Parity and Streett Games with Costs*

Nathanaël Fijalkow[1,2] and Martin Zimmermann[2]

[1] LIAFA, Université Paris 7.
nath@liafa.univ-paris-diderot.fr
[2] Institute of Informatics, University of Warsaw.
zimmermann@mimuw.edu.pl

Abstract. We study two-player games played on finite graphs equipped with costs on edges and introduce two winning conditions, cost-parity and cost-Streett, which require bounds on the cost between requests and their responses. Both conditions generalize the corresponding classical omega-regular conditions and the corresponding finitary conditions.
For parity games with costs we show that the first player has positional winning strategies and that determining the winner lies in NP and coNP. For Streett games with costs we show that the first player has finite-state winning strategies and that determining the winner is EXPTIME-complete. This unifies the complexity results for the classical and finitary variants of these games. Both types of games with costs can be solved by solving linearly many instances of their classical variants.

1 Introduction

In recent years, boundedness problems arose in topics pertaining to automata and logics leading to the development of novel models and techniques to tackle these problems. Although in general undecidable, many boundedness problems for automata turn out to be decidable if the acceptance condition can refer to boundedness properties of variables, but the transitions cannot access variable values. A great achievement was made by Hashiguchi [19] who proved decidability of the star-height problem by reducing it to a boundedness problem for a certain type of finite automaton and by then solving this problem. This led the path to recent developments towards a general theory of bounds in automata and logics, comprising automata and logics with bounds [2, 4], satisfiability algorithms for these logics [2, 5, 27], and regular cost-functions [12].

In this work, we consider boundedness problems in turn-based two-player graph games of infinite duration. We introduce cost-parity and cost-Streett conditions which generalize the (classical) ω-regular parity- respectively Streett condition, as well as the finitary parity- respectively finitary Streett condition [9]. While both finitary variants strengthen the classical conditions by adding bounds,

* A preliminary version of this work appeared in FSTTCS 2012 under the name "Cost-parity and Cost-Streett Games" [18]. The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreements 259454 (GALE) and 239850 (SOSNA).

the complexity of solving these games diverges: (in the state of the art) finitary parity games are simpler than parity games, while finitary Streett games are harder than Streett games. Indeed, solving finitary parity games can be carried out in polynomial time [9], while no polynomial-time algorithm for parity games is yet known, and the decision problem for parity games is in **NP** $\cap$ **coNP**. The situation is reversed for Streett games, since solving them is **coNP**-complete [16] while solving finitary Streett games is **EXPTIME**-complete. The latter result is shown in unpublished work by Chatterjee, Henzinger, and Horn: by slightly modifying the proof of **EXPTIME**-hardness of solving request-response games presented in [10] they prove **EXPTIME**-hardness of solving finitary Streett games.

A game with cost-parity condition is played on an arena whose vertices are colored by natural numbers, and where traversing an edge incurs a non-negative cost. Player 0 wins a play if there is a bound b such that all but finitely many odd colors seen along the play (which we think of as requests) are followed by a larger even color (which we think of as responses) that is reached with cost at most b. The definition of the cost-Streett condition goes along the same lines, but the requests and responses are independent and not hierarchically ordered as in parity conditions. The cost of traversing an edge can be used to model the consumption of a resource. Thus, if Player 0 wins a play she can achieve her goal along an infinite run with bounded resources. On the other hand, Player 1's objective is to exhaust the resource, no matter how big the capacity is. Note that this is not an ω-regular property, which is witnessed by the fact that Player 1 needs infinite memory to win such games.

Since the term "cost-parity games" has been used before [12, 13, 27], we refer to games with cost-parity conditions as parity games with costs. There are two main differences between cost-parity games and parity games with costs as introduced here. First, the bound quantification in cost-parity games requires the counter values to be uniformly bounded over all paths, whereas in parity games with costs the bound can depend on the path. Second, in cost-parity games the counters and the parity conditions are independent, whereas in parity games with costs the counters are used to give a quantitative measure of the satisfaction of the parity condition.

We show that parity games with costs enjoy two nice properties of parity and finitary parity games: Player 0 has memoryless winning strategies and determining the winner lies in **NP** $\cap$ **coNP**. Furthermore, we show that solving parity games with costs can be algorithmically reduced to solving parity games, which allows to solve these games almost as efficiently as parity games. We then consider Streett games with costs and prove that Player 0 has finite-state winning strategies, and that determining the winner is **EXPTIME**-complete. Our complexity results unify the previous results about finitary parity and Streett games and the results about their classical variants.

To obtain our results, we present an algorithm to solve parity games with costs that iteratively computes the winning region of Player 0 employing an algorithm to solve parity games. This "reduction" to parity games also yields finite-state winning strategies for Player 0 in parity games with costs. However, this can be improved: by exploiting the intrinsic structure of the memory introduced in the reduction, we are able to prove the existence of positional winning strategies for Player 0. We also give a second proof of this result: we show how to transform an arbitrary finite-state winning strategy into a positional one. This construction relies on so-called scoring functions (which are reminiscent of the simulation of alternating tree-automata by non-deterministic automata presented in [25] and of scoring functions for Muller games [23]) and presents a general framework to turn finite-state strategies into positional ones, which we believe to be applicable in other situations as well. Finally, we present an algorithm that solves Streett games with costs by solving Streett games. Here, we show the existence of finite-state winning strategies for Player 0 in Streett games with costs.

In our proofs, we reduce games with boundedness winning conditions to games with ω-regular winning conditions. The solution of the domination problem for regular cost-functions on finite trees [13] uses a similar approach. But in contrast to this work, which is concerned with proving decidability of a stronger formalism, we are interested in efficient algorithms. Hence, we need a more sophisticated reduction and a careful analysis of the memory requirements.

Adding quantitative requirements to qualitative winning conditions has been an active field of research during the last decade: much attention is being paid to not just synthesize some winning strategy, but to find an optimal one according to a certain quality measure, e.g., the use of mean-payoff objectives and weighted automata to model quantitative aspects in the winning condition [1, 7, 11]. For request-response games and their extensions waiting times between requests and their responses are used to measure the quality of a strategy and it was shown how to compute optimal (w.r.t. the limit superior of the mean waiting time) winning strategies [20, 29]. However, the optimal finite-state strategies that are obtained are exponentially larger than the ones computed by the classical algorithm.

Finally, there has been a lot of interest in so-called energy games, whose winning conditions are boundedness requirements on the consumption of resources. Solving energy games with multiple resources is in general intractable [17] while so-called consumption games, a subclass of energy games, are shown to be tractable in [6]. Furthermore, energy parity games, whose winning conditions are a conjunction of a (single resource) energy and a parity condition, can be solved in $\mathbf{NP} \cap \mathbf{coNP}$ and one player (the spoiling one) has positional winning strategies while the other one needs exponential memory [8]. Although the first two results are similar to our results on parity games with costs, the energy parity condition does not relate the energy consumption to the parity condition. In contrast, the costs in parity games with costs give a qualitative measure of the satisfaction of the parity condition.

The paper is organized as follows. In Section 2, we define the necessary material related to games and introduce cost-parity and cost-Streett conditions, as well as their bounded variants. In Section 3, we study bounded parity games with costs, providing an algorithm to solve them and tight memory requirements for winning strategies. In Section 4, we show how to reduce the problem of solving parity games with costs to the problem of solving bounded parity games with costs. In Section 5, we give a different proof of the existence of positional strategies for (bounded) parity games with costs, via scoring-functions. In Section 6, we study Streett games with costs.

2 Definitions

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

An arena $\mathcal{A} = (V, V_0, V_1, E)$ consists of a finite, directed graph (V, E) and a partition $\{V_0, V_1\}$ of V into the positions of Player 0 (drawn as circles) and the positions of Player 1 (drawn as rectangles). A play in $\mathcal{A}$ starting in $v \in V$ is an infinite path $\rho = \rho_0\rho_1\rho_2\cdots$ through (V, E) such that $\rho_0 = v$. To avoid the nuisance of dealing with finite plays, we assume every vertex to have an outgoing edge.

A game $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$ consists of an arena $\mathcal{A}$ and a set $\mathrm{Win} \subseteq V^\omega$ of winning plays for Player 0. The set of winning plays for Player 1 is $V^\omega \setminus \mathrm{Win}$. We say that Win is prefix-independent, if $\rho \in \mathrm{Win}$ if and only if $w\rho \in \mathrm{Win}$ for every finite play prefix w and every infinite play ρ.

A strategy for Player i is a mapping $\sigma \colon V^*V_i \to V$ such that $(v, \sigma(wv)) \in E$ 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 mapping $\sigma \colon V_i \to V$. A play $\rho_0\rho_1\rho_2\dots$ is consistent with σ if $\rho_{n+1} = \sigma(\rho_0\cdots\rho_n)$ for every n with $\rho_n \in V_i$. A strategy σ for Player i is a winning strategy from a set of vertices $W \subseteq V$ if every play that starts in some $v \in W$ and is consistent with σ is won by Player i. The winning region $W_i(\mathcal{G})$ of Player i in $\mathcal{G}$ is the set of vertices from which Player i has a winning strategy. We say that a strategy is uniform, if it is winning from all $v \in W_i(\mathcal{G})$. We always have $W_0(\mathcal{G}) \cap W_1(\mathcal{G}) = \emptyset$. On the other hand, if $W_0(\mathcal{G}) \cup W_1(\mathcal{G}) = V$, then we say that $\mathcal{G}$ is determined. All games we consider in this work are determined. Solving a game amounts to determining its winning regions and winning strategies.

A memory structure $\mathcal{M} = (M, \mathrm{Init}, \mathrm{Upd})$ for an arena (V, V_0, V_1, E) consists of a finite set M of memory states, an initialization function $\mathrm{Init} \colon V \to M$, and an update function $\mathrm{Upd} \colon M \times V \to M$. The update function can be extended to $\mathrm{Upd}^+ \colon V^+ \to M$ in the usual way: $\mathrm{Upd}^+(\rho_0) = \mathrm{Init}(\rho_0)$ and $\mathrm{Upd}^+(\rho_0 \dots \rho_n\rho_{n+1}) = \mathrm{Upd}(\mathrm{Upd}^+(\rho_0 \dots \rho_n), \rho_{n+1})$. 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 $\sigma(\rho_0 \dots \rho_n) = \mathrm{Nxt}(\rho_n, \mathrm{Upd}^+(\rho_0 \dots \rho_n))$. A strategy is called finite-state if it can be implemented by a memory structure.

An arena $\mathcal{A} = (V, V_0, V_1, E)$ and a memory structure $\mathcal{M} = (M, \mathrm{Init}, \mathrm{Upd})$ for $\mathcal{A}$ induce the expanded arena $\mathcal{A} \times \mathcal{M} = (V \times M, V_0 \times M, V_1 \times M, E')$ where $((v, m), (v', m')) \in E'$ if and only if $(v, v') \in E$ and $\mathrm{Upd}(m, v') = m'$. Every play ρ in $\mathcal{A}$ has a unique extended play $\rho' = (\rho_0, m_0)(\rho_1, m_1)(\rho_2, m_2)\dots$ in $\mathcal{A} \times \mathcal{M}$ defined by $m_0 = \mathrm{Init}(\rho_0)$ and $m_{n+1} = \mathrm{Upd}(m_n, \rho_{n+1})$, i.e., $m_n = \mathrm{Upd}^+(\rho_0 \cdots \rho_n)$.

A game $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$ is reducible to $\mathcal{G}' = (\mathcal{A}', \mathrm{Win}')$ via $\mathcal{M}$, written $\mathcal{G} \leq_{\mathcal{M}} \mathcal{G}'$, if $\mathcal{A}' = \mathcal{A} \times \mathcal{M}$ and every play ρ in $\mathcal{G}$ is won by the player who wins the extended play ρ' in $\mathcal{G}'$, i.e., $\rho \in \mathrm{Win}$ if and only if $\rho' \in \mathrm{Win}'$.

Lemma 1 *Let $\mathcal{G}$ be a game with vertex set V and $W \subseteq V$. If $\mathcal{G} \leq_{\mathcal{M}} \mathcal{G}'$ and Player i has a positional winning strategy for $\mathcal{G}'$ from $\{(v, \mathrm{Init}(v)) \mid v \in W\}$, then she has a finite-state winning strategy for $\mathcal{G}$ from W which is implemented by $\mathcal{M}$.*

Especially, if Player i has a uniform positional winning strategy for $\mathcal{G}'$, then she has a uniform finite-state winning strategy for $\mathcal{G}$ that is implemented by $\mathcal{M}$.

Let $\mathcal{A} = (V, V_0, V_1, E)$ and $i \in \{0, 1\}$. The i-attractor of $F \subseteq V$ in $\mathcal{A}$, denoted by $\mathrm{Attr}_i^{\mathcal{A}}(F)$, is defined by $\mathrm{Attr}_i^{\mathcal{A}}(F) = \bigcup_{j=0}^{|V|} A_j$, where $A_0 = F$ and

$$\begin{aligned} A_{j+1} = A_j &\cup \{v \in V_i \mid \exists v' \in A_j \text{ such that } (v, v') \in E\} \\ &\cup \{v \in V_{1-i} \mid \forall v', (v, v') \in E \text{ implies } v' \in A_j\} \enspace . \end{aligned}$$

Player i has a (positional) strategy such that every play that starts in $\mathrm{Attr}_i^{\mathcal{A}}(F)$ and is consistent with the strategy visits F. Such strategies are called attractor strategies.

A trap for Player i is a set X of vertices such that the successors of every vertex in $X \cap V_i$ are again in X and every vertex in $X \cap V_{1-i}$ has a successor in X. Player $1 - i$ has a (positional) strategy such that every play that starts in a trap X and is consistent with the strategy stays in X forever. The complement of an attractor $\mathrm{Attr}_i^{\mathcal{A}}(F)$ is a trap for Player i. Furthermore, removing an attractor from an arena never introduces terminal vertices.

The following observation will be useful later: if the sets of winning plays Win in $\mathcal{G}$ is prefix-independent, then we have $W_i(\mathcal{G}) = \mathrm{Attr}_i^{\mathcal{A}}(W_i(\mathcal{G}))$ and $W_i(\mathcal{G})$ is a trap for Player $1 - i$. Furthermore, no play consistent with a winning strategy for Player i will ever leave $W_i(\mathcal{G})$.

2.1 Winning Conditions

In this subsection, we present the winning conditions we consider in this paper. Fix an arena $\mathcal{A}$ with set of edges E. A cost-function for $\mathcal{A}$ is an edge-labelling $\mathrm{Cst}\colon E \to \{\varepsilon, i\}$. An edge with label i is called an increment-edge, and edges labelled by ε are called accordingly ε-edges. We extend the edge-labelling to a cost-function over finite and infinite paths obtained by counting the number of increment-edges traversed along the path. Note that we only allow zero and one as cost. Another option would be to allow arbitrary costs in $\mathbb{N}$ but since all our results depend only on whether an edge has cost zero or not, we refrain from doing this.

Cost-Parity Conditions Let $\mathcal{A} = (V, V_0, V_1, E)$ be an arena and let $\Omega\colon V \to \mathbb{N}$ be a coloring of its vertices by natural numbers. In all games we are about to define in this subsection, we interpret the occurrence of a color as request, which has to be answered by visiting a vertex of larger even color at a later position. By imposing conditions on the responses we obtain several different types of winning conditions. To simplify our notations, let $\mathrm{Ans}(c) = \{c' \in \mathbb{N} \mid c' \geq c \text{ and } c' \text{ even}\}$ be the set of colors that answer a request of color c. Note that $\mathrm{Ans}(c) \subseteq \mathrm{Ans}(c')$ for $c \geq c'$ and $c \in \mathrm{Ans}(c)$ if c is even.

Fix a cost-function Cst and consider a play $\rho = \rho_0\rho_1\rho_2\cdots$ and a position $k \in \mathbb{N}$. We define the cost-of-response at position k of ρ by

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

where we use $\min \emptyset = \infty$, i.e., $\mathrm{Cor}_{\mathrm{Cst}}(\rho, k)$ 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 c, if $\mathrm{Cor}_{\mathrm{Cst}}(\rho, k) = c$. Note that a request at a position k with an even color is answered with cost zero. Finally, we say that a request at position k is unanswered with cost ∞, if there is no position $k' \geq k$ such that $\Omega(\rho_{k'}) \in \mathrm{Ans}(\Omega(\rho_k))$ and we have $\mathrm{Cst}(\rho_k\rho_{k+1}\cdots) = \infty$, i.e., there are infinitely many increment-edges after position k, but no answer.

We begin defining winning conditions by introducing the parity condition, denoted by $\mathrm{Parity}(\Omega)$, which requires that all but finitely many requests are answered. Equivalently, $\rho \in \mathrm{Parity}(\Omega)$ if and only if the maximal color that occurs infinitely often in ρ is even. Both players have uniform positional winning strategies in parity games [15, 24] and their winning regions can be decided in $\mathbf{NP} \cap \mathbf{coNP}$.

By bounding the costs between requests and their responses, we strengthen the parity condition and obtain the cost-parity and the bounded cost-parity condition. The former is defined as

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

i.e., ρ satisfies the cost-parity condition, if there exists a bound $b \in \mathbb{N}$ such that all but finitely many requests are answered with cost less than b.

The bounded cost-parity condition, denoted by $\mathrm{BndCostParity}(\Omega, \mathrm{Cst})$, is again obtained by a strengthening: $\rho \in \mathrm{BndCostParity}(\Omega, \mathrm{Cst})$ if there exists a bound $b \in \mathbb{N}$ such that all but finitely many requests are answered with cost less than b, and there is no unanswered request of cost ∞. Note that this is *not* equivalent to requiring that there exists a bound $b' \in \mathbb{N}$ such that all requests are answered with cost less than b' (e.g., if there are unanswered requests in a play with finitely many increment-edges).

Remark 1 *We have* $\mathrm{BndCostParity}(\Omega, \mathrm{Cst}) \subseteq \mathrm{CostParity}(\Omega, \mathrm{Cst}) \subseteq \mathrm{Parity}(\Omega)$ *and* $V^* \cdot \mathrm{BndCostParity}(\Omega, \mathrm{Cst}) = \mathrm{CostParity}(\Omega, \mathrm{Cst})$. *Furthermore, note that* $\mathrm{CostParity}(\Omega, \mathrm{Cst})$ *and* $\mathrm{Parity}(\Omega)$ *are prefix-independent. This is not the case for* $\mathrm{BndCostParity}(\Omega, \mathrm{Cst})$.

A game $\mathcal{G} = (\mathcal{A}, \mathrm{CostParity}(\Omega, \mathrm{Cst}))$ is called a parity game with costs, and a game with winning condition

$$\mathrm{BndCostParity}(\Omega, \mathrm{Cst})$$

is a bounded parity game with costs. Note that both cost-conditions defined here generalize the classical parity conditions as well as the finitary respectively bounded parity conditions [9]. Indeed, if $\mathcal{A}$ contains no increment-edges, then the three conditions are equivalent, i.e.,

$$\mathrm{CostParity}(\Omega, \mathrm{Cst}) = \mathrm{BndCostParity}(\Omega, \mathrm{Cst}) = \mathrm{Parity}(\Omega) \ .$$

On the other hand, if $\mathcal{A}$ contains no ε-edges, then $\mathrm{CostParity}(\Omega, \mathrm{Cst})$ is equal to the finitary parity condition over Ω and $\mathrm{BndCostParity}(\Omega, \mathrm{Cst})$ is equal to the bounded parity condition over Ω. Hence, parity games with costs generalize both parity and finitary parity games. Similarly, bounded parity games with costs generalize both parity and bounded parity games.

Since (bounded) cost-parity conditions can be shown to be on the third level of the Borel hierarchy, we obtain the following result as a consequence of the Borel determinacy theorem [22].

Remark 2 *(Bounded) parity games with costs are determined.*

Example 1 *Consider the parity game with costs depicted in Figure 1 where all vertices belong to V_1, and the label of a vertex denotes its name (in the upper part) and its color (in the lower part). Player 1 wins from $\{a, b, c\}$ by requesting color 1 at vertex a infinitely often and staying at vertex b longer and longer, but also visiting c infinitely often (and thereby answering the request). Note that this strategy is not finite-state. Indeed, one can easily prove that Player 1 does not have a finite-state winning strategy for this game. Player 0 wins from every other vertex, since Player 1 can raise only finitely many requests from these vertices, albeit these requests are unanswered with cost ∞.*

If we consider the game as a bounded parity game with costs, then Player 1 wins from every vertex but g by moving to g and then staying there ad infinitum. Every such play contains a request of color 1 that is unanswered with cost ∞. From g, Player 0 wins, since there is only one play starting from g, in which no request is ever raised.

Cost-Streett Conditions Fix an arena $\mathcal{A} = (V, V_0, V_1, E)$. Let $\Gamma = (Q_c, P_c)_{c \in [d]}$ be a collection of d (Streett) pairs of subsets of V, i.e., $Q_c, P_c \subseteq V$, and let $\overline{\mathrm{Cst}} = (\mathrm{Cst}_c)_{c \in [d]}$ be a collection of d cost-functions for $\mathcal{A}$. We think of visits to vertices in Q_c as requests, visits to P_c as responses, and measure the cost of these responses using Cst_c. Formally, for $c \in [d]$, a play $\rho = \rho_0 \rho_1 \rho_2 \cdots$, and a position k we define

$$\mathrm{StCor}_{\mathrm{Cst}_c}(\rho, k) = \begin{cases} 0 & \text{if } \rho_k \notin Q_c, \\ \min\{\mathrm{Cst}_c(\rho_k \cdots \rho_{k'}) \mid \rho_{k'} \geq \rho_k \text{ and } \rho_{k'} \in P_c\} & \text{if } \rho_k \in Q_c, \end{cases}$$

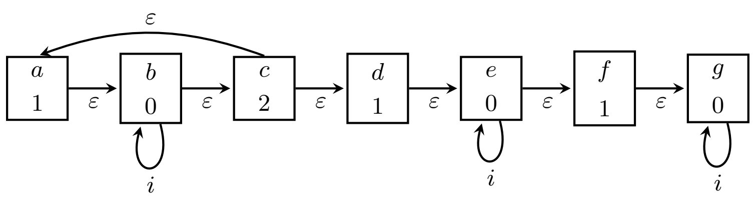

Fig. 1. A (bounded) parity game with costs.

where we use $\min\emptyset = \infty$. We define $\mathrm{StCor}_{\overline{\mathrm{Cst}}}(\rho, k) = \max\{\mathrm{StCor}_{\mathrm{Cst}_c}(\rho, k) \mid c \in [d]\}$ and say that the requests at position k are answered with cost c, if $\mathrm{StCor}_{\overline{\mathrm{Cst}}}(\rho, k) = c$, and that the requests are unanswered with cost ∞, if $\mathrm{StCor}_{\overline{\mathrm{Cst}}}(\rho, k) = \infty$ and there are infinitely many increment-edges after position k (w.r.t. some Cst_c such that $\rho_k \in Q_c$).

We consider the following winning conditions. The (classical) Streett condition $\mathrm{Streett}(\mathit{\Gamma})$ requires for every c that P_c is visited infinitely often if Q_c is visited infinitely often, i.e., all but finitely many requests are answered.

Again, by requiring a bound on the costs between requests and responses, we strengthen the Streett condition: the cost-Streett condition

$$\mathrm{CostStreett}(\mathit{\Gamma}, \overline{\mathrm{Cst}}) = \{\rho \in V^\omega \mid \limsup_{k\to\infty} \mathrm{StCor}_{\overline{\mathrm{Cst}}}(\rho, k) < \infty\}$$

requires the existence of a bound b such that all but finitely many requests are answered with cost less than b.

Finally, the bounded cost-Streett condition $\mathrm{BndCostStreett}(\mathit{\Gamma}, \overline{\mathrm{Cst}})$ requires the existence of a bound b such that all but finitely many requests are answered with cost less than b, and that there is no unanswered request of cost ∞.

Remark 3 *We have*

$$\mathrm{BndCostStreett}(\mathit{\Omega}, \overline{\mathrm{Cst}}) \subseteq \mathrm{CostStreett}(\mathit{\Omega}, \overline{\mathrm{Cst}}) \subseteq \mathrm{Streett}(\mathit{\Omega})$$

and $V^* \cdot \mathrm{BndCostStreett}(\mathit{\Omega}, \overline{\mathrm{Cst}}) = \mathrm{CostStreett}(\mathit{\Omega}, \overline{\mathrm{Cst}})$. *Furthermore, note that* $\mathrm{CostStreett}(\mathit{\Omega}, \mathrm{Cst})$ *and* $\mathrm{Streett}(\mathit{\Omega})$ *are prefix-independent. This is not the case for* $\mathrm{BndCostStreett}(\mathit{\Omega}, \mathrm{Cst})$.

A game $(\mathcal{A}, \mathrm{CostStreett}(\mathit{\Gamma}, \overline{\mathrm{Cst}}))$ where $\mathit{\Gamma}$ and $\overline{\mathrm{Cst}}$ have the same size is called a Streett game with costs. As for (bounded) cost-parity conditions, the winning conditions defined here generalize the classical Streett condition as well as the finitary respectively bounded Streett condition of [9]. Indeed, if $\mathcal{A}$ contains no increment-edges, then the three conditions are equivalent, i.e., $\mathrm{CostStreett}(\mathit{\Gamma}, \overline{\mathrm{Cst}}) = \mathrm{BndCostStreett}(\mathit{\Gamma}, \overline{\mathrm{Cst}}) = \mathrm{Streett}(\mathit{\Gamma})$. Similarly, if $\mathcal{A}$ contains no ε-edges, then $\mathrm{CostStreett}(\mathit{\Gamma}, \overline{\mathrm{Cst}})$ is equal to the finitary Streett condition over $\mathit{\Gamma}$ and $\mathrm{BndCostParity}(\mathit{\Gamma}, \overline{\mathrm{Cst}})$ is equal to the bounded Streett condition over $\mathit{\Gamma}$. Hence, Streett games with costs generalize both Streett and finitary Streett games. Similarly, bounded Streett games with costs generalize both

Streett and bounded Streett games. Furthermore, just as classical Streett games subsume parity games, Streett games with costs subsume parity games with costs, and bounded Streett games with costs subsume bounded parity games with costs.

Figure 2 shows the expressiveness of the winning conditions, e.g., the arrow from "bounded parity" to "bounded Streett" denotes that every bounded parity condition is also a bounded Streett condition.

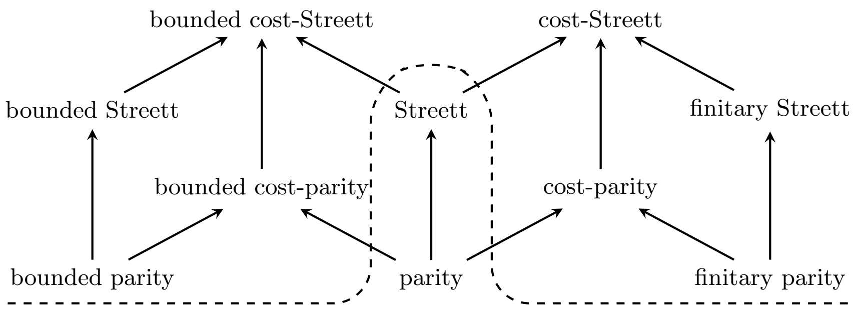

Fig. 2. Expressiveness of winning conditions; those below the dashed line are ω-regular.

Finally, we obtain determinacy via the Borel determinacy theorem [22].

Remark 4 *(Bounded) Streett games with costs are determined.*

3 Bounded Parity Games with Costs

In this section, we study bounded parity games with costs. We first show how to solve such games, and then consider the memory requirements for winning strategies for both players.

3.1 Solving Bounded Parity Games with Costs via ω-regular Games

To solve bounded parity games with costs, we present a relaxation of the bounded cost-parity condition, called PCRR, which essentially replaces the bound b on the cost between a request and its response by just requiring an answer to every request. For plays with finite cost we just require the parity condition to be satisfied, just as the bounded cost-parity condition does. The PCRR-condition is ω-regular, thus both players have finite-state winning strategies. Using the fact that a finite-state winning strategy for Player 0 answers every request within a fixed number of steps (and thereby also with bounded cost), we are able to show that these two games have the same winning regions. Finally, we show how to reduce the PCRR-condition to a parity condition. This completes our algorithm for solving bounded parity games with costs and also yields upper bounds on the memory requirements of both players in bounded parity games with costs.

Let $\mathcal{G} = (\mathcal{A}, \mathrm{BndCostParity}(\Omega, \mathrm{Cst}))$. In the following, we assume that no vertex of $\mathcal{A}$ has both incoming increment- and ε-edges. This can be achieved by subdividing every increment-edge $e = (v, v')$: we add a new vertex $\mathrm{sub}(e)$ and replace e by $(v, \mathrm{sub}(e))$ (which is an increment-edge) and by $(\mathrm{sub}(e), v')$ (which is an ε-edge). Now, only the newly added vertices have incoming increment-edges, but they do not have incoming ε-edges. Furthermore, it is easy to see that Player i wins from a vertex in the original game if and only if she wins from this vertex in the modified game (where we color $\mathrm{sub}(e)$ by $\Omega(v')$). Finally, the modification does not increase the memory requirements, e.g., if Player 0 has a positional winning strategy for the modified game, then also for the original game.

We say that a vertex is an increment-vertex, if it has an incoming increment-edge (which implies that all incoming edges are increment-edges). Let I be the set of increment-vertices. Then, $\mathrm{coBüchi}(I) = \{\rho \mid \mathrm{Cst}(\rho) < \infty\}$ is the set of infinite plays having finite cost. Furthermore, by $\mathrm{RR}(\Omega)$ we denote the set of infinite plays in which every request is answered. We define

$$\mathrm{PCRR}(\Omega, I) = (\mathrm{Parity}(\Omega) \cap \mathrm{coBüchi}(I)) \cup \mathrm{RR}(\Omega) \ ,$$

which is ω-regular. Note that $\mathrm{PCRR}(\Omega, I)$ relaxes $\mathrm{BndCostParity}(\Omega, \mathrm{Cst})$ by giving up the bound on the cost between requests and responses, in other words $\mathrm{PCRR}(\Omega, I) \supseteq \mathrm{BndCostParity}(\Omega, \mathrm{Cst})$.

Lemma 2 *Let $\mathcal{G} = (\mathcal{A}, \mathrm{BndCostParity}(\Omega, \mathrm{Cst}))$ and $\mathcal{G}' = (\mathcal{A}, \mathrm{PCRR}(\Omega, I))$, where I is defined as above. A finite-state winning strategy for Player i in $\mathcal{G}'$ from a set W of vertices is also a winning strategy for Player i in $\mathcal{G}$ from W.*

Proof. The statement for $i = 1$ follows from the inclusion $V^\omega \setminus \mathrm{PCRR}(\Omega, I) \subseteq V^\omega \setminus \mathrm{BndCostParity}(\Omega, \mathrm{Cst})$.

Now, consider the case $i = 0$ and let σ be a finite-state winning strategy for Player 0 in $\mathcal{G}'$ from W. We argue that σ is also a winning strategy for Player 0 for $\mathcal{G}$ from W: let ρ be consistent with σ and starting in W, which implies $\rho \in \mathrm{PCRR}(\Omega, I)$.

If ρ satisfies $\mathrm{Parity}(\Omega)$ and has only finitely many increments (say b many), then all but finitely many requests are answered with cost less than $b + 1$ and there is no unanswered request of cost ∞, i.e., $\rho \in \mathrm{BndCostParity}(\Omega, \mathrm{Cst})$.

Otherwise, ρ satisfies $\mathrm{RR}(\Omega)$, i.e. every request in ρ is answered. We show that every request in ρ is answered with cost at most $b = |V| \cdot |\sigma|$ (where $|\sigma|$ is the size of the memory structure implementing σ), which implies that $\rho \in \mathrm{BndCostParity}(\Omega, \mathrm{Cst})$. Towards a contradiction, assume that there is a request that is answered with cost greater than b. Then, there are two positions between the request and its answer having the same vertex, an increment-edge in between them, and such that the memory structure implementing σ assumes the same state at both positions. Hence, using this loop forever is also a play that is consistent with σ. However, this play contains an unanswered request of cost ∞ and therefore does not satisfy $\mathrm{PCRR}(\Omega, I)$. This yields the desired contradiction to the fact that σ is a winning strategy.

We give an important corollary which shows that solving $\mathcal{G}'$ also solves $\mathcal{G}$.

Corollary 1 *Let $\mathcal{G}$ and $\mathcal{G}'$ as in Lemma 2. Then, $W_i(\mathcal{G}) = W_i(\mathcal{G}')$ for $i \in \{0, 1\}$.*

We now show how to reduce $\mathcal{G}' = (\mathcal{A}, \mathrm{PCRR}(\Omega, I))$ to a *small* parity game. Let O be the set of odd colors in $\Omega(V)$. We define a memory structure $\mathcal{M} = (M, \mathrm{Init}, \mathrm{Upd})$ with $M = O \cup \{\perp\}$,

$$\mathrm{Init}(v) = \begin{cases} \Omega(v) & \text{if } \Omega(v) \text{ odd,} \\ \perp & \text{otherwise,} \end{cases}$$

$\mathrm{Upd}(\perp, v) = \mathrm{Init}(v)$, and

$$\mathrm{Upd}(c, v) = \begin{cases} \max(\Omega(v), c) & \text{if } \Omega(v) \text{ odd,} \\ \perp & \text{if } \Omega(v) \in \mathrm{Ans}(c), \\ c & \text{otherwise.} \end{cases}$$

Intuitively, $\mathrm{Upd}^+(w)$ is the largest unanswered request in w, and is $\perp$ if every request in w is answered. Furthermore, let ℓ be an odd color that is larger than every color in $\Omega(V)$. Now, we define a coloring $\Omega_{\mathcal{M}}$ of the arena $\mathcal{A} \times \mathcal{M}$ via

$$\Omega_{\mathcal{M}}(v, m) = \begin{cases} \ell + 1 & \text{if } m = \perp, \\ \ell & \text{if } m \neq \perp \text{ and } v \in I, \\ \Omega(v) & \text{otherwise.} \end{cases}$$

So, having all requests answered (i.e., being in memory state $\perp$) is most desirable for Player 0 while visiting increment-vertices (i.e., vertices in I) while having an open request is most desirable for Player 1. If neither of these occurs infinitely often, then the old coloring Ω determines the winner (without taking the memory states into account).

Lemma 3 *Let $\mathcal{G}' = (\mathcal{A}, \mathrm{PCRR}(\Omega, I))$ and $\mathcal{G}'' = (\mathcal{A} \times \mathcal{M}, \mathrm{Parity}(\Omega_{\mathcal{M}}))$. Then, $\mathcal{G}' \leq_{\mathcal{M}} \mathcal{G}''$.*

Proof. Let $\rho' = v_0 v_1 v_2 \cdots$ be a play in $\mathcal{A}$ and $\rho'' = (v_0, m_0)(v_1, m_1)(v_2, m_2) \cdots$ be its extended play in $\mathcal{A} \times \mathcal{M}$. By construction, m_j is the largest unanswered request in $v_0 \cdots v_j$. We have to show that the same player wins both ρ' in $\mathcal{G}'$ and ρ'' in $\mathcal{G}''$.

Assume $\rho' \in \mathrm{PCRR}(\Omega, I)$. If $\rho' \in \mathrm{RR}(\Omega)$, then every request is answered, i.e., m_j is infinitely often equal to $\perp$. These vertices have the largest color in $\mathcal{G}''$, which is even. Hence, $\rho'' \in \mathrm{Parity}(\Omega_{\mathcal{M}})$. On the other hand, if $\rho' \in \mathrm{Parity}(\Omega) \cap \mathrm{coBüchi}(I)$ but $\rho' \notin \mathrm{RR}(\Omega)$, then ρ' and ρ'' each have a suffix (starting after the last occurrence of an increment-vertex or the last unanswered request, whichever comes last) such that these suffixes have the same sequence of colors. Hence, ρ'' satisfies $\mathrm{Parity}(\Omega_{\mathcal{M}})$.

Conversely, assume $\rho'' \in \mathrm{Parity}(\Omega_{\mathcal{M}})$. If $\ell + 1$ is the maximal color seen infinitely often, then m_j is infinitely often equal to $\perp$, which implies that every request in ρ' is answered, i.e., $\rho' \in \mathrm{RR}(\Omega) \subseteq \mathrm{PCRR}(\Omega, I)$. On the other hand, if the maximal color seen infinitely often is smaller than $\ell + 1$ (but still even, since we assume Player 0 wins ρ''), then there are only finitely many increment-vertices in ρ' and the plays ρ' and ρ'' each have a suffix such that these suffixes have the same sequence of colors. Hence, ρ' satisfies $\mathrm{Parity}(\Omega)$. Altogether, we have $\rho' \in \mathrm{Parity}(\Omega) \cap \mathrm{coBüchi}(I) \subseteq \mathrm{PCRR}(\Omega, I)$.

Corollary 2 *In bounded parity games with costs, both players have uniform finite-state winning strategies of size $d + 1$, where d is the number of odd colors in the game.*

Proof. The reduction from games with winning condition $\mathrm{PCRR}(\Omega, I)$ to parity games yields uniform finite-state winning strategies of size $d + 1$ for such games. Now apply Lemma 2.

In the next subsection, we show this bound to be tight for Player 1 and show that Player 0 even has positional winning strategies.

The reduction from PCRR games to parity games and Lemma 2 show that solving a parity game suffices to solve a bounded parity games with costs and proves the following theorem. Here, n is the number of vertices, m is the number of edges, and d is the number of colors in the game.

Theorem 1 *Given an algorithm that solves parity games in time $T(n, m, d)$, there is an algorithm that solves bounded parity games with costs with a time complexity $O(T(dn, dm, d + 2))$.*

Furthermore, since solving parity games is in $\mathbf{NP} \cap \mathbf{coNP}$ and the blowup in our reduction is polynomial, we obtain the following remark.

Remark 5 *The following problem is in $\mathbf{NP} \cap \mathbf{coNP}$: given a bounded parity game with costs $\mathcal{G}$, $i \in \{0, 1\}$, and a vertex v, is $v \in W_i(\mathcal{G})$?*

Let us conclude by considering the special case of a bounded parity game with costs $\mathcal{G}$ in which every edge is an increment-edge, i.e., where $\mathcal{G}$ is a bounded parity game. These games can be solved in polynomial time [9]. In this case, $\mathrm{PCRR}(\Omega, \mathrm{Cst})$ is equal to $\mathrm{RR}(\Omega)$, which is a request-response condition [28] where the sets of requests and responses form a hierarchy, induced by the order on the colors. It is easy to derive from the reduction to Büchi games [28] that such games can be solved in polynomial time. Hence, we have recovered the result of [9] on bounded parity games as a special case of our algorithm, although the running time of this algorithm is worse than the running time of the algorithm presented in [9].

3.2 Memory Requirements in Bounded Parity Games with Costs

In this subsection, we determine the exact memory requirements for both players in bounded parity games with costs. We begin by considering Player 0 and improve on Corollary 2.

Lemma 4 *In bounded parity games with costs, Player* 0 *has uniform positional winning strategies.*

Proof. Due to Lemma 2, it suffices to prove the statement for games $\mathcal{G}' = (\mathcal{A}, \mathrm{PCRR}(\Omega, I))$. Recall that we reduced such a game to a parity game $\mathcal{G}'' = (\mathcal{A} \times \mathcal{M}, \mathrm{Parity}(\Omega_{\mathcal{M}}))$ using a memory structure $\mathcal{M}$ that keeps track of the largest open request. Specifically, Lemma 3 reads as follows: $v_0 \in W_0(\mathcal{G}')$ if and only if $(v_0, \mathrm{Init}(v_0)) \in W_0(\mathcal{G}'')$.

We order $M = O \cup \{\bot\}$ with the natural order on integers for O, where $\bot$ is the minimal element. Player 0's winning region in $\mathcal{G}''$ is downwards-closed, i.e., $(v, m) \in W_0(\mathcal{G}'')$ and $m' < m$ implies $(v, m') \in W_0(\mathcal{G}'')$, which can be shown by mimicking a winning strategy from (v, m) to also win from (v, m'). Thus, for $v \in W_0(\mathcal{G}')$, we define

$$\max(v) = \max\{m \in M \mid (v, m) \in W_0(\mathcal{G}'')\} \enspace ,$$

which is well-defined as $(v, \mathrm{Init}(v)) \in W_0(\mathcal{G}'')$.

Now, let σ'' be a uniform positional winning strategy for Player 0 in the parity game $\mathcal{G}''$. We define a positional strategy σ' for $\mathcal{G}'$ by using $\max(v)$, i.e., the worst memory state Player 0 could be in at vertex v while still being able to win from there. Given a vertex $v \in W_0(\mathcal{G}')$, let $\sigma''(v, \max(v)) = (v', m')$. Using this, we define $\sigma'(v) = v'$. We show that σ' is a uniform winning strategy for Player 0 in $\mathcal{G}'$. Consider a play $\rho' = v_0 v_1 v_2 \cdots$ starting in $v_0 \in W_0(\mathcal{G}')$ consistent with σ', and $\rho'' = (v_0, m_0)(v_1, m_1)(v_2, m_2) \cdots$ its extended play in $\mathcal{A} \times \mathcal{M}$. A straightforward induction shows that for every j, we have $(v_j, m_j) \in W_0(\mathcal{G}'')$, so $\max(v_j) \geq m_j$.

By Lemma 3, $\rho' \in \mathrm{PCRR}(\Omega, I)$ if and only if $\rho'' \in \mathrm{Parity}(\Omega_{\mathcal{M}})$. Assume towards contradiction that the maximal color seen infinitely often in ρ'' is odd. This implies that the memory state $\bot$ appears finitely often, so after a position, say n, all memory states are different from $\bot$. Furthermore, from position n, we additionally have $\max(v_j) \leq \max(v_{j+1})$; this follows from the observation that if $\mathrm{Upd}(c, v) \neq \bot$, then $c \leq \mathrm{Upd}(c, v)$.

Consider $\rho^* = (v_n, \max(v_n))(v_{n+1}, \max(v_{n+1}))(v_{n+2}, \max(v_{n+2})) \cdots$. Since the sequence $(\max(v_j))_{j \geq n}$ is non-decreasing, it is ultimately constant. The suffix starting from there is consistent with σ'' and starts in $W_0(\mathcal{G}'')$, so it satisfies $\mathrm{Parity}(\Omega_{\mathcal{M}})$ since σ'' is a winning strategy. Consequently, ρ^* contains finitely many increment-vertices, so ρ'' as well. After the last increment-vertex, ρ^* and ρ'' have the same colors, but ρ'' does not satisfy $\mathrm{Parity}(\Omega_{\mathcal{M}})$, a contradiction.

To conclude this subsection, we prove that the upper bound $d + 1$ on the memory requirements of Player 1 proved in Corollary 2 is tight.

Lemma 5 *For every $d \geq 1$, there is a bounded parity game with costs $\mathcal{G}_d$ such that*

- *the arena of $\mathcal{G}_d$ is of linear size in d and there are d odd colors in $\mathcal{G}_d$,*
- *Player 1 has a uniform finite-state winning strategy for $\mathcal{G}_d$ from every vertex which is implemented with $d+1$ memory states, but*
- *there is a vertex from which Player 1 has no winning strategy that is implemented with less than $d+1$ memory states.*

Proof. We begin by describing the game by an example: Figure 3 depicts the game $\mathcal{G}_4$, where the numbers in the vertices denote their colors. Since each edge is an increment-edge, we do not label them as such in the picture. The arena consists of a hub vertex colored by 0 and four disjoint blades, which are identified by the odd color of their outermost vertex, i.e., by the colors $1, 3, 5$ and 7 (which is $2 \cdot 4 - 1$). From the hub, Player 0 can enter the blade for c at a vertex of color $c - 1$ (which is even) which has a self-loop and an edge to a vertex of color 8 (which answers every request in the game). This vertex has only one outgoing edge to a vertex of color c (this is the identifying color). Again, this vertex has only one successor, the hub. In general, the arena of $\mathcal{G}_d$ has d blades, one for each color in $\{1, 3, \ldots, 2d-1\}$, the hub has color 0, and the second vertex in each blade has color $2d$ and thereby answers every request. Furthermore, every edge is an increment-edge.

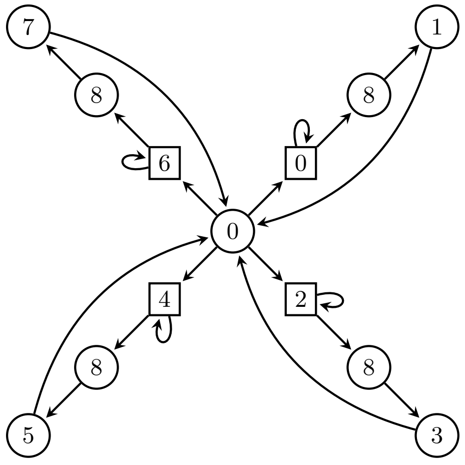

Fig. 3. The bounded parity game with costs $\mathcal{G}_4$ (every edge is an increment-edge).

At the hub, Player 0 picks a blade (say of color c) and then Player 1 decides whether to use the self-loop or to return to the hub. Note that Player 0 loses, if she enters the blade of color c while there is an open request of some color $c' > c$, since Player 1 can use the self-loop of the blade and thereby prevent an answer to the request c'. On the other hand, if Player 1 decides to leave the blade, all requests are answered and then color c is requested. Note that this request is never answered to by moving to the hub.

First, we show that Player 1 has a uniform finite-state winning strategy from every vertex that is implementable with $d+1$ memory states. The memory structure keeps track of the largest open request, i.e., we use states $1, 3, \dots, 2d-1$ and an additional state $\perp$ that is reached, if there is no open request. Now, assume the current memory state is m and the play is in a vertex of Player 1, which is uniquely identified by its even color c. If $m = \perp$, then Player 1 moves to the (unique) successor of color $2d$. Now, assume $m \neq \perp$, i.e., m is some odd color. If $m < c$, then Player 1 again leaves c by moving to the unique successor of color $2d$. If $m > c$, then Player 1 uses the self-loop forever.

Now, consider a play that is consistent with this strategy. If the current memory state is $\perp$, then a request is raised within the next three moves and the play returns to the hub, which implies that the memory is updated to some odd color m. From there, Player 0 has to move to some blade, say for color c (which is odd). If $m > c$, then Player 1 uses the self-loop at the vertex of color $c-1$ forever. The resulting play is winning for him, since the request of c is unanswered with cost ∞. On the other hand, if $m < c$, then Player 1 moves to the vertex of color c and then back to the hub. While doing this, the memory is updated to a larger state, namely c. Hence, the memory states along a play consistent with the strategy described above are increasing, which means that at some point Player 0 has to enter a blade for color $c < m$, where m is the current memory state, i.e., also an open request. Then, Player 1 will win by using the self-loop of this blade. Hence, the strategy described above is a winning strategy from every vertex and is implemented using $d+1$ memory states.

It remains to show that the upper bound $d+1$ is tight. To this end, consider a finite-state strategy τ for Player 1 that is winning from the hub, say τ is implemented by $(M, \mathrm{Init}, \mathrm{Upd})$. We show that M contains at least $d+1$ memory states. To this end, we define a sequence $m_0, m_1, \dots, m_d$ of $d+1$ memory states, as follows. Define $m_0 = \mathrm{Init}(v)$, where v is the hub. Now, Consider the play where Player 0 moves from the hub to the blade with color 1. Since τ is a winning strategy, Player 1 will use the self-loop of this blade only finitely often, i.e., the hub is reached again. We denote this play prefix by w_1 (which is consistent with τ) and define $m_1 = \mathrm{Upd}^+(w_1)$. Consider now the play where after w_1, Player 0 moves to the blade with color 3. Again, Player 1 will use the self-loop only finitely often and the hub is reached again. We denote the prolongation of w_1 through this blade by w_2 and define $m_2 = \mathrm{Upd}^+(w_1 w_2)$. This process is continued for each blade in ascending order. Since Player 1 has to leave each blade we obtain a sequence $m_0, m_1, \dots, m_d$ of memory states assumed at the visits of the hub and a play prefix $w_1 w_2 \cdots w_d$ that is consistent with σ, starts in the hub, and satisfies $\mathrm{Upd}^+(w_1 w_2 \cdots w_j) = m_j$ for every $j \geq 1$. Furthermore, each $w_1 w_2 \cdots w_j$ ends in the hub.

We argue that the states $m_1, m_1, \dots, m_d$ are pairwise distinct. Assume towards contradiction there are $j < j' \leq d$ such that $m_j = m_{j'}$. Then, the play

$\rho = w_1 \cdots w_j \cdot (w_{j+1} \cdots w_{j'})^\omega$ is consistent with τ. However, the maximal color seen infinitely often during ρ is $2d$ (which answers every request), and there is a uniform bound on the distance between the occurrences of $2d$. Hence, the play is winning for Player 0 in $\mathcal{G}_d$, contradicting the fact that τ is a winning strategy for Player 1. Hence, the states m_j are indeed pairwise distinct. Thus, every winning strategy has at least $d + 1$ memory states.

Note that every edge in $\mathcal{G}_d$ is an increment-edge, i.e., $\mathcal{G}_d$ is a bounded parity game. In [9] an upper bound of two on the memory requirements of Player 1 is claimed for bounded parity games. The games presented here refute this claim: there is no constant bound on the memory needed for Player 1 in bounded parity games.

4 Solving Parity Games with Costs via Bounded Parity Games with Costs

In this section, we show that being able to solve bounded parity games with costs suffices to solve parity games with costs. Our algorithm is based on the following lemma which formalizes this claim by relating the winning regions of Player 0 in the parity games with costs and the bounded parity game with costs in the same arena.

Lemma 6 *Let $\mathcal{G} = (\mathcal{A}, \mathrm{CostParity}(\Omega, \mathrm{Cst}))$ a parity game with costs and $\mathcal{G}' = (\mathcal{A}, \mathrm{BndCostParity}(\Omega, \mathrm{Cst}))$ a bounded parity games with costs on the same arena.*

1. *$W_0(\mathcal{G}') \subseteq W_0(\mathcal{G})$.*
2. *If $W_0(\mathcal{G}') = \emptyset$, then $W_0(\mathcal{G}) = \emptyset$.*

Proof. (1) This follows from the inclusion

$$\mathrm{BndCostParity}(\Omega, \mathrm{Cst}) \subseteq \mathrm{CostParity}(\Omega, \mathrm{Cst}) \ .$$

(2) Due to determinacy, if $W_0(\mathcal{G}') = \emptyset$, then we have $W_1(\mathcal{G}') = V$. Due to Corollary 2, Player 1 has a uniform finite-state strategy τ that is winning from every vertex v. Consider a play consistent with τ: either, for every b, there is a request that is open for the next b increment-edges (the request could be the same for every b), or the maximal color seen infinitely often is odd (i.e., there are infinitely many unanswered requests).

We define a strategy τ' for Player 1 as follows: it is guided by a counter b_{cur} which is initialized with 1. Assume a play starts in vertex v. The strategy τ' plays like τ until a request is open for b_{cur} increment-edges. If this is the case, b_{cur} is incremented and τ' plays like τ does from the current vertex (forgetting the history of the play constructed so far).

We show that τ' is winning in $\mathcal{G}$ from every vertex, which implies $W_0(\mathcal{G}) = \emptyset$. Let ρ' be a play that is consistent with τ' and distinguish two cases: if the counter is incremented infinitely often, then ρ' contains for every b a request that is open for at least b increment-edges, so $\rho' \notin \mathrm{CostParity}(\Omega, \mathrm{Cst})$. On the other hand, if b_{cur} is incremented only finitely often (say to value b), then there is a suffix ρ of ρ' that is consistent with the strategy τ. Since the counter is not incremented during ρ, every request in ρ is either answered with cost at most b or not answered, but only followed by at most b increment-edges. Hence τ ensures that the maximal color seen infinitely often in ρ is odd, i.e. $\rho \notin \mathrm{Parity}(\Omega)$, so $\rho' \notin \mathrm{Parity}(\Omega)$, and a fortiori $\rho' \notin \mathrm{CostParity}(\Omega, \mathrm{Cst})$.

Note that the strategy τ' is not finite-state. We have seen in Example 1 that Player 1 needs infinite memory in general. Indeed, the winning strategy for Player 1 described in the example proceeds as described above. It requests color 1 at vertex a, uses the loop at vertex b to keep the request unanswered for several steps and then forgets about this request. At this point, a new request has to be raised by moving from b back to a, thereby answering the old request at vertex c. This request is then kept unanswered for more increment-edges than the previous one, and this goes on ad infinitum.

To conclude this subsection, we show how Lemma 6 can be used to solve parity games with costs. Let $\mathcal{G} = (\mathcal{A}, \mathrm{CostParity}(\Omega, \mathrm{Cst}))$. The following algorithm proceeds by iteratively removing parts of $\mathcal{A}$ that are included in the winning region of Player 0 in $\mathcal{G}$: we have just proven that the winning region of Player 0 in the bounded parity game with costs in $\mathcal{A}$ is a subset of her winning region in the parity game with costs in the same arena. Thus we can remove it and its attractor. This is repeated until Player 0's winning region in the bounded parity game with costs is empty. In this case, her winning region in the parity game with costs is empty as well, again due to Lemma 6. This idea is implemented in the following algorithm.

Algorithm 1 A fixed-point algorithm for solving $(\mathcal{A}, \mathrm{CostParity}(\Omega, \mathrm{Cst}))$.

```
$j \leftarrow 0$; $W_j \leftarrow \emptyset$; $\mathcal{A}_j \leftarrow \mathcal{A}$
repeat
    $j \leftarrow j + 1$
    $X_j \leftarrow W_0(\mathcal{A}_{j-1}, \mathrm{BndCostParity}(\Omega, \mathrm{Cst}))$
    $W_j \leftarrow W_{j-1} \cup \mathrm{Attr}_0^{\mathcal{A}_{j-1}}(X_j)$
    $\mathcal{A}_j \leftarrow \mathcal{A}_{j-1} \setminus \mathrm{Attr}_0^{\mathcal{A}_{j-1}}(X_j)$
until $X_j = \emptyset$
return $W_j$
```

Example 2 *Running on the parity game with costs of Figure 1, Algorithm 1 computes $X_1 = \{g\}$ and $W_1 = \{f, g\}$, $X_2 = \{e\}$ and $W_2 = \{d, e, f, g\}$, and $X_3 = \emptyset$. Thus, it returns W_2, which is the winning region of Player 0 in the game.*

Next, we show the algorithm to be correct and bound its number of iterations.

Lemma 7 *Let $\mathcal{G}$ be a parity game with costs with n vertices. Algorithm 1 returns the winning region $W_0(\mathcal{G})$ after at most $n+1$ iterations.*

Proof. Let $\mathcal{G} = ((V, V_0, V_1, E), \mathrm{CostParity}(\Omega, \mathrm{Cst}))$ and let t be the last iteration with $X_t \neq \emptyset$, i.e., the algorithm returns W_t. We have $t \leq |V|$, since $\emptyset = W_0 \subsetneq W_1 \subsetneq \cdots \subsetneq W_t \subseteq V$ is a strictly increasing chain. Hence, the algorithm terminates after at most $|V| + 1$ iterations.

Next, we show $W_t \subseteq W_0(\mathcal{G})$: to this end, we define a strategy σ for Player 0 on W_t as follows: on sets X_j, which are winning regions of Player 0 in a bounded parity game with costs, we play using some uniform positional winning strategy for this game, which always exists due to Lemma 4. On the attractors $\mathrm{Attr}_0^{\mathcal{A}_{j-1}}(X_j)$ we play using some positional attractor strategy. Thus, σ is a positional strategy that is defined for every vertex in W_t. Next, we show that it is indeed a uniform positional winning strategy from W_t.

Every winning region X_j is a trap for Player 1 in $\mathcal{A}_{j-1}$. Hence, in the whole arena $\mathcal{A}$, Player 1 can leave X_j only to vertices in some $W_{j'}$ with $j' < j$. Player 0 on the other hand only moves to vertices in W_j. Similarly, if the play is in the attractor of some X_j, then it reaches X_j after at most $|V|$ steps or Player 1 moves to some $W_{j'}$ for some $j' < j$. Hence, every play ρ consistent with σ has a suffix ρ' that visits only vertices from some X_j and is consistent with the winning strategy for the corresponding bounded parity game with costs. So, $\rho' \in \mathrm{BndCostParity}(\Omega, \mathrm{Cst})$, which implies $\rho \in \mathrm{CostParity}(\Omega, \mathrm{Cst})$. Thus, σ is a winning strategy.

It remains to consider Player 1, i.e., to show that $V \setminus W_t \subseteq W_1(\mathcal{G})$. Note that $V \setminus W_t$ is the set of vertices of $\mathcal{A}_t$ and that we have

$$W_0(\mathcal{A}_t, \mathrm{BndCostParity}(\Omega, \mathrm{Cst})) = \emptyset \ .$$

Hence, by Lemma 6(2) we conclude $W_0(\mathcal{A}_t, \mathrm{CostParity}(\Omega, \mathrm{Cst})) = \emptyset$, i.e., Player 1 wins the parity game with costs in the arena $\mathcal{A}_t$ from every vertex. Since $V \setminus W_t$ is a trap for Player 0 (it is the complement of an attractor), it follows that Player 1 wins the parity game with costs in the arena $\mathcal{A}$ from every vertex in $V \setminus W_t$.

Corollary 3 *In parity games with costs, Player 0 has uniform positional winning strategies.*

Using Lemma 7, Theorem 1, and the fact that Algorithm 1 terminates after at most $n+1$ iterations, and therefore has to solve at most n bounded parity games with costs, we obtain the following result where again n is the number of vertices, m is the number of edges, and d is the number of colors in the game.

Theorem 2 *Given an algorithm that solves parity games in time $T(n, m, d)$, there is an algorithm that solves parity games with costs with a time complexity $O(n \cdot T(dn, dm, d+2))$.*

Also, we obtain the same computational complexity as for bounded parity games with costs.

Remark 6 *The following problem is in* $\mathbf{NP} \cap \mathbf{coNP}$*: given a parity game with costs* $\mathcal{G}$*,* $i \in \{0, 1\}$*, and a vertex* v*, is* $v \in W_i(\mathcal{G})$*?*

Proof. Use the characterization of W_0 as computed by Algorithm 1: the sets X_j can be determined in **NP** (respectively in **coNP**) due to Remark 5 and the attractors can be computed in (deterministic) linear time.

In the previous section section we have shown that one can recover a polynomial time algorithm for deciding the winning regions of bounded parity games. Hence, using Algorithm 1, we obtain the same for finitary parity games as well. Hence, we also recover polynomial time decidability of finitary parity games as a special case of our algorithm. However, this is not surprising since Algorithm 1 is the same one used in [9] to solve finitary parity games via solving bounded parity games.

5 Positional Winning Strategies for Bounded Parity Games with Costs via Scoring Functions

In the previous section, we have shown how to eliminate the memory introduced in the reduction from PCRR games to parity games, which proved the existence of uniform positional winning strategies for Player 0 in bounded parity games with costs. Using these strategies as building blocks, we also proved the existence of uniform positional winning strategies for Player 0 in parity games with costs. Intuitively, the memory used in the reduction keeps track of the largest open request, but Player 0 does not need this information to implement her winning strategy as proved in Lemma 4. Instead, she can always play assuming the worst situation that still allows her to win.

In this section we generalize this construction to arbitrary memory structures: we define a quality measure for play prefixes and then show that always playing like in the worst possible situation is a positional winning strategy. This gives an alternative proof of half-positional determinacy of (bounded) parity games with costs and presents a general framework that we believe to be applicable to other winning conditions as well.

We begin by defining a so-called scoring function for bounded parity games with costs that measures the quality of a play prefix (from Player 0's vantage point) by keeping track of the largest unanswered request, the number of increment-edges traversed since it was raised, and how often each odd color was seen since the last increment-edge. For the remainder of this section, we fix a bounded parity game with costs $\mathcal{G} = (\mathcal{A}, \mathrm{BndCostParity}(\Omega, \mathrm{Cst}))$ with arena $\mathcal{A} = (V, V_0, V_1, E)$, and a uniform finite-state winning strategy σ for Player 0 in $\mathcal{G}$. Let $\Omega(V) \subseteq \{0, 1, \ldots, \ell\}$, where we assume ℓ to be odd. Furthermore, let $d = \frac{\ell+1}{2}$ be the number of odd colors in $\{1, 3, \ldots, \ell\}$. Finally, let $t = |V| \cdot |\sigma|$, where $|\sigma|$ denotes the size of the memory structure implementing σ.

A proper (score-) sheet is a vector $(c, n, s_\ell, s_{\ell-2}, \dots, s_3, s_1)$ where c is an odd color in $\{1, 3, \dots, \ell\}$, $n \le t$, and $s_{c'} \le t$ for every c'. Finally, we use two special sheets denoted by $\bot$ and $\top$. The reversed ordering of the score values $s_\ell, s_{\ell-2}, \dots, s_3, s_1$ in the sheets is due to the max-parity condition, in which larger colors are more important than smaller ones. This is reflected by the fact that we compare sheets in the lexicographical order induced by $<$ on its components and add $\bot$ as minimal and $\top$ as maximal element. For example, $(3, 3, 0, 1, 1) < (3, 3, 1, 0, 3)$ and $\bot < s < \top$ for every sheet $s \ne \bot, \top$. As usual, we write $s \le s'$ if $s = s'$ or $s < s'$.

Next, we show how to update sheets along a play to use them as a quality measure for play prefixes. Let $s = (x_1, \dots, x_{d+2})$ be a (proper) sheet. We say that s is full in coordinate 1, if $x_1 = \ell$ (recall that ℓ is the largest possible value in the first coordinate), and that s is full in coordinate $k > 1$, if $x_k = t$ (recall that t is the largest possible value in all but the first coordinate). Let k be a coordinate and let $s = (x_1, \dots, x_{d+2})$ be a proper sheet.

- If 1 is the largest coordinate smaller than or equal to k that is not full in s, then incrementing s at coordinate k yields the sheet $(x_1 + 2, 0, \dots, 0)$. If $k > 1$, then we say that there is an overflow in coordinates $2, \dots, k$.
- If $k' > 1$ is the largest coordinate smaller than or equal to k that is not full in s, then incrementing s at coordinate k yields the sheet $(x_1, \dots, x_{k'-1}, x_{k'} + 1, 0, \dots, 0)$. If $k' < k$, then we say that there is an overflow in coordinates $k' + 1, \dots, k$.
- If there is no coordinate k' smaller than or equal to k that is not full in s, then incrementing s at coordinate k yields the sheet $\top$ and we say that there is an overflow in coordinates $1, \dots, k$.

Example 3 *Assume we have $\ell = 5$ and $t = 3$ and consider $s = (3, 3, 0, 1, 3)$. Then, s is full in coordinate 2 and 5, but not in coordinates 1, 3, and 4. Incrementing s at coordinate 1 or 2 yields the sheet $(5, 0, 0, 0, 0)$ (note that there is an overflow of coordinate 2 in the second case), incrementing at 3 yields $(3, 3, 1, 0, 0)$ while incrementing at 4 or 5 yields $(3, 3, 0, 2, 0)$ (and there is an overflow of coordinate 5 in the second case).*

Next, we show that the increment-operation and a reset-operation are compatible with the ordering.

Remark 7 *Let $x = (x_1, \dots, x_{d+2}) \le y = (y_1, \dots, y_{d+2})$ be two sheets and let k be a coordinate.*

1. *Let x' (respectively y') be obtained by incrementing x (respectively y) at coordinate k. Then, $x' \le y'$.*
2. *Let $x'' = (x_1, \dots, x_k, 0, \dots, 0)$ and $y'' = (y_1, \dots, y_k, 0, \dots, 0)$. Then, $x'' \le y''$.*

Now, we want to assign a sheet to every play prefix. To this end, we define the initial sheet $\mathrm{Sh}(v)$ of a vertex v by

$$\mathrm{Sh}(v) = \begin{cases} \bot & \text{if } \Omega(v) \text{ is even,} \\ (\Omega(v), 0, 0, \dots, 0) & \text{if } \Omega(v) \text{ is odd.} \end{cases}$$

Now, let $\mathrm{Sh}(wv)$ for $w \in V^*$ and $v \in V$ be already defined and let (v, v') be an edge. If $\mathrm{Sh}(wv) = \top$, then $\mathrm{Sh}(wvv') = \top$, and if $\mathrm{Sh}(wv) = \bot$, then $\mathrm{Sh}(wvv') = \mathrm{Sh}(v')$. Now, assume we have $\mathrm{Sh}(wv) = (c, n, s_\ell, \ldots, s_1)$. We have to distinguish several cases:

- if $\Omega(v') > c$, then $\mathrm{Sh}(wvv') = \mathrm{Sh}(v')$,
- if $\Omega(v') \leq c$ and $\mathrm{Cst}(v, v') = i$, then $\mathrm{Sh}(wvv')$ is obtained from $\mathrm{Sh}(wv)$ by incrementing the second coordinate (the one associated with costs),
- if $\Omega(v') \leq c$, $\mathrm{Cst}(v, v') = \varepsilon$ and $\Omega(v')$ even, then the scores for the colors that are answered by $\Omega(v')$ are reset to zero, i.e.,

$$\mathrm{Sh}(wvv') = (c, n, s_\ell, \ldots, s_{\Omega(v')+1}, 0, \ldots, 0) ,$$

- if $\Omega(v') \leq c$, $\mathrm{Cst}(v, v') = \varepsilon$ and $\Omega(v')$ odd, then $\mathrm{Sh}(wvv')$ is obtained from $\mathrm{Sh}(wv)$ by incrementing the coordinate storing the score for $\Omega(v')$.

Let $\mathrm{Sh}(w) = (c, n, s_\ell, \ldots, s_1)$. To simplify our notation in the following proofs, we define $\mathrm{Req}(w) = c$, $\mathrm{ReqCst}(w) = n$, and $\mathrm{Sc}_{c'}(w) = s_{c'}$. If $\mathrm{Sh}(w) = \bot$ or $\mathrm{Sh}(w) = \top$, then we leave these functions undefined. Furthermore, let $\mathrm{Lst}(w)$ denote the last vertex of a non-empty finite play w.

In the following, we show that the scoring function has three properties that we need to prove our result. We begin by showing that it is a congruence.

Lemma 8 *If* $\mathrm{Lst}(x) = \mathrm{Lst}(y)$ *and* $\mathrm{Sh}(x) \leq \mathrm{Sh}(y)$*, then* $\mathrm{Sh}(xv) \leq \mathrm{Sh}(yv)$ *for every* $v \in V$.

Before we begin the proof we state the following useful facts.

Remark 8 *Let* $w \in V^*$ *and* $v \in V$.

1. *If* $\mathrm{Req}(w) \neq \mathrm{Req}(wv)$*, then* $\mathrm{Sh}(wv) = \mathrm{Sh}(v)$.
2. *If* $\Omega(v)$ *is odd, then* $\mathrm{Sh}(wv) \geq \mathrm{Sh}(v)$.

We now prove Lemma 8.

Proof. If $\mathrm{Sh}(x) = \mathrm{Sh}(y)$, then $\mathrm{Sh}(xv) = \mathrm{Sh}(yv)$, since the sheets of xv and yv only depend on the sheets of x and y (which are equal) and the last edges of xv and yv (which are also equal).

So, consider the case $\mathrm{Sh}(x) < \mathrm{Sh}(y)$. First, assume we have $\mathrm{Sh}(x) = \bot$, which implies $\mathrm{Sh}(xv) = \mathrm{Sh}(v)$. If $\Omega(v)$ is even, then $\mathrm{Sh}(v) = \bot$ and we are done, since $\bot$ is the minimal element. Otherwise, applying Remark 8(2) to yv yields the desired result. As a last special case assume we have $\mathrm{Sh}(y) = \top$, which implies $\mathrm{Sh}(yv) = \top$. As $\top$ is the maximal element, we have $\mathrm{Sh}(xv) \leq \mathrm{Sh}(yv)$.

We are left with the case $\bot < \mathrm{Sh}(x) < \mathrm{Sh}(y) < \top$ and have to consider two subcases:

1. First, assume we have $\mathrm{Req}(x) = \mathrm{Req}(y)$. We consider several subcases.
   (a) If $\Omega(v) > \mathrm{Req}(x) = \mathrm{Req}(y)$, then $\mathrm{Sh}(xv) = \mathrm{Sh}(yv) = \mathrm{Sh}(v)$.

   (b) If $\Omega(v) \leq \mathrm{Req}(x) = \mathrm{Req}(y)$ and $\mathrm{Cst}(\mathrm{Lst}(x), v) = i$, then both $\mathrm{Sh}(xv)$ and $\mathrm{Sh}(yv)$ are obtained by incrementing $\mathrm{Sh}(x)$ and $\mathrm{Sh}(y)$ respectively at the second coordinate. Hence, Remark 7(1) yields the desired result.
   (c) If $\Omega(v) \leq \mathrm{Req}(x) = \mathrm{Req}(y)$, $\mathrm{Cst}(\mathrm{Lst}(x), v) = \varepsilon$, and $\Omega(v)$ is even, then both $\mathrm{Sh}(xv)$ and $\mathrm{Sh}(yv)$ are obtained by reseting the scores for every c' smaller than $\Omega(v)$. Hence, Remark 7(2) yields the desired result.
   (d) If $\Omega(v) \leq \mathrm{Req}(x) = \mathrm{Req}(y)$, $\mathrm{Cst}(\mathrm{Lst}(x), v) = \varepsilon$, and $\Omega(v)$ is odd, then both $\mathrm{Sh}(xv)$ and $\mathrm{Sh}(yv)$ are obtained by incrementing $\mathrm{Sh}(x)$ and $\mathrm{Sh}(y)$ respectively at the coordinate storing the score for $\Omega(v)$. Hence, Remark 7(1) yields the desired result.
2. Now, assume we have $\mathrm{Req}(x) < \mathrm{Req}(y)$. Note that $\mathrm{Sh}(xv) = \top$ is impossible in this case, since the first coordinate of x is not full, as it is strictly smaller than $\mathrm{Req}(y)$. We again have to consider several subcases:
   (a) If $\Omega(v) > \mathrm{Req}(y) > \mathrm{Req}(x)$, then $\mathrm{Sh}(xv) = \mathrm{Sh}(yv) = \mathrm{Sh}(v)$.
   (b) If $\Omega(v) = \mathrm{Req}(y) > \mathrm{Req}(x)$, then we have $\mathrm{Req}(xv) > \mathrm{Req}(x)$ and an application of Remark 8(1) and 8(2) (to yv) yields the desired result.
   (c) Finally, assume we have $\Omega(v) < \mathrm{Req}(y)$. We again have to consider three subcases.
      i. If we have $\mathrm{Sh}(yv) = \top$, then we are done.
      ii. Assume we have $\mathrm{Req}(yv) > \mathrm{Req}(y)$. The following inequalities hold: $\mathrm{Req}(xv) \leq \mathrm{Req}(x){+}2 \leq \mathrm{Req}(y) < \mathrm{Req}(yv)$, where the first one is due to the fact that in the first component of $\mathrm{Sh}(xv)$ can only increase due to an increment, and the second one due to $\mathrm{Req}(x) < \mathrm{Req}(y)$. Hence, we have $\mathrm{Sh}(xv) < \mathrm{Sh}(yv)$ in this case.
      iii. Finally, consider the case where $\mathrm{Req}(yv) = \mathrm{Req}(y)$. If $\mathrm{Req}(xv) < \mathrm{Req}(yv)$, then we are done. So, assume we have $\mathrm{Req}(xv) = \mathrm{Req}(yv)$. Then, the first component of $\mathrm{Sh}(x)$ is increased to obtain $\mathrm{Sh}(xv)$, which implies that all other components of $\mathrm{Sh}(xv)$ are equal to zero. Hence, we have $\mathrm{Sh}(xv) \leq \mathrm{Sh}(yv)$. □

We continue by showing that the sheets of a play ρ being bounded is a sufficient condition for ρ satisfying the bounded cost-parity condition.

Lemma 9 *If the sheets of all prefixes of a play ρ are strictly smaller than $\top$, then $\rho \in \mathrm{BndCostParity}(\Omega, \mathrm{Cst})$.*

Proof. We prove the converse, i.e., if $\rho \notin \mathrm{BndCostParity}(\Omega, \mathrm{Cst})$, then there is a prefix of ρ whose sheet is $\top$. First, assume that for every b there is a request (say of color c) that is open for at least b increment-edges. Then, the second component of the sheets is incremented every time an increment-edge is traversed before the request is answered. Also, the first component is increased every time the second component overflows or every time a larger odd color is visited. Note that it is not reset in this interval, as the request of c is not answered. Hence, if we pick b large enough, the first component overflows as well. This yields the sheet $\top$.

Now assume the maximal color seen infinitely often, call it c, is odd. We may assume that ρ has only finitely many increment-edges, as we are in the first case otherwise. Pick a position of ρ such that the maximal color appearing after this position is c and such that no increment-edge is traversed after this position. After this position, the coordinate storing the score for c is incremented again and again. Furthermore, this coordinate (and all to the left of this one) are only reset in case of an overflow, which means that there is a coordinate to the left that is incremented. Thus, every coordinate to the left of the one storing the score for c is incremented again and again, too. Hence, the first component overflows at some point, which yields the sheet $\top$.

Recall that the entries in all but the first component of a (proper) sheet are bounded by $t = |V| \cdot |\sigma|$, where σ is a uniform finite-state winning strategy for Player 0 in $\mathcal{G} = (\mathcal{A}, \mathrm{BndCostParity}(\Omega, \mathrm{Cst}))$. Next, we show that this strategy keeps the sheets smaller than $\top$.

Lemma 10 *Let ρ be starting in $W_0(\mathcal{G})$ and consistent with σ. Then, the sheets of all prefixes of ρ are strictly smaller than $\top$.*

Proof. First, we show that every request in ρ is answered with cost less than or equal to t or followed by at most t increment-edges. Towards a contradiction, assume there is a request that is followed by $t + 1$ increment-edges, but no answer before the last of these increment-edges. Then, there are two positions in this interval that have the same vertex, the memory structure implementing σ assumes the same state after both positions, and there is at least one increment-edge between these positions. Hence, there is also a play consistent with σ and starting in $W_0(\mathcal{G})$ that contains an unanswered request with cost ∞. However, this contradicts the fact that σ is a winning strategy.

Similarly, one can show that ρ has no infix that contains $t + 1$ vertices of some odd color c, but no vertex of a larger color. Using the second property, a simple induction over the number of odd colors (starting with 1) shows that no coordinate storing a score s_c overflows during ρ. Using this and the first property shows that the second coordinate does not overflow either. Finally, if the second component does not overflow, then the first component does not overflow either, since it stores the largest unanswered request in this case. Hence, the sheets of ρ are strictly smaller than $\top$.

Now we are able to prove our main technical result of this section: using the score-sheets we can turn an arbitrary uniform finite-state winning strategy into a positional one. For every $v \in V$, let P_v denote the set of play prefixes that begin in $W_0(\mathcal{G})$, are consistent with σ, and end in v. Due to Lemma 10, the sheets of the prefixes in P_v are strictly smaller than $\top$. Hence, for every nonempty P_v there exists a play prefix $\max_v \in P_v$ such that $\mathrm{Sh}(w) \leq \mathrm{Sh}(\max_v) < \top$ for every $w \in P_v$. We define a positional strategy σ' by $\sigma'(wv) = \sigma(\max_v)$.

Lemma 11 *The strategy σ' is a uniform positional winning strategy for Player* 0 *in $\mathcal{G}$.*

Proof. An inductive application of Lemma 8 shows that we have $\mathrm{Sh}(\rho_0 \cdots \rho_n) \leq \mathrm{Sh}(\max_{\rho_n})$ for every n and every play ρ that is consistent with σ'. Hence, the sheets of ρ are strictly smaller than $\top$, which implies $\rho \in \mathrm{BndCostParity}(\Omega)$ due to Lemma 9.

In the preliminary version of this work [18], we presented a similar construction, the main difference being that we did not use overflows there, but updated a sheet to $\top$ if a full coordinate is incremented. This construction can also shown to be correct, but the proof of the analogue of Lemma 11 in [18] (called Lemma 15 there) has a gap (the claim in its last line is incorrect). This gap can be closed using pumping arguments which rely on properties of the bounded cost-parity condition. Since one of our aims in this section is to give a general framework that works for other winning conditions as well, we refrained from presenting the fix and instead changed the definition of the sheets (adding overflows) to achieve this goal: indeed, our construction only relies on the following properties:

1. The score-sheets constitute a finite total order.
2. The score-sheet function is a congruence w.r.t. this order.
3. If the score-sheets of a play are strictly smaller than the maximal element, then it is winning for Player 0.
4. A finite-state winning strategy allows only plays whose score-sheets are strictly smaller than the maximal element.

It follows that for every winning condition for which one can define a scoring function meeting these conditions, one can turn a finite-state winning strategy into a positional one. For example, one could extend the sheets presented above by a new first coordinate that counts how often the second coordinate (the largest open request) overflows, which corresponds to requests that are open for *many* increment-edges. Since a finite-state winning strategy for a parity game with costs bounds the number of such requests, our framework is applicable to parity games with costs as well.

6 Streett Games with Costs

In this section, we present an algorithm to solve Streett and bounded Streett games with costs following the same ideas as in the section about (bounded) parity games with costs, and prove **EXPTIME**-completeness of the corresponding decision problems. From our algorithm, we also obtain upper bounds on the memory requirements of both players.

The main result of this section is the following theorem. Here, n is the number of vertices, m is the number of edges, and d is the number of Streett pairs in the game.

Theorem 3 *Given an algorithm that solves Streett games in time $T(n, m, d)$, there is*

1. *an algorithm that solves bounded Streett games with costs with a time complexity $O(T(2^d n, 2^d m, 2d))$, and*

2. *an algorithm that solves Streett games with costs with a time complexity $O(n \cdot T(2^d n, 2^d m, 2d))$.*

We begin by considering bounded games and again assume for every c that no vertex has both an incoming increment-edge (w.r.t. Cst_c) and an incoming epsilon-edge (again, w.r.t. Cst_c). Having different types of incoming edges with respect to different cost-functions is allowed. This property can again be established by subdividing edges. Assuming this, let I_c denote the vertices with incoming increment-edges w.r.t. Cst_c. Then, $\mathrm{coBüchi}(I_c) = \{\rho \mid \mathrm{Cst}_c(\rho) < \infty\}$ is the set of plays with finitely many increment-edges w.r.t. Cst_c. Let $I = (I_c)_{c \in [d]}$. Furthermore, we define $\mathrm{RR}(Q_c, P_c)$ to be the set of plays in which every request of c is eventually answered. Finally, we define

$$\mathrm{SCRR}(\Gamma, I) = \bigcap_{c \in [d]} \left[\left(\mathrm{Streett}(Q_c, P_c) \cap \mathrm{coBüchi}(I_c)\right) \ \cup \ \mathrm{RR}(Q_c, P_c) \right] \ ,$$

which is ω-regular. This condition is a relaxation of the bounded cost-Streett condition, as we have $\mathrm{SCRR}(\Gamma) \supseteq \mathrm{BndCostStreett}(\Gamma)$.

Lemma 12 *Let $\mathcal{G} = (\mathcal{A}, \mathrm{BndCostStreett}(\Gamma, \overline{\mathrm{Cst}}))$ and $\mathcal{G}' = (\mathcal{A}, \mathrm{SCRR}(\Gamma, I))$, where I is defined as above. A winning strategy for Player i in $\mathcal{G}'$ from a set W is also a winning strategy for Player i in $\mathcal{G}$ from W. Especially, $W_i(\mathcal{G}) = W_i(\mathcal{G}')$ for $i \in \{0, 1\}$.*

The proof of this lemma is similar to the one for Lemma 2 and relies on finite-state determinacy of ω-regular games.

Next, we show how to reduce $(\mathcal{A}, \mathrm{SCRR}(\Gamma, I))$ to a classical Streett game: first, we add a memory structure $\mathcal{M}$ of size 2^d that keeps track of the open requests during a play and let F_c denote the vertices in which request c is not open. Then, we have $(\mathcal{A}, \mathrm{SCRR}(\Gamma, I)) \leq_{\mathcal{M}} (\mathcal{A} \times \mathcal{M}, L)$ with

$$L = \bigcap_{c \in [d]} \left[\left(\mathrm{Streett}(Q_c, P_c) \cap \mathrm{coBüchi}(I_c)\right) \ \cup \ \mathrm{Büchi}(F_c) \right] \ ,$$

i.e., we have reduced the request-response conditions $\mathrm{RR}(Q_c, P_c)$ to Büchi conditions[3]. Finally, we have

$$L = \bigcap_{c \in [d]} \mathrm{Streett}(Q_c, P_c \cup F_c) \ \cap \ \mathrm{Streett}(I_c, F_c) \ ,$$

which is a Streett condition. Thus, we have reduced $(\mathcal{A}, \mathrm{SCRR}(\Gamma, I))$ to a Streett game in an arena that is exponential in d with $2d$ Streett pairs. This proves the first claim of Theorem 3. Furthermore, we obtain the following upper bound on the size of finite-state winning strategies. Here, we use the fact that Player 0 has finite-state winning strategies of size $d!$ in Streett games with d pairs (which is tight), while Player 1 has positional winning strategies [14]. Note that the lower bound of $d!$ for Player 0 is also a lower bound for her in (bounded) Streett games with costs, since classical Streett games are a special case of both.

[3] $\mathrm{Büchi}(F_c)$ is the set of plays visiting F_c infinitely often.

Remark 9

1. *In bounded Streett games with costs, Player* 0 *has uniform finite-state winning strategies of size* $2^d(2d)!$*, where* d *is the number of Streett pairs.*
2. *In bounded Streett games with costs, Player* 1 *has uniform finite-state winning strategies of size* 2^d*, where* d *is the number of Streett pairs.*

Now, consider Streett games with costs: we again show that solving the bounded variant suffices to solve such games.

Lemma 13 *Let* $\mathcal{G} = (\mathcal{A}, \mathrm{CostStreett}(\Gamma, \overline{\mathrm{Cst}}))$ *be a Streett game with costs and* $\mathcal{G}' = (\mathcal{A}, \mathrm{BndCostStreett}(\Gamma, \overline{\mathrm{Cst}}))$ *be a bounded Streett game with costs on the same arena.*

1. $W_0(\mathcal{G}') \subseteq W_0(\mathcal{G})$.
2. *If* $W_0(\mathcal{G}') = \emptyset$*, then* $W_0(\mathcal{G}) = \emptyset$.

The proof is exactly the same as the one for Lemma 6. Also, Algorithm 1 (where X_j is now Player 0's winning region in the bounded Streett game with costs) works for this pair of winning conditions as well. This proves the second claim of Theorem 3.

Furthermore, using the same construction as in the proof of Lemma 7, one can built a winning strategy for a Streett game with costs out of the winning strategies for the bounded Streett games with costs solved by (the modified) Algorithm 1. By reusing memory states, we obtain the following upper bound for Player 0.

Remark 10 *In Streett games with costs, Player* 0 *has uniform finite-state winning strategies of size* $2^d(2d)!$*, where* d *is the number of Streett pairs.*

Player 1 on the other hand needs infinite memory in Streett games with costs, as witnessed by the game in Example 1, which can be easily transformed into a Streett game with costs.

Using the algorithm presented in [26], which solves a Streett game with a time complexity $O(mn^d dd!)$, one can solve (bounded) Streett games with costs in exponential time, although the Streett games that need to be solved are of exponential size (but only in d). Together with the **EXPTIME**-hardness of solving bounded and finitary Streett games[4], which are a special case of (bounded) Streett games with costs, we obtain the following result.

Theorem 4 *The following problem is* **EXPTIME***-complete: Given a (bounded) Streett game with costs* $\mathcal{G}$*,* $i \in \{0, 1\}$*, and a vertex* v*, is* $v \in W_i(\mathcal{G})$*.*

Again, we recover the results on bounded Streett and finitary Streett games (winning regions for both games can be decided in **EXPTIME**) proved in [9] as special cases of our algorithm, since $\mathrm{SCRR}(\Gamma, \overline{\mathrm{Cst}})$ is equal to a request-repsonse condition in case every edge is an increment-edge. Winning regions in request-response games can be determined in **EXPTIME** [28].

[4] Shown in unpublished work by Chatterjee, Henzinger, and Horn, obtained by slightly modifying the proof of **EXPTIME**-hardness of solving request-response games [10].

7 Conclusion

We introduced infinite games with cost conditions, generalizing both classical conditions and finitary conditions. For parity games with costs, we proved half-positional determinacy and that solving these games is not harder than solving parity games. Furthermore, the corresponding decision problem is in **NP**$\cap$**coNP**. For Streett games with costs, we showed that Player 0 has finite-state winning strategies and that solving these games is not harder than solving finitary Streett games and can be done by solving linearly many (classical) Streett games of exponential size (in the number of Streett pairs). Our results unify the previous results on both classical and finitary variants. Table 1 sums up all our results and compares them to related games. Here, d denotes the number of odd colors in the game and "exponential" is always meant to be "exponential in the number of Streett-pairs". The memory bounds for the different types of parity games are tight, while there are gaps between the exponential lower and the exponential upper bounds for the different types of Streett games with costs.

winning condition	computational complexity	memory Player 0	memory Player 1
parity	**NP** $\cap$ **coNP**	positional	positional
bounded parity	**PTIME**	positional	$d+1$
finitary parity	**PTIME**	positional	infinite
bounded cost-parity	**NP** $\cap$ **coNP**	positional	$d+1$
cost-parity	**NP** $\cap$ **coNP**	positional	infinite
Streett	**coNP**-complete	exponential	positional
bounded Streett	**EXPTIME**-complete	exponential	exponential
finitary Streett	**EXPTIME**-complete	exponential	infinite
bounded cost-Streett	**EXPTIME**-complete	exponential	exponential
cost-Streett	**EXPTIME**-complete	exponential	infinite

Table 1. Overview of results

Let us discuss two variations of the games presented here. In a parity game with costs, the requests and responses are hierarchical and there is a single cost-function that is used for every request. On the other hand, in Streett games with costs, the requests and responses are independent and there is a cost-function for every pair of requests and responses. Thus, there are two other possible combinations.

First, consider parity games with multiple cost-functions (one for each odd color): a reduction from QBF shows that solving such games is **PSPACE**-hard. On the other hand, the problem is in **EXPTIME**, since every such game is a Streett game with costs. Furthermore, one can show that Player 0 needs exponential memory (in the number of odd colors) to implement her winning strategies. All these results even hold for the bounded variant of these game, which is defined as one would expect. In these games, both players need exponential memory. In further research we aim at closing the gap in complexity of solving parity games with multiple cost-functions.

The second variation are Streett games with a single cost-function. Solving finitary Streett games is already **EXPTIME**-complete and our lower bounds on memory requirements are derived from Streett games. Note that both finitary Streett and classical Streett games can be seen as Streett games with a single cost-function. Thus, the results mentioned above already hold for such games. Hence, Streett games with a single cost-function are as hard as Streett games with multiple cost-functions.

Finally, there are at least two other directions to extend our results presented here: first, our winning conditions do not cover all acceptance conditions (for automata) discussed in [4, 27]. In ongoing research, we investigate whether our techniques are applicable to these more expressive conditions and to winning conditions specified in weak-MSO with the unbounding quantifier [3, 5]. Finally, one could add decrement-edges.

References

1. Roderick Bloem, Krishnendu Chatterjee, Thomas A. Henzinger, and Barbara Jobstmann. Better quality in synthesis through quantitative objectives. In Ahmed Bouajjani and Oded Maler, editors, *CAV*, volume 5643 of *LNCS*, pages 140–156. Springer, 2009.
2. Mikołaj Bojańczyk. A bounding quantifier. In Jerzy Marcinkowski and Andrzej Tarlecki, editors, *CSL*, volume 3210 of *LNCS*, pages 41–55. Springer, 2004.
3. Mikołaj Bojańczyk. Weak MSO with the unbounding quantifier. *Theory Comput. Syst.*, 48(3):554–576, 2011.
4. Mikołaj Bojańczyk and Thomas Colcombet. Bounds in ω-regularity. In LICS [21], pages 285–296.
5. Mikolaj Bojanczyk and Szymon Torunczyk. Weak MSO+U over infinite trees. In Christoph Dürr and Thomas Wilke, editors, *29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012)*, volume 14 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 648–660, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
6. Tomás Brázdil, Krishnendu Chatterjee, Antonín Kucera, and Petr Novotný. Efficient controller synthesis for consumption games with multiple resource types. In P. Madhusudan and Sanjit A. Seshia, editors, *CAV*, volume 7358 of *LNCS*, pages 23–38. Springer, 2012.
7. Pavol Černý, Krishnendu Chatterjee, Thomas A. Henzinger, Arjun Radhakrishna, and Rohit Singh. Quantitative synthesis for concurrent programs. In Ganesh Gopalakrishnan and Shaz Qadeer, editors, *CAV*, volume 6806 of *LNCS*, pages 243–259. Springer, 2011.
8. Krishnendu Chatterjee and Laurent Doyen. Energy parity games. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *ICALP (2)*, volume 6199 of *LNCS*, pages 599–610. Springer, 2010.
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*, volume 6638 of *LNCS*, pages 227–237. Springer, 2011.
11. Krishnendu Chatterjee, Thomas A. Henzinger, and Marcin Jurdziński. Mean-payoff parity games. In *LICS*, pages 178–187. IEEE Computer Society, 2005.
12. Thomas Colcombet. The theory of stabilisation monoids and regular cost functions. In Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris E. Nikoletseas, and Wolfgang Thomas, editors, *ICALP (2)*, volume 5556 of *LNCS*, pages 139–150. Springer, 2009.
13. Thomas Colcombet and Christof Löding. Regular cost functions over finite trees. In *LICS*, pages 70–79. IEEE Computer Society, 2010.
14. Stefan Dziembowski, Marcin Jurdziński, and Igor Walukiewicz. How much memory is needed to win infinite games? In *LICS*, pages 99–110, 1997.
15. E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *FOCS*, pages 368–377. IEEE, 1991.
16. E. Allen Emerson and Charanjit S. Jutla. The complexity of tree automata and logics of programs. *SIAM J. Comput.*, 29(1):132–158, 1999.
17. Uli Fahrenberg, Line Juhl, Kim G. Larsen, and Jirí Srba. Energy games in multiweighted automata. In Antonio Cerone and Pekka Pihlajasaari, editors, *ICTAC*, volume 6916 of *LNCS*, pages 95–115. Springer, 2011.
18. Nathanaël Fijalkow and Martin Zimmermann. Cost-Parity and Cost-Streett Games. In Deepak D'Souza, Telikepalli Kavitha, and Jaikumar Radhakrishnan, editors, *FSTTCS 2012*, volume 18 of *LIPIcs*, pages 124–135, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
19. Kosaburo Hashiguchi. Limitedness theorem on finite automata with distance functions. *J. Comput. Syst. Sci.*, 24(2):233–244, 1982.
20. Florian Horn, Wolfgang Thomas, and Nico Wallmeier. Optimal strategy synthesis in request-response games. In Sung Deok Cha, Jin-Young Choi, Moonzoo Kim, Insup Lee, and Mahesh Viswanathan, editors, *ATVA*, volume 5311 of *LNCS*, pages 361–373. Springer, 2008.
21. *21th IEEE Symposium on Logic in Computer Science (LICS 2006), 12-15 August 2006, Seattle, WA, USA, Proceedings*. IEEE Computer Society, 2006.
22. Donald A. Martin. Borel determinacy. *Annals of Mathematics*, 102:363–371, 1975.
23. Robert McNaughton. Playing infinite games in finite time. In Arto Salomaa, Derick Wood, and Sheng Yu, editors, *A Half-Century of Automata Theory*, pages 73–91. World Scientific, 2000.
24. Andrzej Mostowski. Games with forbidden positions. Technical Report 78, University of Gdańsk, 1991.
25. David E. Muller and Paul E. Schupp. Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems of Rabin, McNaughton and Safra. *Theor. Comput. Sci.*, 141(1&2):69–107, 1995.
26. Nir Piterman and Amir Pnueli. Faster solutions of Rabin and Streett games. In LICS [21], pages 275–284.
27. Michael Vanden Boom. Weak cost monadic logic over infinite trees. In Filip Murlak and Piotr Sankowski, editors, *MFCS*, volume 6907 of *LNCS*, pages 580–591. Springer, 2011.
28. Nico Wallmeier, Patrick Hütten, and Wolfgang Thomas. Symbolic synthesis of finite-state controllers for request-response specifications. In Oscar H. Ibarra and Zhe Dang, editors, *CIAA*, volume 2759 of *LNCS*, pages 11–22. Springer, 2003.

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