices to bound this counter by n, since, if Player 1 can enforce n overflows in $\mathcal{G}$, then, by a pumping argument, he can also enforce infinitely many.

The initial memory state m_I is the pair $(r_I, 0)$, where r_I is the function mapping all odd colors to $\bot$, if $\Omega(v_I)$ is even. If $\Omega(v_I)$ is odd, however, r_I maps $\Omega(v)$ to 0, and all other odd colors to $\bot$. The update function $\text{Upd}(m, e)$ is defined such that traversing an edge $e = (v, v')$ updates $m = (r, o)$ to $m' = (r', o')$ by performing the following steps in order:

- If e is an increment-edge, then for each c with $r(c) \neq \bot$, $r(c)$ is increased by one.
- If there exists a color c such that $r(c) > b$, then reset all open requests to $\bot$ and set o to the minimum of $o+1$ and n.
- If $\Omega(v')$ is even, reset all requests for colors c' with $c' \leq \Omega(v')$ to $\bot$.
- If $\Omega(v')$ is odd, then set $r'(\Omega(v'))$ to the maximum of $r(\Omega(v'))$ and 0, with $\max\{\bot, 0\} = 0$.

The resulting function r' is an element of R and the resulting o' is at most n.

The evolution of the memory states on a play prefix is depicted in Figure 2. The prefix and the sequence of memory states for $b = 2$ are shown in the upper and lower row, respectively. The request functions r are given in vector notation, where the upper and the lower entry denote $r(1)$ and $r(3)$, respectively.

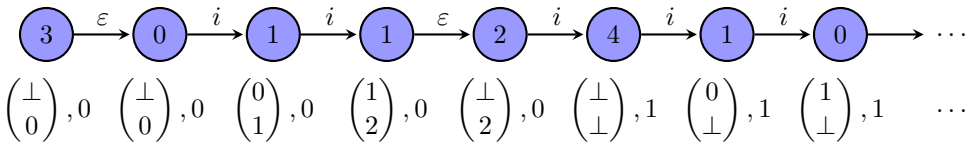

Figure 2 Example of the evolution of the request-functions during a play for the bound $b = 2$.

We define the parity game $\mathcal{G}' = (\mathcal{A} \times \mathcal{M}, \text{Parity}(\Omega'))$, with $\Omega'(v, r, o) = \Omega(v)$ for $o < n$ and $\Omega'(v, r, n) = 1$. Note that every play that encounters a vertex of the form (v, r, n) at some point is winning for Player 1. Although $\mathcal{G}'$ has no cost function, we say that an edge $((v, m), (v', m'))$ of $\mathcal{G}'$ is an increment-edge, if (v, v') is an increment-edge in $\mathcal{G}$, otherwise it is an ε-edge.

It suffices to solve $\mathcal{G}'$ to determine whether Player 0 can bound the cost in $\mathcal{G}$ by b.

▶ **Proposition 5.** *There exists a strategy σ for Player 0 in $\mathcal{G}$ with $\text{Cst}(\sigma) \leq b$ if and only if Player 0 wins $\mathcal{G}'$.*

The parity game $\mathcal{G}'$ is of exponential size, since the number of possible request functions is exponential. Hence, explicitly constructing and solving $\mathcal{G}'$ using standard methods does not yield a polynomial space algorithm. Rather, we now show that it suffices to consider a finite-duration variant $\mathcal{G}'_f$ of $\mathcal{G}'$, in which each play ends after a polynomial number of steps. The winner of $\mathcal{G}'_f$ can then be determined by simulating $\mathcal{G}'_f$ on the fly on an alternating polynomial time Turing machine.

We say that a play prefix $(v_0, r_0, o_0) \cdots (v_j, r_j, o_j)$ of $\mathcal{G}'$ is *settled* (for Player 1), if $o_j = n$ or if its projection $v_0 \cdots v_j$ contains a cycle of cost zero whose maximal color is odd. Note that a cycle with odd maximal color that contains an increment-edge does not settle the play, as its existence does not imply the existence of a cycle in $\mathcal{G}'$. Fix $\ell = 3(n+1)^5$. We construct the finite duration variant $\mathcal{G}'_f = (\mathcal{A} \times \mathcal{M}, \text{Win}_\ell)$ of $\mathcal{G}'$, where a play $\rho = (v_0, r_0, o_0)(v_1, r_1, o_1)(v_2, r_2, o_2) \cdots$ is winning for Player 0 if and only if the prefix $(v_0, r_0, o_0) \cdots (v_\ell, r_\ell, o_\ell)$ of length $\ell + 1$ is not settled. Note that $\mathcal{G}'_f$ is indeed a game of finite duration, as the winner is certain after ℓ moves. Hence, $\mathcal{G}'_f$ is determined [30].

▶ **Proposition 6.** *Player 0 wins $\mathcal{G}'$ if and only if she wins $\mathcal{G}'_f$.*

Combining Propositions 5 and 6 implies that Player 0 has a strategy σ for $\mathcal{G}$ with $\text{Cst}(\sigma) \leq b$ if and only if she wins $\mathcal{G}'_f$. Thus it remains to show that we can simulate $\mathcal{G}'_f$ on an alternating Turing machine in polynomial time.

We show how to simulate a play of the finite-duration game $\mathcal{G}'_f$ on an alternating polynomial time Turing machine using the game semantics of such machines, i.e., two players construct a single path of a run of the machine. The existential player takes the role of Player 0, the universal one the role of Player 1. The Turing machine keeps track of the current vertex of the simulated play of $\mathcal{G}'_f$, whether a settling cycle has been encountered, and of the number of moves already simulated. Once ℓ moves have been simulated, the machine terminates and accepts if and only if the play constructed during the run is not settled. To check for zero cycles with odd maximal color, the machine uses the latest-appearance data structure (see, e.g., [17]), which can be updated in linear time and is reset every time an increment-edge is traversed. Note that this algorithm involves neither the explicit construction of $\mathcal{G}'$ nor that of $\mathcal{G}'_f$.

Thus, the Turing machine accepts $\mathcal{G}$ and b if and only if Player 0 wins $\mathcal{G}'_f$. Hence, APTIME = PSPACE [6] completes the proof.

4.2 Playing Parity Games with Costs Optimally is PSPACE-hard

Next, we turn our attention to proving a matching lower bound, which already holds for finitary parity games, i.e., parity games with costs in which every edge is an increment-edge. The result is proven by a reduction from the canonical PSPACE-hard problem QBF: given a quantified boolean formula $\varphi = Q_1 x_1 Q_2 x_2 \dots Q_n x_n \psi$ with $Q_i \in \{\exists, \forall\}$ and where ψ is a boolean formula over the variables $x_1, x_2, \dots, x_n$, determine whether φ evaluates to true. We assume w.l.o.g. that ψ is in conjunctive normal form such that every conjunct has exactly three literals, i.e., $\psi = \bigwedge_{j=1}^{m} (\ell_{j,1} \vee \ell_{j,2} \vee \ell_{j,3})$, where every $\ell_{j,k}$ is either x_i or $\overline{x_i}$ for some i. We call each $\ell_{j,k}$ for $k \in \{1,2,3\}$ a literal and each conjunct of three literals a clause. Furthermore, we assume w.l.o.g. that the quantifiers Q_i are alternating with $Q_1 = Q_n = \exists$.

▶ **Lemma 7.** *The following problem is* PSPACE*-hard: "Given a finitary parity game $\mathcal{G}$ and a bound $b \in \mathbb{N}$, does Player 0 have a strategy σ for $\mathcal{G}$ with $\mathrm{Cst}(\sigma) \leq b$?"*

Proof. Let $\varphi = Q_1 x_1 Q_2 x_2 \dots Q_n x_n \psi$ be a quantified boolean formula with $\psi = \bigwedge_{j=1}^{m} C_j$ and $C_j = (\ell_{j,1} \vee \ell_{j,2} \vee \ell_{j,3})$, where every $\ell_{j,k}$ is either x_i or $\overline{x_i}$ for some i. We construct a finitary parity game $\mathcal{G}_\varphi$ such that Player 0 has a strategy σ for $\mathcal{G}_\varphi$ with $\mathrm{Cst}_{\mathcal{G}_\varphi}(\sigma) = 3n + 5$ if and only if the formula φ evaluates to true. The arena consists of three parts: In the first part, which begins with the initial vertex v, Player 0 and Player 1 determine an assignment for the variables x_1 through x_n, where Player 0 and Player 1 pick values for the existentially and universally quantified variables, respectively. Each choice of a truth value by either player incurs a request. In the second part, Player 1 first picks a clause, after which Player 0 picks a literal from that clause. In the last part, the play then proceeds without any choice by the players and checks whether or not that literal was set to true in the first part of the arena. If it was set to true, then its corresponding request is answered with cost $3n + 5$. Otherwise, its corresponding request is answered with cost $3n + 6$. Furthermore, all other potentially open requests are answered with cost at most $3n + 5$ and the play returns to the initial vertex v. Thus, all these gadgets are traversed infinitely often and the traversals are independent of each other.

If φ evaluates to true, then Player 0 can always enforce a play in which all requests are answered with cost at most $3n + 5$. Hence, there exists a strategy σ for Player 0 with $\mathrm{Cst}_{\mathcal{G}_\varphi}(\sigma) \leq 3n + 5$. If φ evaluates to false, however, then Player 1 can enforce requests that remain unanswered for at least $3n + 6$ steps. Thus, there exists no strategy σ for Player 0 with $\mathrm{Cst}_{\mathcal{G}_\varphi}(\sigma) \leq 3n + 5$. We begin by constructing the arena $\mathcal{A}$ together with its coloring Ω.

The left-hand side of Figure 3 shows the gadgets that assign a truth-value to variable x_j. The vertex a_j belongs to Player 0 if x_j is existentially quantified, and to Player 1 if x_j is universally quantified. The dashed edges indicate the connections to the pre- and succeeding gadget, respectively. We construct the first part of $\mathcal{A}$ out of n copies of this gadget. Moreover, the vertex a_1 has an incoming edge from the end of $\mathcal{A}$, in order to allow for infinite plays, and is the initial vertex v of the arena. In the remainder of this proof, let $c_{x_j} = 4j + 3$ and $c_{\overline{x_j}} = 4j + 1$ be the colors associated with assigning true or false to x_j, respectively.

The second part of the arena starts with a vertex ψ of Player 1, from which he picks a clause by moving to a vertex C_j of Player 0. Each vertex C_j is connected to three gadgets, one for each of the three literals contained in C_j. We show this construction in the right-hand side of Figure 3. Note that the distance between a vertex of color c_{x_j} and the vertex ψ is $3(n - j) + 1$, whereas the distance between a vertex of color $c_{\overline{x_j}}$ and the vertex ψ is $3(n - j) + 2$.

The last part of the arena consists of one gadget for each literal x_1, $\overline{x_1}$ through x_n, $\overline{x_n}$ occurring in φ. These gadgets check whether or not the literal picked in the middle part was

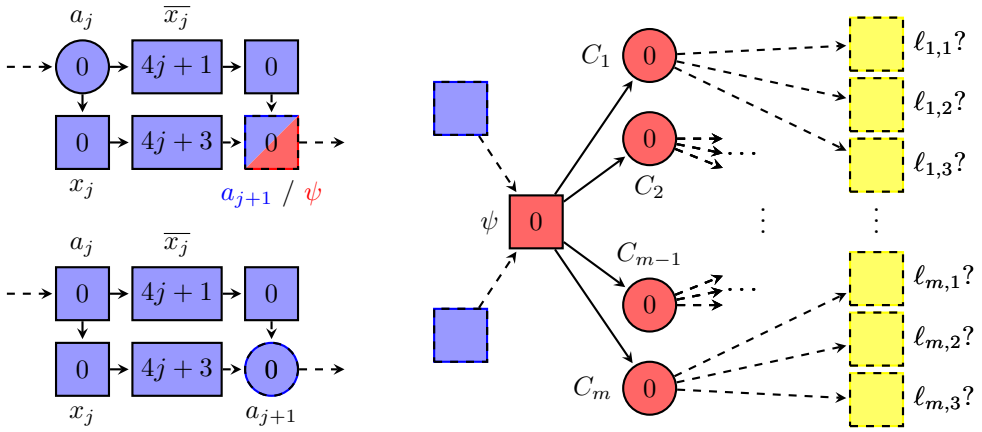

Figure 3 The gadget for existentially and universally quantified variables (left, from top to bottom), and the middle part of the arena (right).

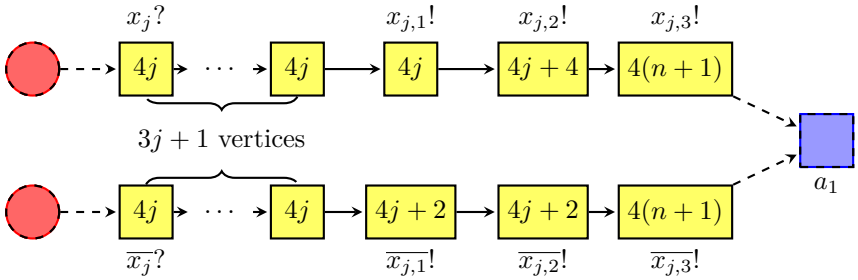

Figure 4 Gadgets checking the assignment of true (top) or false (bottom) to x_j.

actually set to true in the first part of the arena. We show them in Figure 4. Neither player has any control over the play in these gadgets.

Instead, it is determined by their structure. The play proceeds by first answering requests for all colors smaller than c_{x_j} and $c_{\overline{x_j}}$. It then either grants the request for color c_{x_j} after $3j+2$ steps, or the request for color $c_{\overline{x_j}}$ after $3j+1$ steps, both counted from the vertices $x_j?$ and $\overline{x_j}?$, respectively. Since traversing the middle part of the arena incurs a constant cost of 2, a request for color c_{x_j} has incurred a cost of $3(n-j)+3$ at $x_j?$ and $\overline{x_j}?$, while a request for color $c_{\overline{x_j}}$ has incurred a cost of $3(n-j)+4$ at these vertices. Hence, the total cost incurred by the request for color c_{x_j} is $(3(n-j)+3)+(3j+2)$ in the gadget corresponding to x_j, and $(3(n-j)+3)+(3j+3)$ in the gadget corresponding to $\overline{x_j}$. The dual reasoning holds true for requests for color $c_{\overline{x_j}}$. Hence, the bound of $3n+5$ is only achieved if the request corresponding to the chosen literal was posed in the initial part of the arena.

After traversing this last gadget, all requests are answered and the game resets to the initial state via an edge to a_1.

The size of $\mathcal{A}$ is polynomial in the size of φ: The first part consists of one constant-size gadget for each variable, while the second part is of linear size in the number of clauses in φ. The final part contains a gadget of size $\mathcal{O}(n)$ for each literal occurring in φ. Thus, the size of the arena is in $\mathcal{O}(n^2+m)$. It remains to argue that Player 0 has a strategy σ with $\mathrm{Cst}_{\mathcal{G}_\varphi}(\sigma) = 3n+5$ if and only if φ evaluates to true. For any quantifier-free boolean formula ψ that contains variables x_1 through x_n and any partial assignment $\alpha\colon \{x_1,\dots,x_n\} \dashrightarrow \{\mathit{true}, \mathit{false}\}$ we denote by $\alpha(\psi)$ the formula resulting from replacing the

variables in α's domain with their respective truth values.

It suffices to argue about finite plays that begin and end in a_1, as all plays start in a_1, visit a_1 infinitely often, and all requests are answered before moving back to a_1. Hence, for the remainder of this proof, we only consider a finite play infix ρ starting and ending in a_1.

First assume that φ evaluates to true. We construct a strategy σ for Player 0 with the properties described above. Pick j as some index such that x_j is existentially quantified and consider the play prefix ρ' of ρ up to, but not including a_j. We associate ρ' with an assignment $\alpha_{j-1}\colon \{x_1,\dots,x_{j-1}\} \to \{\mathit{true},\mathit{false}\}$, where $\alpha_{j-1}(x_k) = \mathit{true}$ if there is an infix $a_k x_k$ in ρ', and $\alpha_{j-1}(x_k) = \mathit{false}$ if there is an infix $a_k \overline{x_k}$ in ρ'. Due to the structure of the arena exactly one of these cases holds true, hence α_{j-1} is well-defined.

For $j = 1$, $\exists x_j \cdots Q_n x_n \alpha_{j-1}(\psi)$ evaluates to true by assumption. Let $t \in \{\mathit{true},\mathit{false}\}$ such that $\forall x_{j+1} \cdots Q_n x_n (\alpha_{j-1}[x_j \mapsto t])(\psi)$ evaluates to true as well, where $\alpha_{j-1}[x_j \mapsto t]$ denotes the mapping α_{j-1} augmented by the mapping $x_j \mapsto t$. Moreover, we define $\sigma(wa_j) = x_j$ if $t = \mathit{true}$, and $\sigma(wa_j) = \overline{x_j}$ otherwise. We proceed inductively, constructing $\sigma(wa_j)$ for all existentially quantified variables x_j according to the boolean values that satisfy the formulas $\exists x_j Q_{j+1} x_{j+1} \dots Q_n x_n \alpha_{j-1}(\psi)$, until we reach the vertex ψ.

At this point, the analysis of the play prefix so far yields an assignment α_n, which is a total function mapping each variable of x_1 through x_n to either *true* or *false*, such that $\alpha_n(\psi)$ evaluates to true. We define $\alpha = \alpha_n$.

At vertex ψ there exist n open requests. As previously argued, if $\alpha(x_j) = \mathit{true}$, then there is an open request for c_{x_j} with cost $3(n-j)+1$. Otherwise, there is an open request for $c_{\overline{x_j}}$ with cost $3(n-j)+2$. At vertex ψ, Player 1 picks a clause C_j by moving to its vertex. Since $\alpha(\psi) \equiv \mathit{true}$, there must exist a $k \in \{1,2,3\}$ with $\alpha(\ell_{j,k}) = \mathit{true}$. We pick $\sigma(wC_j) = \ell_{j,k}?$.

If $\ell_{j,k} = x_l$, then $\alpha(x_l) = \mathit{true}$ and hence, there is an open request for c_{x_j}. As argued previously, this request is then answered with cost $3n+5$, since we picked the gadget corresponding to x_j. Similarly, if $\ell_{j,k} = \overline{x_l}$, then $\alpha(x_l) = \mathit{false}$ and thus there is an open request for $x_{\overline{x_j}}$, which is answered with cost $3n+5$ as well. All other open requests are answered with cost at most $3n+5$, as argued previously.

After this traversal of the final gadget, all requests are answered, and the play automatically moves to vertex a_1 to begin anew. The same reasoning then applies ad infinitum. Thus, Player 0 is able to answer all requests with a cost of at most $3n+5$.

Now assume that φ evaluates to false. Then, irrespective of the choices made by Player 0 when constructing α in the first part of the arena, Player 1 can construct an α such that $\alpha(\psi) \equiv \mathit{false}$ and then pick a clause C_j with $\alpha(C_j) \equiv \mathit{false}$. Hence, Player 0 has to pick some $\ell_{j,k}$ with $\alpha(\ell_{j,k}) = \mathit{false}$. If $\ell_{j,k} = x_l$, then there is an open request for $c_{\overline{x_j}}$ at $x_{l,1}!$, which is answered with cost $3n+6$. Similarly, if $\ell_{j,k} = \overline{x_l}$, then $\alpha(x_l) = \mathit{true}$, hence there is an open request for c_{x_j}, which is also answered with cost $3n+6$. Thus, in each round Player 1 can open a request that is answered with cost at least $3n+6$, i.e., Player 0 has no strategy with cost $3n+5$.

◀

5 Memory Requirements of Optimal Strategies in Parity Games with Costs

Next, we study the memory requirements of optimal strategies in parity games with costs. Recall that Player 0 always has a positional winning strategy if she wins the game. In contrast, our main result of this section shows that the memory requirements of optimal strategies are exponential, i.e., playing optimally comes at a price in terms of memory,

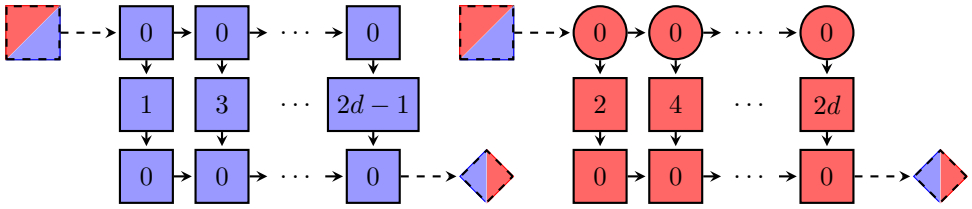

Figure 5 The gadgets for Player 1 (left) and Player 0 (right).

too. Our lower bound is obtained by a generalization of a construction of Chatterjee and Fijalkow [8] which yielded a linear lower bound.

First, however, let us state a corollary of the construction of the parity game $\mathcal{G}'$ in the proof of Lemma 4, which gives an exponential upper bound on the necessary memory states. Recall that the memory structure used in that proof has one counter with a range of size $b+2$ for each odd color. Furthermore, it has an additional counter that is bounded by n, which counts the number of times the bound b is exceeded.

▶ **Corollary 8.** *Let $\mathcal{G}$ be a parity game with costs. If Player 0 has a strategy σ for $\mathcal{G}$ with $\mathrm{Cst}_{\mathcal{G}}(\sigma) = b$, then she also has a strategy σ' with $\mathrm{Cst}_{\mathcal{G}}(\sigma') \leq b$ and $|\sigma'| = (b+2)^d \cdot (n+1)$, where n is the number of vertices and d the number of odd colors in $\mathcal{G}$.*

Using similar techniques to [16], it is possible to remove the overflow counter, since it is the goal of Player 0 to avoid excesses of the bound b. Hence, she can play assuming the largest value for this counter that still allows her to win. Again, our matching lower bound already holds for finitary parity games, i.e., parity games with costs in which all edges are increment-edges.

▶ **Theorem 9.** *For every $d > 1$, there exists a finitary parity game $\mathcal{G}_d$ such that*

- *$\mathcal{G}_d$ has d odd colors and $|\mathcal{G}_d| \in \mathcal{O}(d^2)$, and*
- *every optimal strategy for Player 0 in $\mathcal{G}_d$ has at least size $2^d - 2$.*

Proof. Let $d > 1$. We construct a finitary parity game $\mathcal{G}_d$ that has the stated properties. To this end, we first construct an optimal strategy for Player 0 and argue that every such strategy has cost $d^2 + 2d$, followed by the proof that every optimal strategy has at least size $2^d - 2$.

The game $\mathcal{G}_d$ is played in rounds. In each round, which starts at the initial vertex of $\mathcal{G}_d$, Player 1 poses d requests for odd colors in the range 1 through $2d-1$. Subsequently, Player 0 gives d answers using colors in the range 2 through $2d$. After each round, the play returns to the initial vertex in order to allow for infinite plays.

The arena $\mathcal{A}$ consists of gadgets that allow exactly one request or response to be made. Moreover, each path through a gadget has the same length. However, low-priority requests and responses must be made earlier in the traversal of such a gadget than high-priority ones. We show both gadgets in Figure 5. The dashed lines show the connection to the pre- and succeeding gadget and the connection between the final and the initial gadget. As the owner of the succeeding vertex depends on the succeeding gadget's owner, we draw it as a diamond.

The arena $\mathcal{A}$ consists of d repetitions of the gadget for Player 1, followed by d repetitions of the gadget for Player 0. The initial vertex v of the arena is the top-left node of the first gadget for Player 1. Moreover, the final gadget of Player 0 has a single back-edge to the initial vertex. Clearly, $\mathcal{A}$ satisfies the first statement of the theorem.

Similarly to the proof of Lemma 7, it suffices to consider finite plays infixes. Even though the requests are not necessarily all answered after each round, we argue that Player 0 can always do so while playing optimally.

We now construct an optimal strategy from v for Player 0. In order to play optimally, Player 0 needs to track the requests made by Player 1 in the first part of each round. Instead of tracking each request precisely, however, it suffices to only store those requests that are of higher priority than all previous ones in the current round. Moreover, by defaulting to visiting the vertex of color $2d$ after having answered all requests, it suffices to store at most $d-1$ requests. If there is a repetition in the requests made, then the final request will be answered by an answer previous to the final one. If there is no repetition, the final request is for $2d-1$ and will be answered by the default answer of $2d$. We define the set of strictly increasing odd sequences $\mathit{IncSeq}_d = \{(c_1, \ldots, c_k) \mid 1 \leq c_1 < \cdots < c_k \leq 2d-1, \text{ all } c_i \text{ are odd}\}$ and use the set of memory states $M_d = \mathit{IncSeq}_d \setminus \{(), (1, 3, \ldots, 2d-1)\}$. Note that $|M_d| = 2^d - 2$.

Each round starts with Player 0 observing the requests of Player 1. While the first request of Player 1 replaces the entry of the initial memory state (1), subsequent requests are only appended if they are larger than the last element of the current memory state. During her turn, Player 0 then pops the first element of the sequence, call it c, in each of her gadgets and answers that request by moving to $c+1$. After popping the final request in her memory state, Player 0 pushes a request for $2d-1$ and answers with $2d$ for the remainder of her turn.

Now consider a play in which Player 0 plays according to this strategy and consider the request for color c made by Player 1 in his j-th gadget. If Player 1 has posed strictly increasing requests up to j, then Player 0 answers the j-th request from Player 1 in her j-th gadget. The cost of this request then consists of three components. First, the play has to leave Player 1's j-th gadget, incurring a cost of $(2d-1-c)/2+2$. Then, the play passes through $d-1$ gadgets, each incurring a cost of $d+2$. Finally, moving to $c+1$ in Player 0's gadget incurs a cost of $(c-1)/2+1$ Hence, answering Player 1's request incurs a cost of d^2+2d. If Player 1 has, however, not posed requests in strictly increasing order up to j, then the request will be answered in some gadget $j' < j$ of Player 0. Hence, Player 0 answers the request with cost at most $(d-1)(d+2) < d^2+2d$.

This cost is indeed optimal. Consider the play in which Player 1 always requests $2d-1$. Even if Player 0 answers this request in her first gadget, it still incurs a cost of d^2+2d.

It remains to show that no optimal strategy of size less than $|M_d|$ exists. We associate with each $s \in M_d$ the partial play $\mathit{req}(s)$, which starts in the initial vertex, where Player 1 requests the colors occurring in s in order. Subsequently, he requests the final color of s until the end of his turn. Hence, for $s \neq s' \in M_d$, we have $\mathit{req}(s) \neq \mathit{req}(s')$.

Pick a strategy σ for Player 0 that is implemented by $\mathcal{M} = (M, m_I, \mathrm{Upd})$ with $|M| < |M_d|$. Let $m \in M$. Due to the pigeon-hole principle, there exist $s \neq s' \in M_d$, such that $\mathrm{Upd}^+(m, \mathit{req}(s)) = \mathrm{Upd}^+(m, \mathit{req}(s'))$. Hence, Player 0 answers both sequences of requests in the same way. Since $\mathit{req}(s) \neq \mathit{req}(s')$, there exists a gadget of Player 1 in which the requests posed during $\mathit{req}(s)$ and $\mathit{req}(s')$ differ. Pick j as the minimal index of such gadgets and assume that in his j-th gadget, Player 1 requests color c during $\mathit{req}(s)$, and color c' during $\mathit{req}(s')$, where, w.l.o.g., $c < c'$. If Player 0 has already answered either the request for c or the request for c' before her j-th gadget, then some earlier request is not answered optimally. Thus, assume that neither request has been answered upon entering Player 0's j-th gadget. If she visits some color $c'' < c'$ in this gadget, she will only answer c' in some later gadget, thereby incurring a cost of at least $(d+1)(d+2)$. If she visits some color $c'' > c'$, then she does not answer the request for c optimally, thus incurring a cost of at least $d^2+2d+c'-c$. Hence, one of the sequences of requests s or s' leads to Player 0 answering at least one

request non-optimally. As there exist such sequences s and s' for each $m \in M$, Player 1 can force such a costly request in each round. Thus, $\mathrm{Cst}(\sigma) > d^2 + 2d$, i.e., σ is not optimal. ◀

6 Tradeoffs Between Time and Memory

In the previous section, we have shown that an optimal strategy for Player 0 in a parity game with costs requires exponential memory in general. In contrast, minimal winning strategies for Player 0 in parity games with costs are known to be positional [16]. Here we show that, in general, there exists a gradual tradeoff between the size and the cost of a strategy.

▶ **Theorem 10.** *Fix some $d > 1$ and let the game $\mathcal{G}_d$ be as defined in the proof of Theorem 9. For every i with $1 \leq i \leq d$ there exists a strategy σ_i for Player 0 in $\mathcal{G}_d$ such that*

- $d^2 + 3d - 1 = \mathrm{Cst}_{\mathcal{G}_d}(\sigma_1) > \mathrm{Cst}_{\mathcal{G}_d}(\sigma_2) > \cdots > \mathrm{Cst}_{\mathcal{G}_d}(\sigma_d) = d^2 + 2d$, *and*
- $1 = |\sigma_1| < |\sigma_2| < \cdots < |\sigma_d| = 2^d - 2$.

Also, for every strategy σ' for Player 0 in $\mathcal{G}_d$ with $\mathrm{Cst}_{\mathcal{G}_d}(\sigma') \leq \mathrm{Cst}_{\mathcal{G}_d}(\sigma_i)$ we have $|\sigma'| \geq |\sigma_i|$.

Proof. Recall that we defined the set of strictly increasing odd sequences IncSeq_d in Theorem 9 and showed that a memory structure using $\mathit{IncSeq}_d \setminus \{(), (1, 3, \ldots, 2d-1)\}$ as memory states implements an optimal strategy with cost $d^2 + 2d$. Intuitively, such a strategy stores up to $d - 1$ requests made by Player 1 in his part of each round. The idea behind the construction of the strategies σ_i is to restrict the memory of Player 0 such that she can only store up to i requests. In the extremal cases of $i = 1$ and $i = d$ this implements a positional strategy and the strategy from the proof of Theorem 9, respectively.

We implement σ_i by again using strictly increasing odd sequences, where we restrict the maximal length, but not the maximal value of entries in the memory states for Player 0. In strategy σ_i, Player 0 stores at most $i - 1$ requests, using sequences of length at most $i - 1$.

To this end, we define the length-restricted set of strictly increasing odd sequences $\mathit{IncSeq}_d^i = \{s \in \mathit{IncSeq}_d \mid |s| < i\}$, where $|s|$ denotes the number of elements contained in the sequence s. If $i > 1$, then we may remove the empty sequence from the set of memory states, since Player 1 has to pose at least one request in each of his turns.

Otherwise, Player 0's memory consists only of the empty sequence.

Hence, we pick $M_d^i = \mathit{IncSeq}_d^i \setminus \{()\}$ for $i > 1$. For $i = 1$, however, we define $M_d^1 = \mathit{IncSeq}_d^1 = \{()\}$. Note that $M_d^d = M_d$ as defined in the proof of Theorem 9. Clearly, the claim about the size of the σ_j holds true, since $|M_d^1| < |M_d^2|$ and $M_d^i \subsetneq M_d^{i+1}$ for each $d > 1$, $i > 1$. The initial memory state, the update function, and the next-move function for Player 0 are defined similarly to those from the proof of Theorem 9 in order to obtain the memory structure $\mathcal{M}_i$ implementing σ_i. In particular, after answering all requests stored in her memory state, Player 0 defaults to visiting $2d$ repeatedly in order to answer any requests by Player 1 that were not stored in her memory.

It remains to show that each strategy σ_j realizes a cost of $d^2 + 3d - i$ and that each σ_i is minimal for its respective cost. To this end, we fix some i with $1 \leq i \leq d$ for the remainder of this proof. First, we show that Player 1 can enforce a cost of $d^2 + 3d - i$ if Player 0 plays consistently with σ_i. Intuitively, Player 1 fills the memory of Player 0 as quickly as possible, and requests the minimal color that has not yet been requested repeatedly afterwards. Thus, he maximizes the gap between the smallest unstored request and the default answer of $2d$.

More precisely, in each turn Player 1 requests the colors $1, 3, \ldots, 2i-3, 2i-1, 2i-1, \ldots$. Playing consistently with σ_i, Player 0 answers these requests with $2, 4, \ldots, 2i-2, 2d, 2d, \ldots$. In general,the cost of the resulting play is the cost incurred by answering a request for $2i - 1$ in the i-th gadget of Player 1 with $2d$ in the i-th gadget of Player 0. As argued in the proof

of Theorem 9, the cost incurred by such a request-response-pair amounts to

$$[(2d-1-(2i-1))/2+2]+[(d-1)(d+2)]+[(2d-2)/2+1]=d^2+3d-i.$$

As the game restarts after Player 0's turn, Player 1 can enforce this cost infinitely often. Hence, $\mathrm{Cst}_{\mathcal{G}_d}(\sigma_i) \geq d^2+3d-i$.

This sequence of requests is indeed optimal for Player 1, i.e., he cannot enforce a higher cost. Assume that Player 1 does not pose requests as specified above, but poses the requests $c_1,\ldots,c_d$. Then either there exist some j and j' with $j<j'$, such that $c_j \geq c_{j'}$, or there exists a j with $2i-1 < c_j \leq 2d-1$. In the former case, after encountering the first such j', Player 0 can answer all remaining requests with costs at most $(d-1)(d+2)$, as she can ignore the request for $c_{j'}$. In the latter case, Player 0 answers that request with cost at most $d^2+2d+(2d-1-c_j)/2 \leq d^2+3d-i$, independent of whether she was able to store it. Hence, there exists no play ρ consistent with σ_i and $\mathrm{Cst}(\rho) > d^2+3d-i$.

In general,the cost of the resulting play is the cost incurred by answering a request for $2i-1$ in the i-th gadget of Player 1 with $2d$ in the i-th gadget of Player 0. As argued in the proof of Theorem 9, the cost incurred by such a request-response-pair amounts to

$$[(2d-1-(2i-1))/2+2]+[(d-1)(d+2)]+[(2d-2)/2+1]=d^2+3d-i.$$

As the game restarts after Player 0's turn, Player 1 can enforce this cost infinitely often. Hence, $\mathrm{Cst}_{\mathcal{G}_d}(\sigma_i) \geq d^2+3d-i$.

This sequence of requests is indeed optimal for Player 1, i.e., he cannot enforce a higher cost. Assume that Player 1 does not pose requests as specified above, but poses the requests $c_1,\ldots,c_d$. Then either there exist some j and j' with $j<j'$, such that $c_j \geq c_{j'}$, or there exists a j with $2i-1 < c_j \leq 2d-1$. In the former case, after encountering the first such j', Player 0 can answer all remaining requests with costs at most $(d-1)(d+2)$, as she can ignore the request for $c_{j'}$. In the latter case, Player 0 answers that request with cost at most $d^2+2d+(2d-1-c_j)/2 \leq d^2+3d-i$, independent of whether she was able to store it. Hence, there exists no play ρ consistent with σ_i and $\mathrm{Cst}(\rho) > d^2+3d-i$.

As the final part of the proof, we observe that there exists no strategy σ' with $|\sigma'| < |\sigma_i|$ and $\mathrm{Cst}_{\mathcal{G}_d}(\sigma') \leq \mathrm{Cst}_{\mathcal{G}_d}(\sigma_i)$. The argument is nearly identical to the argument of minimality of the strategy constructed in the proof of Theorem 9 and can in fact be obtained by replacing all occurrences of 2^d-2 and d^2+2d by $|\sigma_i|$ and d^2+3d-i, respectively, and by having Player 1 request $2i-1$ instead of the final entry of s after requesting all colors in s in that proof. Hence, the strategies σ_i are minimal for their respective cost. ◀

7 Conclusion

In this work we have shown that playing parity games with costs optimally is harder than just winning them, both in terms of computational complexity as well as in terms of memory requirements of strategies. We proved checking an upper bound on the value of an optimal strategy to be complete for polynomial space. Moreover, we have shown that optimal strategies in general require exponential memory, but also that exponential memory is always sufficient to implement optimal strategies. Finally, we have shown that, in general, there exists a gradual tradeoff between the size and the cost of strategies.

All these results also hold true for the case of bounded parity games and bounded parity games with costs [9, 16]. While the parity condition with costs only restricts the cost-of-response in the limit, the bounded parity condition prohibits any unanswered request with cost ∞ (but still allows finitely many unanswered requests with finite cost).

There are at least two directions in which these results can be extended, namely towards Streett conditions (finitary or with costs) [9, 16] and towards a binary weights.

In current work, we investigate the former extension. Our lower bounds carry over trivially, as every parity condition is a Streett condition of polynomial size. On contrast, the upper bounds are more involved, since solving the boundedness question for finitary Streett games is already ExpTime-complete and exponential memory is necessary for Player 0 (see [16] for a discussion).

Throughout this work, we have only considered unary weights, i.e., cost functions that assign the abstract costs ε and i. Allowing arbitrary non-negative costs would constitute an extension of the model considered here, i.e., the PSpace-hardness result as well as the necessity of exponential memory remain true without any modifications. Again, the upper bounds are non-trivial, as the upper bound on the cost of an optimal strategy is now exponential in the size of the game and its largest weight. Thus, e.g., there is a blowup in the size of $\mathcal{G}'$ as constructed in the proof of Theorem 4 and in the play length of its finite-duration variant $\mathcal{G}'_f$. We are currently investigating whether these can be avoided.

References

1 Rajeev Alur, Kousha Etessami, Salvatore La Torre, and Doron Peled. Parametric temporal logic for "model measuring". *ACM Trans. Comput. Log.*, 2(3):388–407, 2001.

2 Benjamin Aminof and Sasha Rubin. First cycle games. In Fabio Mogavero, Aniello Murano, and Moshe Y. Vardi, editors, *SR 2014*, volume 146 of *EPTCS*, pages 83–90, 2014.

3 Patricia Bouyer, Ulrich Fahrenberg, Kim Guldstrand Larsen, Nicolas Markey, and Jirí Srba. Infinite runs in weighted timed automata with energy constraints. In Franck Cassez and Claude Jard, editors, *FORMATS 2008*, volume 5215 of *LNCS*, pages 33–47. Springer, 2008.

4 Véronique Bruyère, Quentin Hautem, and Mickael Randour. Window parity games: an alternative approach toward parity games with time bounds. *arXiv*, 1606.01831, 2016.

5 Arindam Chakrabarti, Luca de Alfaro, Thomas A. Henzinger, and Mariëlle Stoelinga. Resource interfaces. In Rajeev Alur and Insup Lee, editors, *EMSOFT 2003*, volume 2855 of *LNCS*, pages 117–133. Springer, 2003.

6 Ashok K. Chandra, Dexter Kozen, and Larry J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, 1981.

7 Krishnendu Chatterjee, Luca de Alfaro, and Thomas A. Henzinger. Trading memory for randomness. In Gethin Norman and William Sanders, editors, *QEST 2004*, pages 206–217. IEEE, 2004.

8 Krishnendu Chatterjee and Nathanaël Fijalkow. Infinite-state games with finitary conditions. *arXiv*, 1301.2661, 2013.

9 Krishnendu Chatterjee, Thomas A. Henzinger, and Florian Horn. Finitary winning in ω-regular games. *ACM Trans. Comput. Log.*, 11(1), 2009.

10 Krishnendu Chatterjee, Thomas A. Henzinger, and Florian Horn. The complexity of request-response games. In Adrian-Horia Dediu, Shunsuke Inenaga, and Carlos Martín-Vide, editors, *LATA 2011*, pages 227–237. Springer, 2011. URL: `http://dx.doi.org/10.1007/978-3-642-21254-3_17`.

11 Krishnendu Chatterjee, Thomas A. Henzinger, and Vinayak S. Prabhu. Trading infinite memory for uniform randomness in timed games. In Magnus Egerstedt and Bud Mishra, editors, *HSCC 2008*, volume 4981 of *LNCS*, pages 87–100. Springer, 2008.

12 Andrzej Ehrenfeucht and Jan Mycielski. Positional strategies for mean payoff games. *Int. J. Game Theory*, 8:109–113, 1979.

13 E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *FOCS 1991*, pages 368–377. IEEE, 1991.

14 Peter Faymonville and Martin Zimmermann. Parametric linear dynamic logic. In Adriano Peron and Carla Piazza, editors, *GandALF 2014*, volume 161 of *EPTCS*, pages 60–73, 2014.

15 Nathanaël Fijalkow, Florian Horn, Denis Kuperberg, and Michal Skrzypczak. Trading bounds for memory in games with counters. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *ICALP 2015, Part II*, volume 9135 of *LNCS*, pages 197–208. Springer, 2015.

16 Nathanaël Fijalkow and Martin Zimmermann. Parity and Streett Games with Costs. *LMCS*, 10(2), 2014.

17 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research*, volume 2500 of *LNCS*. Springer, 2002.

18 Florian Horn. Streett games on finite arenas. In *GDV 2005*, 2005.

19 Florian Horn, Wolfgang Thomas, Nico Wallmeier, and Martin Zimmermann. Optimal strategy synthesis for request-response games. *RAIRO - Theor. Inf. and Applic.*, 49(3):179–203, 2015.

20 Marcin Jurdziński. Deciding the winner in parity games is in UP ∩ co-UP. *Inf. Process. Lett.*, 68(3):119–124, 1998.

21 Orna Kupferman, Nir Piterman, and Moshe Y. Vardi. From liveness to promptness. *Form. Met. in Sys. Des.*, 34(2):83–103, 2009.

22 Martin Lang. Resource reachability games on pushdown graphs. In Anca Muscholl, editor, *FOSSACS 14*, volume 8412 of *LNCS*, pages 195–209. Springer, 2014.

23 Fabio Mogavero, Aniello Murano, and Loredana Sorrentino. On promptness in parity games. *Fundam. Inform.*, 139(3):277–305, 2015.

24 Andrzej Mostowski. Games with forbidden positions. Technical Report 78, University of Gdańsk, 1991.

25 Nir Piterman and Amir Pnueli. Faster solutions of rabin and streett games. In *LICS 2006*, pages 275 – 284. IEEE, 2006.

26 Amir Pnueli and Roni Rosner. On the synthesis of an asynchronous reactive module. In Giorgio Ausiello, Mariangiola Dezani-Ciancaglini, and Simona Ronchi Della Rocca, editors, *ICALP 1989*, volume 372 of *LNCS*, pages 652–671. Springer, 1989.

27 Mickael Randour. *Synthesis in Multi-Criteria Quantitative Games*. PhD thesis, University of Mons, 2014.

28 Robert S. Streett. Propositional dynamic logic of looping and converse. In *STOC 1981*, pages 375 – 383. ACM, 1981.

29 Alexander Weinert and Martin Zimmermann. Easy to win, hard to master: Optimal strategies in parity games with costs. *arXiv*, 1604.05543, 2016.

30 Ernst Zermelo. Über eine Anwendung der Mengenlehre auf die Theorie des Schachspiels. In *Proc. Fifth Congress of Mathematicians, Vol. 2*, pages 501–504. Cambridge Press, 1913.

31 Martin Zimmermann. Time-optimal winning strategies for poset games. In Sebastian Maneth, editor, *CIAA 2009*, volume 5642 of *LNCS*, pages 217–226. Springer, 2009.

32 Martin Zimmermann. Optimal Bounds in Parametric LTL Games. *Theoret. Comput. Sci.*, 493(0):30 – 45, 2013.

33 Martin Zimmermann. Parameterized linear temporal logics meet costs: Still not costlier than LTL. In Javier Esparza and Enrico Tronci, editors, *GandALF 2015*, volume 193 of *EPTCS*, pages 144–157, 2015.

34 Uri Zwick and Mike Paterson. The complexity of mean payoff games. In Ding-Zhu Du and Ming Li, editors, *COCOON 1995*, volume 959 of *LNCS*, pages 1–10. Springer, 1995.

A Appendix

Here, we present the proofs omitted in the main part showing

- the equivalence of the parity game with costs $\mathcal{G}$ with respect to the bound b and the parity game $\mathcal{G}'$ (Proposition 5), and
- and the equivalence of $\mathcal{G}'$ and its finite-duration variant $\mathcal{G}'_f$ (Proposition 6).

To simplify the proofs, we introduce some notation. Recall that both $\mathcal{G}'$ and $\mathcal{G}'_f$ are played in the arena $\mathcal{A} \times \mathcal{M}$. To simplify our notation, let $\mathcal{A} \times \mathcal{M} = (V', V'_0, V'_1, E', v'_I)$, in particular $V' = V \times M$ and $V'_i = V_i \times M$.

Furthermore, if $\mathrm{Upd}((r, o), e) = (r', o')$ for some memory state (r, o) and some edge e, then we define $\mathrm{Upd}_r((r, o), e) = r'$ and $\mathrm{Upd}_o((r, o), e) = o'$.

Moreover, we define the request function r_v for some $v \in V$ as $r_v(c) = 0$, if $\Omega(v) = c$, and $\bot$ otherwise. In particular, r_{v_I} is the request function of the initial vertex of $\mathcal{G}'$ as defined above, i.e., $m_I = (r_{v_I}, 0)$.

Finally, let $(v_0, r_0, o_0)(v_1, r_1, o_1)(v_2, r_2, o_2) \cdots$ be a finite or infinite play in $\mathcal{A}'$. An *overflow position* is a j such that either $j = 0$ or $o_j = o_{j-1} + 1$. Note that we have $r_j = r_{v_j}$ for every overflow position, i.e., the request function is reset at each such position.

A.1 Proof of Proposition 5

Recall that we have to show that there exists a strategy σ for Player 0 in $\mathcal{G}$ with $\mathrm{Cst}(\sigma) \leq b$ if and only if Player 0 wins $\mathcal{G}'$.

Proof. First, let Player 0 win $\mathcal{G}'$, i.e., let $\sigma' \colon V'_0 \to V'$ be a positional winning strategy for her from v'_I in $\mathcal{G}'$. We define the finite-state strategy σ for Player 0 in $\mathcal{G}$ using $\mathcal{M}$ and the next-move function Nxt with $\mathrm{Nxt}(v, m) = v'$, if $\sigma'(v, m) = (v', m')$. Let $\rho = v_0 v_1 v_2 \cdots$ be a play that is consistent with σ. A straightforward induction shows that the unique extended play $\mathrm{ext}(\rho) = (v_0, r_0, o_0)(v_1, r_1, o_1)(v_2, r_2, o_2) \cdots$ in $\mathcal{G}'$ is consistent with σ' and therefore winning for Player 0. Hence, there is a position j such that $o_{j'} = o_j < n$ for every $j' > j$.

We claim $\mathrm{Cor}(\rho, k) \leq b$ for every $k > j$, which finishes this direction of the proof. Assume towards a contradiction that a request at some position after j is unanswered for $b + 1$ increment-edges and let k be such a position. We have $r_k(\Omega(v_k)) \geq 0$ and during every increment-edge, this counter is increased by one and not reset until $b + 1$ increment-edges are traversed, which triggers an overflow. This contradicts the choice of $k > j$. Thus, if ρ has infinitely many increment-edges, then almost every request is answered with cost at most b, i.e., $\mathrm{Cst}(\rho) \leq b$.

Now, consider the case where ρ has finitely many increment-edges. Such a play satisfies the parity condition if and only if $\mathrm{Cor}(\rho, k) = 0$ for almost all k, i.e., $\mathrm{Cst}(\rho) = 0$. Thus, it suffices to note that ρ and $\mathrm{ext}(\rho)$ coincide on their color sequences, as the overflow counter does not reach value n, and that $\mathrm{ext}(\rho)$ satisfies the parity condition, as it is winning for Player 0.

For the other direction, we prove the contrapositive. Assume that Player 0 does not win $\mathcal{G}'$. Then, due to determinacy of parity games, Player 1 wins $\mathcal{G}'$. Thus, let τ' be a winning strategy for Player 1 in $\mathcal{G}'$. We construct a strategy τ for Player 1 in $\mathcal{G}$ that enforces a play ρ with $\mathrm{Cst}(\rho) > b$ against every strategy for Player 0. Then, $\mathrm{Cst}(\sigma) > b$ for every strategy σ for Player 0 in $\mathcal{G}$.

To this end, we first construct a mapping h that simulates play prefixes in $\mathcal{G}$ by play prefixes in $\mathcal{G}'$. The mapping h maintains the following invariant:

For every play prefix π in $\mathcal{G}$ ending in a vertex v, $h(\pi) = (v_0, r_0, o_0) \cdots (v_j, r_j, o_j)$ is a play prefix in $\mathcal{G}'$ with $v_j = v$. Furthermore, $h(\pi)$ contains no two overflow positions $j_0 < j_1$ with $v_{j_0} = v_{j_1}$.

Note that $h(\pi)$ not containing two overflow positions with the same vertex implies $o_j < n$.

For $\pi = v_I$, we define $h(\pi) = v'_I$, which satisfies the invariant. Now let π be some play prefix in $\mathcal{G}$ ending in v and let $h(\pi)$ end in (v, r, o). We fix a successor v' of v and need to define $h(\pi \cdot v')$. In order to do so, we use two actions: Either we define $h(\pi \cdot v')$ by appending v' and an appropriate memory state to $h(\pi)$, or by cutting off a suffix of $h(\pi)$. We then define $h(\pi \cdot v')$ based on a case analysis: Either the move to v' at the end of $h(\pi)$ does not cause an overflow, i.e., $\mathrm{Upd}_o((r, o), (v, v')) = o$, or it does, i.e., $\mathrm{Upd}_o((r, o), (v, v')) = o + 1$ (recall that we have $o < n$ by assumption).

In the former case, we define $h(\pi \cdot v')$ by concatenating $h(\pi)$ and $(v', \mathrm{Upd}((r, o), (v, v')))$, which is a successor of (v, r, o). Furthermore, as we do not introduce a new overflow position, the second requirement of the invariant is satisfied as well. In the latter case, we again consider two possibilities: If $h(\pi)$ contains a vertex of the form $(v', r_{v'}, o')$ at an overflow position, which is unique by the invariant, we obtain $h(\pi \cdot v')$ by removing all later vertices from $h(\pi)$. This satisfies the invariant, in particular the second requirement, as the invariant is a prefix-closed property. If there is no such vertex in $h(\pi)$ then we again concatenate $h(\pi)$ and $(v', \mathrm{Upd}((r, o), (v, v')))$ to obtain $h(\pi \cdot v')$. We have to argue that this satisfies the invariant. In particular, assume there are two overflow positions that share the same vertex in their first component. As such a repetition did not appear in $h(\pi)$ by assumption, the last vertex $(v', \mathrm{Upd}((r, o), (v, v')))$ is part of this repetition. Hence, the first case is applicable and we have derived the desired contradiction.

Using the mapping h we define a strategy τ for Player 1 in $\mathcal{G}$ by $\tau(\pi) = v$, if $\tau'(h(\pi)) = (v, r, o)$. This is well-defined due to the invariant. A straightforward induction shows that if π is consistent with τ, then $h(\pi)$ is consistent with τ'.

It remains to show $\mathrm{Cst}(\rho) > b$ for every play that is consistent with τ. Fix such a play $\rho = v_0 v_1 v_2 \cdots$ and define $\pi_i = v_0 \cdots v_i$ for every i. First assume that there are infinitely many positions $i > 0$ such that $h(\pi_i)$ is obtained from $h(\pi_{i-1})$ by the removal operation. Let i be such a position, i.e., $h(\pi_i) = (v_0, r_0, o_0) \cdots (v_{j'}, r_{j'}, o_{j'})$ is a prefix of $h(\pi_{i-1}) = (v_0, r_0, o_0) \cdots (v_j, r_j, o_j)$. By definition of $\mathcal{M}$, there is suffix of π_i that caused the overflow that triggers the removal operation, i.e., there is a request in this suffix that is open for at least $b + 1$ increment-edges. As there are infinitely many such positions i and as the requests are all reset at overflow positions, there exist infinitely many such requests in ρ, and hence $\mathrm{Cst}(\rho) > b$.

Now assume that there are only finitely many π_i such that $h(\pi_i)$ is obtained using the removal operation and pick π_i as the longest such prefix. Then, for all $i' > i$, we have $h(\pi_{i'}) = h(\pi_{i'-1}) \cdot (v_{i'}, r_{i'}, o_{i'})$ for some $r_{i'}$ and some $o_{i'} < n$. Thus, define ρ' to be the limit of the $h(\pi_{i'})$ for $i < i' \to \infty$, which is a play in $\mathcal{G}'$. Then, as every prefix of ρ' is consistent with τ', ρ' is consistent with τ' and thus winning for Player 1. Also, as all $o_{i'}$ are strictly smaller than n due to the invariant of h, the coloring of ρ and the coloring of ρ' have a common suffix. As ρ' violates the parity condition, ρ violates this condition as well, and therefore also the parity condition with costs for any bound. Hence, $\mathrm{Cst}(\rho) > b$. ◀

A.2 Proof of Proposition 6

For this proof, we need to introduce even more notation and some simple facts about it.

As usual, a cycle in $\mathcal{A}$ is a play infix $v_0 \cdots v_j$ with $j > 0$ and $v_0 = v_j$. The cycle is an

ε-cycle, if its cost is zero, i.e., if it only contains ε-edges. We say that the cycle is even (odd), if its maximal color is even (odd). We lift this notion to play infixes in $\mathcal{A}'$ as follows: $(v_0, m_0, o_0) \cdots (v_j, m_j, o_j)$ is an even (odd) ε-cycle, if $v_0 \cdots v_j$ is an even (odd) ε-cycle. Note that this is not necessarily a cycle in $\mathcal{A}'$, since we allow $m_0 \neq m_j$ and $o_0 \neq o_j$. Also note that the existence of an odd ε-cycle settles a play.

Two request functions r_0 and r_1 are *on par*, if there is an odd color c^* such that

- $r_0(c^*) = r_1(c^*) \geq 0$,
- for every odd $c > c^*$: $r_0(c) = r_1(c)$, and
- for every odd $c < c^*$: either $r_0, r_1 \in \{\bot, 0\}$, or $r_0(c) = r_1(c)$.

Being on par is preserved by memory updates and two request functions that are on par have the same influence on the overflow counter.

▶ **Remark 11.** *Let r_0 and r_1 be on par and define $(r_0', o_0') = \mathrm{Upd}((r_0, o), e)$ and $(r_1', o_1') = \mathrm{Upd}((r_1, o), e)$ for some $o \leq n$ and some edge e. Then, r_0' and r_1' are on par and $o_0' = o_1'$.*

Now, we show that removing even ε-cycles can be captured by a pair of request functions that are on par.

▶ **Lemma 12.** *Let $(v_0, r_0, o_0) \cdots (v_j, r_j, o_j)$ be a play prefix in $\mathcal{A}'$ such that*

- *$(v_{j'}, r_{j'}, o_{j'}) \cdots (v_j, r_j, o_j)$ is an even ε-cycle, say with maximal color c_m, and*
- *$r_{j'-1}(c^*) \neq \bot$ for some $c^* > c_m$, i.e., a request of color c^* is open throughout the cycle.*

Then, $r_{j'}$ and r_j are on par and $o_{j'} = o_j$.

Proof. Identify the colors c^* in the definition of being on par and in the assumptions of the lemma and note that an ε-cycle can only have an overflow position at its first position. ◀

In particular, combining Lemma 12 and an inductive application of Remark 11 shows that removing even ε-cycles during which a request is continuously open does not influence the overflow counter.

In the following proof it is also useful to view a strategy for Player 1 in an arbitrary arena (V, V_0, V_1, E, v_I) as the set of play prefixes that are consistent with it. Such a set $P \subseteq V^*$ satisfies the following four properties:

1. P is prefix-closed.
2. $P \cap V = \{v_I\}$.
3. For every $wv \in P$ with $v \in V_1$, there is exactly one $v' \in V$ with $wvv' \in P$, which has to satisfy $(v, v') \in E$.
4. For every $wv \in P$ with $v \in V_0$ and every v' with $(v, v') \in E$, we have $wvv' \in P$.

The set of consistent play prefixes of a strategy for Player 1 satisfies these four properties and every set satisfying these properties can be turned into a strategy for Player 1.

Now we prove Proposition 6, i.e., that Player 0 wins $\mathcal{G}'$ if and only if she wins $\mathcal{G}_f'$.

Proof of Proposition 6. Due to determinacy of both games, it is sufficient to prove the equivalence for Player 1, i.e., that he wins $\mathcal{G}'$ if and only if he wins $\mathcal{G}_f'$

First, let Player 1 win $\mathcal{G}'$, say with the positional winning strategy τ'. Ideally, we would show that every play consistent with τ' is settled within $\ell + 1$ moves. While it is true that the play is eventually settled, it is in general not settled within $\ell + 1$ moves. The strategy may spend some time idling around before settling the play, even though it is positional. This is due to the fact that we have exponentially many vertices in $\mathcal{G}'$, which allows even a positional strategy to make exponentially many unproductive moves. Thus, in the following,

we show that we can transform τ' into a strategy τ'_f that does settle all consistent plays quickly. Thus, τ'_f is a winning strategy for Player 1 in $\mathcal{G}'_f$.

Let $P_{\tau'}$ be the set of play prefixes that are consistent with τ'. Fix one such $\pi = (v_0, r_0, o_0) \cdots (v_i, r_i, o_i) \in P_{\tau'}$ that is unsettled.

First we show that π contains no vertex repetition. If it does, consider the play ρ' obtained by reaching the induced cycle and then traversing it ad infinitum, which is consistent with the strategy τ'. If the maximal color on the cycle is even, then ρ' is winning for Player 0 in $\mathcal{G}'$, which contradicts τ being a winning strategy for Player 1. Thus, assume the maximal color on the cycle is odd. Then this cycle only contains ε-edges, which contradicts π being unsettled: If it contains an increment-edge, then traversing the cycle sufficiently often would incur a cost of $b+1$ for the maximal color and therefore an overflow to some value strictly smaller than n. However, such overflow positions are not on cycles, as the overflow counter is non-decreasing and strictly smaller than n on the cycle, as π is settled.

Now, we bound the length of π. The structure of our argument is sketched in Figure 6: We define debt-free and request-adding positions, and then show

1. that there are at most n overflow positions in π,
2. that there are at most n debt-free positions between any two adjacent overflow positions,
3. that there are at most d request-adding positions between any two adjacent debt-free positions,
4. that there are at most $b+1$ increment-edges between any two adjacent request-adding positions, where d is the number of odd colors, and
5. that we can remove cycles from π such that the resulting play prefix has at most n positions between any two adjacent increment-edges.

Aggregating these bounds shows that the play prefix resulting from the cycle removal satisfies the desired upper bound ℓ. Subsequently, we show how to construct the strategy τ'_f from such play prefixes.

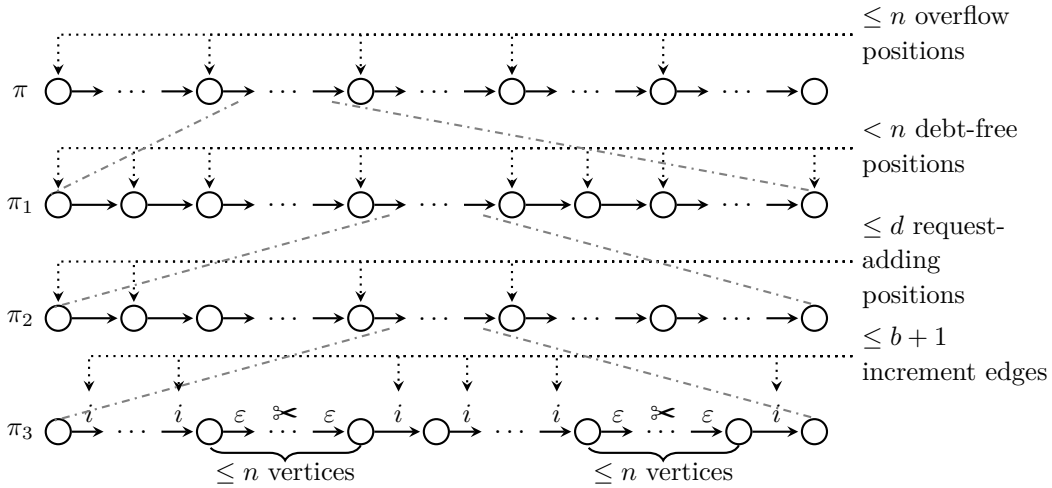

Figure 6 Bounding the length of unsettled play prefixes.

Recall that an overflow position of π is a j with $j = 0$ or with $o_j > o_{j-1}$. As π is unsettled and the o_j are non-decreasing, π has at most n overflow positions, $n-1$ real increments and the initial position. Hence, by splitting π at the overflow positions we obtain at most $n+1$ non-empty infixes of π, each without overflow positions. We say such an infix has type 1.

Fix a type 1 infix π_1. A debt-free position of π is a j with $r_j = r_{v_j}$, i.e., a position that has no other costs than those incurred by visiting v_j. As all vertices of π_1 share the same overflow counter value o_j, there are at most n debt-free positions in π_1: $n+1$ such positions would induce a vertex repetition, which we have ruled out above. Hence, by splitting π_1 at the debt-free positions we obtain at most $n+1$ non-empty infixes of π_1, each without debt-free and overflow positions. We say such an infix has type 2.

Fix a type 2 infix π_2. A request-adding position of π is a j with odd $\Omega(v_j)$ such that $r_{j-1}(c) = \bot$ for all $c \geq \Omega(v_j)$. We define d as the number of odd colors assigned by Ω. We claim that there are at most d request-adding positions in π_2. Assume there are $d+1$. Then, two request-adding positions $j < j'$ share a color, call it c. As j' is request-adding, only requests strictly smaller than c are open at position $j-1$, i.e., c and all larger requests have to be answered in between j and j'. Hence, there is a debt-free position between j and j', which contradicts π_2 being of type 2. Hence, by splitting π_2 at the request-adding positions we obtain at most $d+1$ non-empty infixes of π_2, each without request-adding, debt-free, and overflow positions. We say such an infix has type 3.

Fix a type 3 infix π_3. We show that π_3 contains at most b increment-edges. First, consider the case where there is an open request at the beginning of π_3. As π_3 has no request-adding positions, no larger request occurs in π_3. Thus, as π_3 also has no debt-free positions, the request is not answered during π_3. Thus, $b+1$ increment-edges in π_3 would lead to an overflow position. However, π_3 has no overflow positions by construction. Thus, there are at most b increment-edges in π_3. The other case cannot occur: assume there is no open request at the beginning of π_3: if the color of π_3's first vertex is even, then the position is debt-free, if it is odd, then the position is request-adding. Both types of positions do not appear in π_3. Thus, by splitting π_3 at the increment-edges, we obtain a decomposition of π_3 into at most $b+1$ infixes, each without increment-edges and without request-adding, debt-free, and overflow positions. We say such an infix has type 4.

Fix a type 4 infix π_4, which might be exponentially long in the number of vertices of $\mathcal{G}$,[2] but not longer, as π was shown to be cycle-free. We show how to remove even ε-cycles from type 4 infixes. First, let us note that we argued above that there is a request that is open throughout π_4.

Recall that an ε-cycle of π is not necessarily a cycle in $\mathcal{A}'$, as we only require the projection to V to form a cycle in $\mathcal{A}$. Thus, it is more convenient to work with the projections of the $\pi \in P_{\tau'}$. This is possible, as the memory state that is projected away can be reconstructed from the projected sequence of vertices in V^*.

More formally, let $P \subseteq V^*$ be the projections of elements in $P_{\tau'}$. Due to the determinism of the memory computation, the projection is a bijection between $P_{\tau'}$ and P and the function ext yielding the extended play is its inverse. Now, fix a $\pi \in P$ and the corresponding $\mathrm{ext}(\pi)$ from $P_{\tau'}$. For every type 4 infix of $\mathrm{ext}(\pi)$ we repeatedly (and until no longer applicable) remove even ε-cycles from the corresponding infix of π, as shown in Figure 7. Let $P' \subseteq V^*$ be the set of resulting play prefixes and let $P_{\tau'_f} = \{\mathrm{ext}(\pi) \mid \pi \in P'\} \subseteq (V \times M)^*$ be the set of extensions of play prefixes from P' to play prefixes in $\mathcal{A}'$ obtained by adding the memory states. As we have only removed cycles, $P_{\tau'_f}$ still satisfies the four properties required to induce a strategy, call it τ'_f.

We claim that this strategy is winning for Player 1 in $\mathcal{G}'_f$. Let $\pi_f \in P_{\tau'_f}$ be unsettled and let $\pi \in P_{\tau'}$ be the play prefix from which π_f was obtained by cycle-removal. As we only

[2] It is straightforward to use open requests of k colors to implement a binary counter with k bits, i.e., there is an exponential lower bound on the length of π_4.

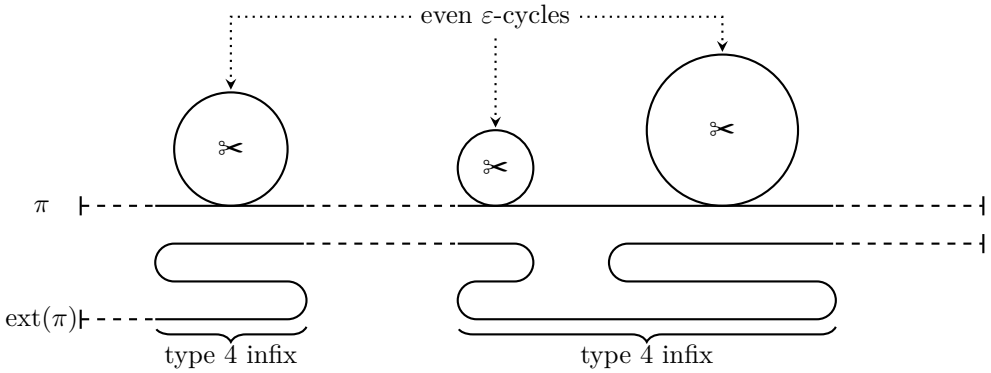

Figure 7 The removal of cycles from π.

remove even ε-cycles, which preserves the evolution of the overflow counter (see Lemma 12) and the existence of odd ε-cycles, π is unsettled as well.

Thus, the following upper bounds shown above are satisfied:

- π has at most n overflow positions and at most n type 1 infixes.
- Every type 1 infix has at most n debt-free positions and at most $n+1$ type 2 infixes.
- Every type 2 infix has at most d request-adding positions and at most $d+1$ type 3 infixes.
- Every type 3 infix has at most b increment-edges and at most $b+1$ type 4 infixes.

Finally, π_f is obtained from π by shortening type 4 prefixes until they have length at most n. Hence, we can bound the length of π_f by $3(n+1)^5 = \ell$, using the facts $d \leq n$ and $b \leq n$.

Thus, every $\pi \in P'_f$ of length $\ell+1$ is settled, which implies that τ'_f is indeed a winning strategy for Player 1.

Now, let us prove the other direction. Let τ'_f be a winning strategy for Player 1 in $\mathcal{G}'_f$. We construct a winning strategy for Player 1 in $\mathcal{G}'$ by simulating a play in $\mathcal{G}'_f$ that is consistent with τ'_f. As this strategy is only useful for the first $\ell+1$ moves, we have to keep the simulating play short. We define the simulation $h\colon (V \times M)^+ \to (V \times M)^+$ and the new strategy τ' simultaneously. The function h satisfies the following invariant:

> Let π be consistent with τ' and end in (v,r,o) with $o < n$. Then, $h(\pi)$ is consistent with τ'_f, is unsettled, and ends in (v,r',o) such that r and r' are on par.

Note that once we have reached a vertex in $\mathcal{G}'$ whose overflow counter has value n we can stop the simulation, since the play has reached the winning sink states for Player 1.

To begin, let $h(v'_I) = v'_I$, which satisfies the invariant. Now, assume we have a play prefix π consistent with τ' ending in (v,r,o) and let $h(\pi) = (v_0,r_0,o_0)\cdots(v_j,r_j,o_j)$. We consider two cases, depending on whose turn it is at the last vertex (v,r,o) of π.

If $(v,r,o) \in V'_1$, i.e., it is Player 1's turn, we distinguish two subcases: if $o = n$, i.e., Player 1 has reached the winning sink states, then we define $\tau'(\pi)$ to be an arbitrary successor of (v,r,o). If $o < n$, then the invariant on h yields that $h(\pi)$ is consistent with τ'_f and unsettled, $v_j = v$, and $o_j = o$. Let $\tau'_f(h(\pi)) = (v^*, \mathrm{Upd}((r_j,o_j),(v_j,v^*)))$. We mimic the move to v^* to continue π by defining $\tau'(\pi) = (v^*, \mathrm{Upd}((r,o),(v,v^*)))$. This is well-defined due to $v_j = v$. We show that the invariant is satisfied after considering the other case.

If $(v,r,o) \in V'_0$, i.e., it is Player 0's turn, she moves to some successor of (v,r,o), say $(v^*, \mathrm{Upd}((r,o),(v,v^*)))$ to keep the notation consistent among both cases.

Let $\mathrm{Upd}((r,o),(v,v^*)) = (r',o')$, i.e., the unique memory state m such that π can be prolonged by a move to (v^*,m), and $\mathrm{Upd}((r_j,o_j),(v,v^*)) = (r'_f,o'_f)$, i.e., the unique memory state m' such that $h(\pi)$ can be prolonged by a move to (v^*,m').

It remains to define $h(\pi \cdot (v^*, r', o'))$ in case o' is smaller than n. Recall that, if $o' = n$, Player 1 has already won and we can define $h(\pi \cdot (v^*, r', o'))$ arbitrarily. Otherwise we distinguish two cases: if $\pi' = h(\pi) \cdot (v^*, r'_f, o'_f)$ is not settled, then we define $h(\pi \cdot (v^*, r', o')) = \pi'$. This satisfies the invariant due to Remark 11.

Now, assume π' is settled. The overflow counter o'_f is equal to o' and thus smaller than n, due to Remark 11. Thus, π' is settled by virtue of having an odd ε-cycle, which has to be a suffix of π'. Thus, the cycle starts in a vertex $(v_{j'}, r_{j'}, o_{j'})$ with $v_{j'} = v^*$, $r_{j'}$ and r'_f are on par, and $o_{j'} = o'_f = o'$. We define $h(\pi \cdot (v^*, r', o')) = (v_0, r_0, o_0) \cdots (v_{j'}, r_{j'}, o_{j'})$, which satisfies the invariant by Lemma 12.

Now, consider a play ρ that is consistent with τ'. If the overflow counter along ρ reaches the value n, then ρ is winning for Player 1. Thus, we consider the case where the counter is always smaller than n.

Towards a contradiction, assume that the maximal color occurring infinitely often in ρ is even, call it c. Let π_i be the prefix of length i of ρ. As the last vertex of π_i and the last vertex of $h(\pi_i)$ share the same vertex from V and the same overflow counter value, which is smaller than n by assumption, they also have the same color. Such vertices occur only finitely often in a removed odd ε-cycle, as every such cycle has an odd color larger than c. Thus, for every removed cycle, there is an occurrence of an odd color larger than c in ρ. By assumption there are only finitely many such occurrences.

Thus, infinitely many vertices of color c are appended to the $h(\pi_i)$ and never removed. This contradicts the $h(\pi_i)$ being unsettled and consistent with τ'_f, as τ'_f settles every play of length $\ell + 1$ and longer. ◀