Optimal Strategy Synthesis
for Request-Response Games *

Florian Horn[†], Wolfgang Thomas[††], Nico Wallmeier[††] and
Martin Zimmermann[†††]

[†]LIAFA, Université Denis Diderot - Paris 7, 75205 Paris CEDEX 13, France
`florian.horn@liafa.jussieu.fr`
[††]Lehrstuhl für Informatik 7, RWTH Aachen University, 52056 Aachen, Germany
`{thomas, wallmeier}@automata.rwth-aachen.de`
[†††]Reactive Systems Group, Saarland University, 66123 Saarbrücken, Germany
`zimmermann@react.uni-saarland.de`

Abstract

We show the existence and effective computability of optimal winning
strategies for request-response games in case the quality of a play is mea-
sured by the limit superior of the mean accumulated waiting times between
requests and their responses.

1 Introduction

Request-response (RR) conditions are ubiquitous in the formal verification of
reactive systems, e.g., every request to access a shared resource is eventually
granted. Formally, such a condition is expressed as a pair (Q, P) of state prop-
erties, the first one representing the requests and the second one representing
the responses. The corresponding request-response condition is satisfied if each
time a state in Q is visited, then at this or a later time a state in P is visited, i.e.,
every request is answered by a response. In linear temporal logic, this require-
ment is formalized as $\mathbf{G}(Q \to \mathbf{F}P)$. In practice, one considers conjunctions of
request-response conditions which occur in conjunction with safety conditions.
In the following, we assume (w.l.o.g.) the state space to be restricted to those
states satisfying the safety conditions. Alternatively, one can encode a safety
condition as a request-response condition with empty set of responses.

Synthesis for RR conditions was investigated in a series of works [7, 17, 18]
considering request-response games. The winning condition of such a game is

a conjunction of request-response conditions, i.e., it is given by a finite family $(Q_j, P_j)_{j \in [k]}$ of k request-response pairs. Player 0 wins a play (an infinite path through the finite game graph) if the request-reponse condition (Q_j, P_j) is satisfied for every j. Wallmeier et al. [17] presented a reduction from RR to Büchi games using an exponentially-sized memory structure and thereby gave an EXPTIME-algorithm and an exponential upper bound on the memory requirements for both players. Furthermore, they proved an exponential lower bound on the memory requirements for Player 0. These results where complemented by Chatterjee et al. [7] who proved EXPTIME-completeness of solving RR games and tight exponential lower bounds on the memory requirements for both players.

In request-response games, there is an intuitive notion of the waiting time between a request and its earliest response, which can be used to measure the quality of plays and strategies (from Player 0's point of view). There are several ways to aggregate the waiting times of a play to measure the quality of this play. A simple one is to take the maximal waiting time reached during the play and measure the quality of a strategy in terms of the maximal waiting time it allows during a play that is consistent with the strategy. It is straightforward to show that a finite-state winning strategy of size s bounds the waiting times by ns, where n is the number of vertices of the game graph. Thus, the exponential upper bound on memory requirements in RR games also yields an exponential upper bound on the maximal waiting time during plays consistent with this strategy. On the other hand, there are games witnessing matching exponential lower bounds on the maximal waiting time allowed by winning strategies. These results are presented in Section 3.

However, our main result pertains to a more sophisticated way of aggregating waiting times along a play: the value of a play is defined to be the limit superior of the mean accumulated waiting times of the play. In turn, the value of a strategy is the supremum over the values of all plays that are consistent with it. These considerations add a quantitative aspect to the synthesis problem that goes beyond the mere satisfaction of the winning condition and deciding whether there is a bound on the waiting time, by asking for an optimal winning strategy. Our main result states that an optimal strategy always exists, can be presented as a finite-state strategy, and can be effectively computed. To show this, we first prove an upper bound on the value of optimal strategies. Then, we show that for every strategy whose value is smaller than the bound (in particular optimal ones) there is a strategy of smaller or equal value that bounds the waiting times by some doubly-exponential bound. Thus, the search space for an optimal winning strategy is finite and the problem of finding one can be reduced to computing an optimal strategy for a mean-payoff game of doubly-exponential size which encodes the search space.

Our result fits into a larger series of works which aim at extending synthesis from a decision problem to an optimization problem by asking for optimal winning strategies according to a given quality measure for the winning condition under consideration, e.g., the use of mean-payoff objectives and weighted automata to model quantitative aspects in the winning condition [2, 4, 8] and

quantitative strengthenings of parity and Streett conditions [6, 13]. In another line of research, linear temporal logic is extended by parameterized operators [1, 15] equipped with variables that bound their scope, e.g., the formula $\mathbf{G}(Q \to \mathbf{F}_{\leq x}P)$ of parametric linear temporal logic (PLTL) is satisfied, if there is some value $\alpha(x)$ such that every request is answered within $\alpha(x)$ steps. Thus, measuring the quality of plays and strategies in RR games using the maximal waiting time is expressible in PLTL. Asking whether there exists a variable valuation and a winning strategy for Player 0 in a game with a PLTL winning condition is 2ExpTime-complete, while optimal winning strategies can be computed in triply-exponential time [19].

Finally, there has been a lot of interest in so-called energy games, whose winning conditions ask for the existence of an initial amount of energy such that a positive energy level is maintained throughout the play, where energy is consumed or recharged while traversing edges of the game graph. Solving energy games with multiple resources is in general intractable [12] while so-called consumption games, a subclass of energy games, are shown to be tractable in [3]. Energy parity games, whose winning conditions are a conjunction of a (single resource) energy and a parity condition, can be solved in $\mathrm{NP} \cap \mathrm{CO\text{-}NP}$ and one player (the spoiling one) has positional winning strategies while the other needs exponential memory [5].

The paper is structured as follows: in Section 2, we introduce basic definitions about infinite games. In Section 3, we introduce RR games, define waiting times and the induced quality measure and prove some preliminary results. In Section 4, we show that for every strategy of small value there is a strategy of smaller or equal value that additionally bounds the waiting times by some doubly-exponential bound. To this end, we give a quantitive version of Dickson's Lemma in Subsection 4.1 and use this to obtain an upper bound in Subsection 4.2. Using this upper bound, we are able to construct a mean-payoff game whose optimal strategy induces an optimal strategy for the RR game. This reduction is presented in Section 5. We conclude in Section 6 with a discussion and some open questions.

The present paper is a revised version with simplified proofs of results announced in the conference paper [14], which in turn extended results of the third author's dissertation [16].

2 Definitions

We denote the set of non-negative integers by $\mathbb{N}$. For every $k \in \mathbb{N}$ we define $[k] = \{1, \ldots, k\}$, so in particular $[0] = \emptyset$. The power set of a set S is denoted by 2^S. The last letter of a finite non-empty word w is denoted by $\mathrm{Last}(w)$.

An arena $\mathcal{A} = (V, V_0, V_1, E)$ consists of a finite, directed graph (V, E), $V_0 \subseteq V$ and $V_1 = V \setminus V_0$, where V_i denotes the vertices of Player i. In examples, we denote the vertices of Player 0 by circles and the vertices of Player 1 by squares. We require every vertex to have an outgoing edge to avoid having to deal with finite plays. The size $|\mathcal{A}|$ of $\mathcal{A}$ is the cardinality of V. A play in $\mathcal{A}$ starting in

$v \in V$ is an infinite sequence $\rho = \rho_0 \rho_1 \rho_2 \cdots$ with $\rho_0 = v$ and $(\rho_n, \rho_{n+1}) \in E$ for all $n \in \mathbb{N}$.

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, which is often defined implicitly. The set of winning plays for Player 1 is $V^\omega \setminus \mathrm{Win}$.

A strategy for Player i is a mapping $\sigma \colon V^* V_i \to V$ 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 $wv \in V^* V_i$. A play $\rho_0 \rho_1 \rho_2 \cdots$ is consistent with σ if $\rho_{n+1} = \sigma(\rho_0 \cdots \rho_n)$ for every n with $\rho_n \in V_i$. Given a set $W \subseteq V$, we denote by $\mathrm{Beh}(W, \sigma)$ the set of plays that start in W and are consistent with σ. A strategy σ for Player i is a winning strategy from W if every play in $\mathrm{Beh}(W, \sigma)$ is winning for Player i. The winning region $W_i(\mathcal{G})$ of Player i in $\mathcal{G}$ contains all vertices from which Player i has a winning strategy. We always have $W_0(\mathcal{G}) \cap W_1(\mathcal{G}) = \emptyset$ and $\mathcal{G}$ is determined if $W_0(\mathcal{G}) \cup W_1(\mathcal{G}) = V$. A winning strategy for Player i is uniform, if it is winning from $W_i(\mathcal{G})$.

A memory structure $\mathfrak{M} = (M, \mathrm{Init}, \mathrm{Upd})$ 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 $\mathrm{Upd}^* \colon V^+ \to M$ by defining $\mathrm{Upd}^*(\rho_0) = \mathrm{Init}(\rho_0)$ and $\mathrm{Upd}^*(\rho_0 \cdots \rho_n \rho_{n+1}) = \mathrm{Upd}(\mathrm{Upd}^*(\rho_0 \cdots \rho_n), \rho_{n+1})$. 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 $m \in M$. The next-move function induces a strategy σ for Player i with memory $\mathfrak{M}$ via the definition $\sigma(\rho_0 \cdots \rho_n) = \mathrm{Nxt}(\rho_n, \mathrm{Upd}^*(\rho_0 \cdots \rho_n))$. The size of $\mathfrak{M}$ (and, slightly abusive, σ) is $|M|$. A strategy σ is finite-state if it can be implemented with a memory structure.

An arena $\mathcal{A} = (V, V_0, V_1, E)$ and a memory structure $\mathfrak{M} = (M, \mathrm{Init}, \mathrm{Upd})$ for $\mathcal{A}$ induce the expanded arena $\mathcal{A} \times \mathfrak{M} = (V \times M, V_0 \times M, V_1 \times M, E')$ where we have $((v, m), (v', m')) \in E'$ if and only if $(v, v') \in E$ and $\mathrm{Upd}(m, v') = m'$. Furthermore, every play $\rho = \rho_0 \rho_1 \rho_2 \cdots$ in the original arena $\mathcal{A}$ has a unique extended play $\mathrm{ext}(\rho) = (\rho_0, m_0)(\rho_1, m_1)(\rho_2, m_2) \cdots$ in $\mathcal{A} \times \mathfrak{M}$ defined by $m_0 = \mathrm{Init}(\rho_0)$ and $m_{n+1} = \mathrm{Upd}(m_n, \rho_{n+1})$, i.e., we have $m_n = \mathrm{Upd}^*(\rho_0 \cdots \rho_n)$. Dually, every play $\rho = (\rho_0, m_0)(\rho_1, m_1)(\rho_2, m_2) \cdots$ in $\mathcal{A} \times \mathfrak{M}$ has a projected play $\mathrm{proj}(\rho) = \rho_0 \rho_1 \rho_2 \cdots$ in $\mathcal{A}$. Note that we have $\mathrm{proj}(\mathrm{ext}(\rho)) = \rho$, but $\mathrm{ext}(\mathrm{proj}(\rho')) = \rho'$ is only true if ρ' starts in a vertex of the form $(v, \mathrm{Init}(v))$.

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

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

So in particular, if a player has a uniform positional winning strategy for $\mathcal{G}'$, then she has a uniform finite-state winning strategy with memory $\mathfrak{M}$ for $\mathcal{G}$.

3 Request-Response Games

A request-response game (RR game for short) is denoted by $(\mathcal{A}, (Q_j, P_j)_{j \in [k]})$ where $\mathcal{A}$ is an arena and Q_j and P_j are subsets of the set of $\mathcal{A}$'s vertices. A vertex in Q_j is referred to as a request of the j-th condition, while a vertex in P_j is a response for the j-th condition. Intuitively, Player 0's goal is to answer every request by a later visit to a corresponding response. Formally, a play $\rho_0 \rho_1 \rho_2 \cdots$ is winning for Player 0, if for every $j \in [k]$ and every n, if $\rho_n \in Q_j$, then there exists an $n' \geq n$ such that $\rho_{n'} \in P_j$. We say that a request of condition j is open after a play prefix w, if w contains a vertex in Q_j that is not followed by a vertex in P_j.

Example 1. *Consider the RR game in Figure 1. At vertex q Player 1 can request either condition 1 and/or condition 2, while at vertex p, Player 0 can either answer condition 1 or condition 2 or none of them. Alternatingly answering condition 1 and 2 is a uniform winning strategy for Player 0 from every vertex.*
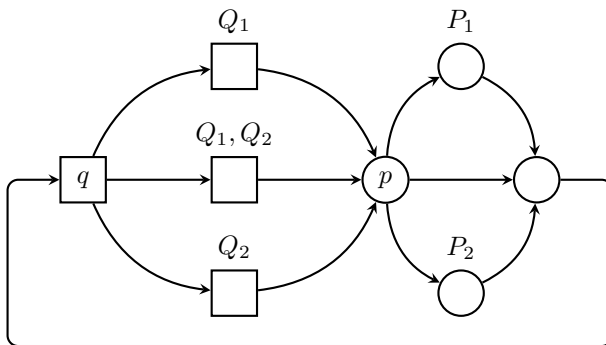

Figure 1: The RR game for Example 1 and Example 2.

There is an intuitive notion of the waiting time between a request and its (earliest) response, which we formalize in the following. The waiting times are then aggregated to measure the quality of a play and the quality of a strategy (both from Player 0's point of view). First, we define the waiting time for condition j, denoted by $\mathrm{wt}_j \colon V^* \to \mathbb{N}$, inductively via $\mathrm{wt}_j(\varepsilon) = 0$, and

$$
\mathrm{wt}_j(wv) = \begin{cases} 0 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \notin Q_j \setminus P_j, \\ 1 & \text{if } \mathrm{wt}_j(w) = 0 \text{ and } v \in Q_j \setminus P_j, \\ 0 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \in P_j, \\ \mathrm{wt}_j(w) + 1 & \text{if } \mathrm{wt}_j(w) > 0 \text{ and } v \notin P_j. \end{cases}
$$

Note that while a request of condition j is open, additional requests are ignored, i.e., we are only interested in the waiting time of the earliest request

that is open, but not in the number of requests (of a single condition) that are open. In [18] an extension of RR games is investigated, where the waiting times take the number of open requests into account as well.

Remark 1. *If* $\mathrm{wt}_j(x) \leq \mathrm{wt}_j(y)$*, then* $\mathrm{wt}_j(xz) \leq \mathrm{wt}_j(yz)$ *for every* $z \in V^*$.

We summarize the waiting times of a play prefix w in its waiting time vector $\overline{\mathrm{wt}}(w) = (\mathrm{wt}_1(w), \ldots, \mathrm{wt}_k(w)) \in \mathbb{N}^k$ and compare such vectors componentwise, i.e., $\overline{\mathrm{wt}}(x) \leq \overline{\mathrm{wt}}(y)$ if $\mathrm{wt}_j(x) \leq \mathrm{wt}_j(y)$ for every j.

We say that a strategy σ for Player 0 (uniformly) bounds the waiting times for condition j by $b \in \mathbb{N}$, if every play prefix w that starts in $W_0(\mathcal{G})$ and is consistent with σ satisfies $\mathrm{wt}_j(w) \leq b$. If σ bounds the waiting times for every condition, then it is a uniform winning strategy.

Now, we use the waiting times to define the quality of plays and strategies from Player 0's point of view: for every j we fix a strictly increasing penalty function $f_j \colon \mathbb{N} \to \mathbb{N}$ (which implies that f_j is unbounded) and define the penalty of a play prefix w for the j-th condition by $p_j(w) = f_j(\mathrm{wt}_j(w))$ and the overall penalty of w by $p(w) = \sum_{j \in [k]} p_j(w)$. We aggregate the penalties of an infinite play ρ to the value of this play by taking the limit superior of the mean accumulated penalties, i.e., we define

$$\mathrm{val}(\rho) = \limsup_{n \to \infty} \frac{1}{n} \sum_{\ell=0}^{n-1} p(\rho_0 \cdots \rho_\ell) \ .$$

Finally, the value of a strategy σ from a vertex v is

$$\mathrm{val}(\sigma, v) = \sup_{\rho \in \mathrm{Beh}(v,\sigma)} \mathrm{val}(\rho) \ .$$

Note that we do not parameterize the functions p_j, p, and val with the penalty functions f_j, although they depend on them. This is done to improve readability. In the following, we will always ensure that the penalty functions are clear from the context.

Example 2. *Using the identity function as penalty functions* f_j*, the uniform winning strategy described in Example 1 has value* $\frac{56}{10}$ *from every vertex, which is witnessed by Player 1 always requesting both conditions every time when at vertex* q*. Every play consistent with this strategy and the alternating-response strategy for Player 0 ends up in a loop of length* 10*, in which the sum of the waiting times (which are also the penalties) is 56. The value of this play is equal to the length of the loop divided by its length, hence* $\frac{56}{10}$*. Every other play has a smaller or equal value. Thus, the value of the strategy is also equal to* $\frac{56}{10}$*, independently of the initial vertex.*

It is important to note that we still consider the game as a zero-sum one; we just associate values to plays and strategies and are interested in optimal strategies, i.e., a winning strategy σ such that every other winning strategy σ' satisfies $\mathrm{val}(\sigma', v) \geq \mathrm{val}(\sigma, v)$ for every vertex v. Note that it is a priori not even clear whether an optimal strategy exists.

The sum of penalties $\sum_{\ell=n}^{n+n'} p(\rho_0 \cdots \rho_\ell)$ for a play infix $\rho_n \cdots \rho_{n+n'}$ with an open request grows (at least) quadratically in n', since the penalty functions f_j are strictly increasing. Our result on the existence of optimal finite-state strategies relies on this growth, as evidenced by the following example, which shows that optimal finite-state strategies do not necessarily exist if we allow constant penalty functions.

Example 3. *Assume we use constant penalty functions (e.g., $f_j(0) = 0$ and $f_j(n) = 1$ for every $n > 0$) to measure the quality of plays and consider the RR game depicted in Figure 2. Player 0 wins from every vertex by traversing both loops infinitely often, which is also necessary to win.*
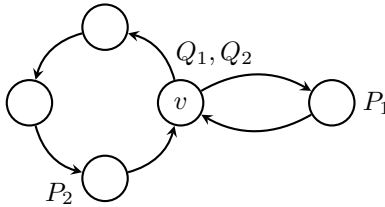

Figure 2: The RR game for Example 3.

In the following, we only consider plays starting in v. As it is Player 0's turn at every vertex, we can identify strategies and plays (and their values are equal).

If a strategy is finite-state, then its play is ultimately periodic, i.e., of the form $\rho = \rho_0 \cdots \rho_{m-1}(\rho_m \cdots \rho_n)^\omega$, where we assume w.l.o.g. $\rho_m = v$. Then, $\mathrm{val}(\rho)$ is equal to the number of positions of the period $\rho_m \cdots \rho_n$ where condition 1 is open plus the number of positions of $\rho_m \cdots \rho_n$ where condition 1 is open.

Now, consider the play infix $\rho'_m \cdots \rho'_n$ obtained from the period by replacing every visit to the left loop by two visits to the right loop. The value of the play $\rho' = \rho_0 \cdots \rho_{m-1}(\rho_m \cdots \rho_n \rho'_m \cdots \rho'_n)^\omega$ is strictly smaller than the value of ρ, since visiting the left loop is more costly than visiting the right one twice.

As ρ' can also be generated by a finite-state strategy, we have shown that there is no optimal finite-state strategy when considering constant penalty functions.

The values $\mathrm{val}(\rho)$ and $\mathrm{val}(\sigma, v)$ measure the quality of a play and a strategy from Player 0's point of view. However, it is not true that a play (a strategy) is winning for Player 0 if, and only if, it has a finite value. One direction holds, as claimed in the next lemma, while the other one can shown to be false by considering a play in which Player 0 allows herself more and more time to answer the requests.

Lemma 2. *Let v be a vertex, let ρ be a play, and let σ be a strategy for Player 0.*

  1. If $\mathrm{val}(\rho) < \infty$, then ρ is a winning play for Player 0.

2. If $\mathrm{val}(\sigma, v) < \infty$*, then* σ *is a winning strategy for Player 0 from* v*.*

3. If $v \in W_1(\mathcal{G})$*, then* $\mathrm{val}(\sigma, v) = \infty$*.*

Proof. (1) Consider the contraposition: let $\rho = \rho_0\rho_1\rho_2\cdots$ be winning for Player 1. Then, some condition j is requested at some position n, but never answered afterwards. Thus, $p_j(\rho_0 \cdots \rho_{n+n'}) \geq f_j(n') \geq n'$ for every n' (recall that f_j is strictly increasing) and therefore

$$
\frac{1}{n+n'} \sum_{\ell=0}^{n+n'-1} p(\rho_0 \cdots \rho_\ell) \geq \frac{1}{n+n'}\frac{n'(n'-1)}{2} = \frac{n'-1}{2\left(\frac{n}{n'}+1\right)}
$$

for all n', which diverges to infinity when n' tends to infinity. Thus, $\mathrm{val}(\rho) = \infty$.

(2) Again, we consider the contraposition: let σ not be a winning strategy from v. Then, there exists a play $\rho \in \mathrm{Beh}(v, \sigma)$ that is winning for Player 1. Thus, $\mathrm{val}(\rho) = \infty$ as we have just shown and $\mathrm{val}(\sigma, v) = \infty$, too.

(3) Let τ be a winning strategy for Player 1 from v and consider the unique play ρ that starts in v and is consistent with σ and τ. We have $\mathrm{val}(\rho) = \infty$, as τ is a winning strategy for Player 1, and therefore $\mathrm{val}(\sigma, v) = \infty$, as ρ is also consistent with σ and starts in v. $\square$

To conclude this introductory section on RR games, we recall the proof of finite-state determinacy of RR games, which proceeds by a reduction to Büchi games. The strategy obtained by this reduction yields a first upper bound on the value of an optimal strategy in an RR game.

The winning condition of a Büchi game is a set F of vertices and Player 0 wins a play if it visits F infinitely often. Alternatively, one can define a Büchi game to be an RR game with a single condition of the form $(V \setminus F, F)$ which is satisfied if and only if F is visited infinitely often. As Büchi games are positionally determined, such a reduction suffices to prove the following result.

Theorem 1 ([17])**.** *RR games are determined with finite-state strategies of size* $k2^{k+1}$*, where* k *denotes the number of RR conditions.*

Proof. Let $\mathcal{G} = (\mathcal{A}, (Q_j, P_j)_{j \in [k]})$ be an RR game with (w.l.o.g.) $k > 1$ conditions. Consider the memory structure $\mathfrak{M} = (M, \mathrm{Init}, \mathrm{Upd})$ with $M = 2^{[k]} \times [k] \times \{0, 1\}$, $\mathrm{Init}(v) = (\{j \mid v \in Q_j \setminus P_j\}, 1, 0)$, and $\mathrm{Upd}(R, c, f) = (R', c', f')$ where

- $R' = (R \cup \{j \mid v \in Q_j\}) \setminus \{j \mid v \in P_j\}$,

- $c' = c$ if $c \in R \cap R'$, and $c' = (c \mod k) + 1$ otherwise, and

- $f = 1$ if $c' \neq c$, and $f = 0$, otherwise.

So, R keeps track of open requests, c is a cyclic counter over $[k]$ that is incremented every time its current value is not an open request, and the flag f is equal to 1 if and only if c has changed its value. So, there is an unanswered request if and only if f is equal to 0 from some point onwards. Thus, consider the set $F = V \times (2^{[k]} \times [k] \times \{1\})$. Then, we have $\mathcal{G} \leq_{\mathfrak{M}} (\mathcal{A} \times \mathfrak{M}, F)$, i.e., the result follows from Lemma 1. $\square$

The upper bound on the memory requirements was slightly lowered and (almost) matching lower bounds were proven in [7]. However, for our purposes, the simple bound presented here is sufficient to obtain an upper bound on the value of an optimal strategy.

Corollary 1. *In every RR game $\mathcal{G}$, Player 0 has a winning strategy σ with $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} f_j(sk2^k)$ for every $v \in W_0(\mathcal{G})$, where s denotes the size of the arena and k the number of RR conditions.*

Proof. Let $\mathfrak{M}$ be the memory structure defined in the proof of Lemma 1 and let σ be a uniform winning strategy σ for $\mathcal{G}$ with memory $\mathfrak{M}$. We show that σ has the desired properties. To this end, we prove that σ bounds the waiting time of every condition by $sk2^k$. Then, we have

$$\mathrm{val}(\rho) = \limsup_{n \to \infty} \frac{1}{n} \sum_{\ell=0}^{n-1} \sum_{j \in [k]} f_j(\mathrm{wt}_j(\rho_0 \cdots \rho_\ell))$$

$$\leq \limsup_{n \to \infty} \frac{1}{n} \sum_{\ell=0}^{n-1} \sum_{j \in [k]} f_j(sk2^k) \leq \sum_{j \in [k]} f_j(sk2^k) \ .$$

for all ρ starting in $W_0(\mathcal{G})$ that are consistent with σ, which implies our claim.

Towards a contradiction, assume we have $\mathrm{wt}_j(\rho_0 \cdots \rho_n) > sk2^k$. Then, j is in the first component (which keeps track of open requests) of the last $sk2^k + 1$ memory states reached during $\rho_0 \cdots \rho_n$. But there are only $k2^k$ memory states that contain j in the first component. Hence, there are positions $m < m'$ in this interval such that $\rho_m = \rho_{m'}$ and $\mathrm{Upd}^*(\rho_0 \cdots \rho_m) = \mathrm{Upd}^*(\rho_0 \cdots \rho_{m'})$.

Now, consider the play $\rho' = \rho_0 \cdots \rho_{m-1}(\rho_m \cdots \rho_{m'-1})^\omega$ obtained by repeating the loop between positions m and m', which is also in $\mathrm{Beh}(W_0(\mathcal{G}), \sigma)$. But ρ' contains an unanswered request, since condition j is open at ρ_m and never answered during $\rho_m \cdots \rho_{m'-1}$. This contradicts the fact that σ is winning from $W_0(\mathcal{G})$. $\qquad\square$

The exponential upper bound on the waiting times used in the proof of Corollary 1 gives a correction to a claim of [14, 18] where the bound sk is used. Next, we give an example showing a matching exponential lower bound.

Example 4. *Consider the RR game depicted in Figure 3. We consider plays starting at vertex i, where all four RR conditions are requested. From there, the only move leads to the hub vertex h, where Player 0 has to move into one of the four blades, each one of them associated with one of the RR conditions. The first vertex of the blade for condition $j \in [4]$ (c_j in the figure) is in P_j, i.e., condition j is responded to. From this vertex, Player 1 can either move to a sink vertex (called s_j) where every condition with index larger than j is answered, too, or he can move to a vertex (called v_j) where all conditions with index smaller than j are requested again. From this vertex, the only move leads back to the hub. Due to the existence of the former move, Player 1 can win*
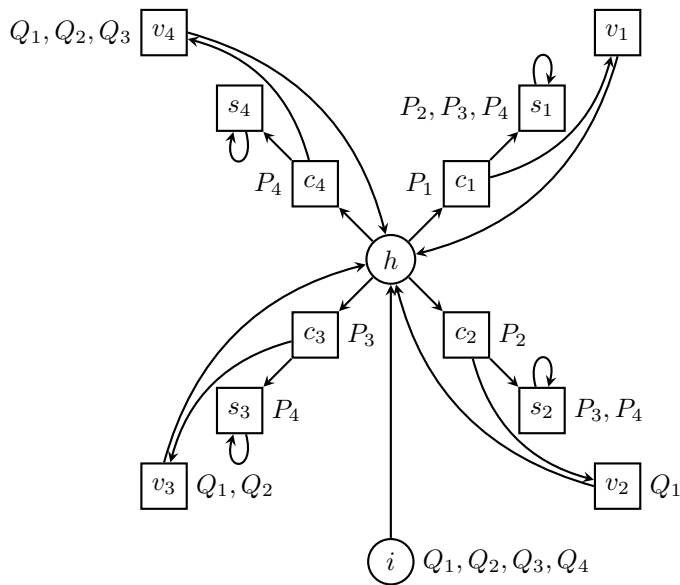
Figure 3: An RR game with exponential waiting times.

if the j-th blade is entered while a request of a condition with smaller index is open.

Player 0 has a winning strategy for this RR game from vertex i by always moving to the blade of the smallest open condition. This strategy takes $2^4 - 1$ visits to the hub to answer the request of condition 4, since every smaller condition is requested after answering the smallest open condition. Once all requests are answered, Player 0 can always move to the first blade, which does not generate new requests. Deviating from this strategy either generates additional requests (if moving to a blade of a condition with smaller index than the currently smallest open one) and thereby prolongs the time it takes to answer condition 4, or allows Player 1 to move to a sink vertex where he wins (if moving to a blade of a condition with larger index than the currently smallest open one). Thus, every winning strategy from vertex i for Player 0 in this game takes at least $2^4 - 1$ visits to the hub before condition 4 is answered.

This game can be generalized by having k conditions and k blades. Then, it takes at least $2^k - 1$ visits to the hub to answer the request of condition k. Thus, the waiting time for condition k is larger than 2^k before it is eventually answered.

4   Bounding the Waiting Times in RR Games

In this section, we show that for every strategy whose value is small from every vertex in $W_0(\mathcal{G})$ there is a strategy with smaller or equal values that additionally bounds the waiting times by some bound b, which only depends on the size of the arena and the number of RR conditions. This restricts the search space for optimal strategies to a finite one (in terms of possible waiting time vectors). In the next section, this space is turned into an arena for a mean-payoff game. Intuitively, the arena tracks plays of the RR game and their waiting times up to the threshold b. The value of a play in the mean-payoff game is the value of the tracked play in the original game. Thus, an optimal strategy for Player 0 in the mean-payoff game (which can be effectively computed) can be turned into an optimal winning strategy for Player 0 in the RR game.

We proceed as follows: in Subsection 4.1, we derive the bound b and in Subsection 4.2 we show that we can turn every strategy of small value into a strategy of smaller or equal value whose waiting times are bounded by b.

4.1   Dickson's Lemma for Waiting Times

Given a strategy σ with small values for $W_0(\mathcal{G})$ we need to construct a strategy with smaller or equal values that also bounds the waiting times by a constant that only depends on the number of vertices and RR conditions and the penalty functions. We achieve this by removing loops from plays in which the waiting time is high for some condition. However, this might have an effect on the waiting times for other conditions as well: in the worst case we might remove an answer to a request, thereby increasing the waiting time or even generating a losing play. To avoid this, we only remove a loop if the waiting time vector at the end of the loop is larger than at the beginning.

This removal process is iterated until ad infinitum, i.e., in the limit there are no more such loops. Hence, our bound b has to be an upper bound on the length of play infixes without such a loop. We derive b in this subsection by giving a quantitative version of Dickson's Lemma [10], which states that there is no infinite play prefix without such a loop. However, the lemma does not give an explicit bound on the length of a play without such a loop. Indeed, if we allow arbitrary vectors of natural numbers (this is the setting of Dickson's Lemma), there are arbitrarily long sequences. But by exploiting the simple update-rule of the waiting times – increment or reset – we are able to obtain a doubly-exponential bound b.

Let ρ be a play of the RR game $\mathcal{G}$. We say that a pair of positions (n_1, n_2) of ρ with $n_1 < n_2$ is dickson, if we have $\rho_{n_1} = \rho_{n_2}$ and $\overline{\mathrm{wt}}(\rho_0 \cdots \rho_{n_1}) \leq \overline{\mathrm{wt}}(\rho_0 \cdots \rho_{n_2})$. Note that the notion is defined with respect to the whole play prefix $\rho_0 \cdots \rho_{n_2}$, since the waiting times are computed starting at the first position of ρ. An infix is dickson, if it contains a dickson pair of positions, otherwise it is non-dickson.

The goal of this section is to define a function $b \colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such the following is true for every RR game with s vertices and k RR conditions: every play infix of length at least $b(s, k)$ has a dickson pair. Note that this implies

that we have to deal with arbitrarily high waiting times at the beginning of the infix.

We define b by induction over k, the number of RR conditions. For $k = 0$, we have $b(s, k) = s + 1$, since every state repetition yields a dickson pair. Now, consider a game with $k > 0$ RR conditions. We begin by stating a lemma that restricts the combinations of values that can appear in waiting time vectors in a non-dickson infix: intuitively, not too many waiting times can be large at the same time, since this would imply the existence of a dickson pair in the remaining conditions. This is also a dickson pair for all conditions, since the large values only increase between these two positions.

Lemma 3. *Let* $\rho_m \cdots \rho_{m+\ell}$ *be a non-dickson play infix of a play* ρ *in an RR game with* s *vertices and* k *RR conditions. For every* j *in the range* $0 \le j \le k-1$ *and every* n *in the range* $b(s, k - (j+1)) \le n \le \ell$, $\overline{\mathrm{wt}}(\rho_0 \cdots \rho_{m+n})$ *contains at most* j *entries that are larger than* $b(s, k - (j+1))$.

Proof. Towards a contradiction, assume there is a j such that $\overline{\mathrm{wt}}(\rho_0 \cdots \rho_{m+n})$ contains $j+1$ entries that are larger than $b(s, k-(j+1))$, where $b(s, k-(j+1)) \le n \le m+\ell$. We denote the set of coordinates of these entries in the waiting time vectors by J, i.e., $J \subseteq [k]$. The entries at the coordinates in J are updated by increasing them during the last $b(s, k - (j + 1))$ positions before $m + n$, which are all positions contained in the infix.

Now, consider the projection to the $k - (j+1)$ coordinates not in J: there is a dickson pair in the infix $\rho_{m+n+1-b(s,k-(j+1))} \cdots \rho_{m+n}$, as it has length $b(s, k - (j + 1))$. This is also a dickson pair when considering all coordinates, since the values at the coordinates in J are strictly increasing during this infix. This yields the desired contradiction. $\qquad\square$

Due to the previous lemma, after $b(s, k - 1)$ positions in a non-dickson infix, every vector has no entry larger than $b(s, k - 1)$, at most one entry larger than $b(s, k - 2)$, at most two entries larger than $b(s, k - 3)$, and in general, at most j entries larger than $d(s, k - (j + 1))$ for every $j \in \{0, \ldots, k - 1\}$. Rephrasing this, we obtain that every such vector contains an entry smaller than $b(s, 0)$, another entry smaller than $b(s, 1)$, another entry smaller than $b(s, 2)$, and so on. The number of such vectors is bounded by $k! \prod_{j=0}^{k-1} b(s, j)$. So, we can define for $k > 0$

$$b(s, k) = \left(b(s, k - 1) + s k! \prod_{j=0}^{k-1} b(s, j) \right) + 1 \ .$$

The first summand is due to the fact that the bounds only hold after $b(s, k - 1)$ steps, and the factor s in the second summand takes account of the fact that we need a state repetition in a dickson-pair as well.

Lemma 4. *Let* $\mathcal{G}$ *be an RR game with* s *vertices and* k *RR conditions.*

1. *Every play infix of length* $b(s, k)$ *has a dickson pair.*

2. We have $b(s, k) \in \mathcal{O}(2^{2^{s \cdot k + 2}})$.

Proof. (1) This follows directly from Lemma 4.1 and from the arguments presented after it.

(2) We show $b(s, k) \le 2^{2^{k-1}}(s+1)^{2^k} k! \prod_{j=1}^{k-1}(j!)^{2^{k-(j+1)}}$ for every $k > 0$, which implies the claim. Throughout the proof we use the following upper bound

$$b(s, k) = \left(b(s, k-1) + sk! \prod_{j=0}^{k-1} b(s, j) \right) + 1 \le 2(s+1)k! \prod_{j=0}^{k-1} b(s, j)$$

for $k > 0$, which also proves our claim for $k = 1$. Now, consider $k > 1$. We have

$$b(s, k) \le 2(s+1)k! \prod_{j=0}^{k-1} b(s, j)$$

$$= 2(s+1)^2 k! \prod_{j=1}^{k-1} b(s, j)$$

$$\le 2(s+1)^2 k! \prod_{j=1}^{k-1} \left(2^{2^{j-1}}(s+1)^{2^j} j! \left(\prod_{j'=1}^{j-1} (j'!)^{2^{j-(j'+1)}} \right) \right)$$

$$= 2^{1+\sum_{j=1}^{k-1} 2^{j-1}} \cdot (s+1)^{2+\sum_{j=1}^{k-1} 2^j} \cdot k! \prod_{j=1}^{k-1} j! \prod_{j'=1}^{j-1} (j'!)^{2^{j-(j'+1)}}$$

$$= 2^{1+\sum_{j=0}^{k-2} 2^j} \cdot (s+1)^{1+\sum_{j=0}^{k-1} 2^j} \cdot k! \prod_{j=1}^{k-1} (j!)^{2^{k-(j+1)}}$$

$$= 2^{2^{k-1}} \cdot (s+1)^{2^k} \cdot k! \prod_{j=1}^{k-1} (j!)^{2^{k-(j+1)}} \ .$$

The equality $\prod_{j=1}^{k-1} j! \prod_{j'=1}^{j-1} (j'!)^{2^{j-(j'+1)}} = \prod_{j=1}^{k-1} (j!)^{2^{k-(j+1)}}$ used in the second-to-last equality can be shown by a straightforward induction. $\square$

Recently, Czerwiński et al. complemented our doubly-exponential upper bound by proving a doubly-exponential lower bound of $2^{2^{k/2}}$ [9].

4.2 Strategies with Small Values and Bounded Waiting Times

In this subsection, we show how to turn a strategy with small values from every vertex in $W_0(\mathcal{G})$ into a strategy of smaller or equal values whose waiting times are bounded. To this end, we remove loops of plays in which the waiting time for some condition j is large. By doing this ad infinitum, we obtain a limit strategy with the desired properties.

Throughout this subsection, we fix an RR game $\mathcal{G} = (\mathcal{A}, (Q_j, P_j)_{j \in [k]})$ with $\mathcal{A} = (V, V_0, V_1, E)$ and $|V| = s$ as well as a penalty function f_j for every condition j. The goal of this section is to prove the following lemma, which shows that for every strategy of small value there is a strategy of smaller or equal value that additionally bounds the waiting times. In particular, the result applies to the uniform winning strategy for Player 0 from Corollary 1 which satisfies

$$\text{val}(\sigma, v) \leq \sum_{j \in [k]} f_j(sk2^k) =: \text{val}_{\mathcal{G}}$$

for every $v \in W_0(\mathcal{G})$.

Lemma 5. *Let σ be a strategy such that $\text{val}(\sigma, v) \leq \text{val}_{\mathcal{G}}$ for every $v \in W_0(\mathcal{G})$. There is a strategy σ' with $\text{val}(\sigma', v) \leq \text{val}(\sigma, v)$ for every $v \in V$ that uniformly bounds the waiting times for every condition j by $f_j^{-1}(\text{val}_{\mathcal{G}}) + b(s, k-1)$.*

Note that $\text{val}(\sigma', v) \leq \text{val}(\sigma, v) \leq \text{val}_{\mathcal{G}}$ for every $v \in W_0(\mathcal{G})$ implies that σ and σ' are uniform winning strategies.

In this subsection, it is convenient to view a strategy as the set of play prefixes that are consistent with it. This representation simplifies the process of removing loops from the plays which are consistent with the strategy. Also, we only consider plays starting in $W_0(\mathcal{G})$ since we need to bound the waiting times for such plays, the waiting times for plays starting in $W_1(\mathcal{G})$ cannot be bounded and are ignored.

Formally, a strategy tree is a prefix-closed language $t \subseteq V^*$ such that the following conditions are satisfied:

1. For every $w_0 \cdots w_n \in t$ we have $w_0 \in W_0(\mathcal{G})$ and $(w_{n'}, w_{n'+1}) \in E$ for every $n' < n$ (only play prefixes starting in $W_0(\mathcal{G})$ are in t).

2. $t \cap V = W_0(\mathcal{G})$ (every initial vertex from $W_0(\mathcal{G})$ is in t, but no initial vertex from $W_1(\mathcal{G})$).

3. For every $w_0 \cdots w_n \in t$ with $w_n \in V_0$ there is a unique $v \in V$ such that $w_0 \cdots w_n v \in t$ (there is a unique successor in t for play prefixes ending in V_0).

4. For every $w_0 \cdots w_n \in t$ with $w_n \in V_1$ and every successor v of w_n in $\mathcal{A}$ we have $w_0 \cdots w_n v \in t$ (all successors are in t for play prefixes ending in V_1).

Every strategy σ can be turned into a strategy tree $t(\sigma)$ containing exactly the prefixes (including the empty prefix ε) of plays that start in $W_0(\mathcal{G})$ and are consistent with σ. Vice versa, every strategy tree t defines a strategy $\sigma(t)$ mapping w ending in V_0 to the unique vertex v with $wv \in t$. Note that this strategy is only defined for play prefixes starting in Player 0's winning region that are consistent with σ. However, this is sufficient for our purposes, since $\sigma(t)(w)$ can be defined arbitrarily for every other w.

Given a strategy tree t and $w \in t$, define

$$S_j(t, w) = \{ww' \in t \mid w' \in (V \setminus P_j)^*\}$$

14

to be the set of continuations of w in which no vertex from P_j is visited, i.e., condition j is not responded to.

Remark 2. *Let t be a strategy tree such that $\sigma(t)$ is a uniform winning strategy, and let $w \in t$ with $\mathrm{wt}_j(w) > 0$. Then, $S_j(t, w)$ is finite.*

Proof. Assume $S_j(t, w)$ is infinite. Then, König's Lemma implies the existence of an infinite play $w\rho$ whose prefixes are all in t, in which a request of condition j is open after w (as the waiting time is non-zero), but ρ contains no answer. As such a play is consistent with $\sigma(t)$ this contradicts the fact that $\sigma(t)$ is a winning strategy. $\square$

We now formalize the removal of loops, which turns a strategy tree t into a new one denoted by t'. Fix some condition $j \in [k]$ for which we want to remove loops with large waiting times and fix a strategy tree t such that $\mathrm{val}(\sigma(t), v) \leq \mathrm{val}_{\mathcal{G}}$ for every $v \in W_0(\mathcal{G})$. Next, we define the tree $t' \subseteq V^*$ and a mapping $h \colon t' \to t$ satisfying $\mathrm{Last}(h(w)) = \mathrm{Last}(w)$ for every $w \in t' \setminus \{\varepsilon\}$.

We have $\varepsilon \in t'$ and $W_0(\mathcal{G}) \subseteq t'$ and define $h(\varepsilon) = \varepsilon$ and $h(v) = v$ for every $v \in W_0(\mathcal{G})$. Now, consider some $w \in t'$: we have $h(w) \in t$ with $\mathrm{Last}(h(w)) = \mathrm{Last}(w)$. For every $h(w)v \in t$ we add wv to t' and it remains to define $h(wv)$. Here, we consider two cases:

1. If $\mathrm{wt}_j(h(w)v) \leq f_j^{-1}(\mathrm{val}_{\mathcal{G}})$, then $h(wv) = h(w)v$.

2. If $\mathrm{wt}_j(h(w)v) > f_j^{-1}(\mathrm{val}_{\mathcal{G}})$, then consider the set $S_j(t, h(w)v)$. As it is finite we can pick a longest element[1] x from $S_j(t, h(w)v)$ satisfying $\overline{\mathrm{wt}}(h(w)v) \leq \overline{\mathrm{wt}}(x)$ and $\mathrm{Last}(x) = v$. Such an element always exists, since $h(w)v \in S_j(t, h(w)v)$ satisfies both requirements. We define $h(wv) = x$.

Note that the property $\mathrm{Last}(h(w)) = \mathrm{Last}(w)$ is satisfied in both cases of the definition. We begin by listing some straightforward properties of the function h we use to show that t' is also a strategy tree.

Remark 3. *Let $w = w_0 \cdots w_n \in t'$.*

1. *$h(w) = w_0 s_0 w_1 s_1 \cdots s_{n-1} w_n$ for some $s_0, \ldots, s_{n-1} \in V^*$.*

2. *$h(w_0 \cdots w_{n'})$ is a proper prefix of $h(w_0 \cdots w_n)$ for every $n' < n$.*

3. *h is injective.*

4. *If $h(w) = w$, then $h(w_0 \cdots w_{n'}) = w_0 \cdots w_{n'}$ for every $n' < n$.*

5. *Let $w' \in t$. If there is no $w \in t'$ with $h(w) = w'$ then $\mathrm{wt}_j(w') > f_j^{-1}(\mathrm{val}_{\mathcal{G}})$.*

Now, we prove that t' is a strategy tree if t is one, and that transforming t into t' preserves waiting time bounds and does not increase the values of the strategy.

---

[1] Using the lexicographic order w.r.t. some fixed ordering of V to break ties.

Lemma 6. *Let t be a strategy tree such that $\mathrm{val}(\sigma(t), v) \leq \mathrm{val}_{\mathcal{G}}$ for every $v \in W_0(\mathcal{G})$, let t' be constructed as described above, and let $h\colon t' \to t$ be the function defined in the construction.*

1. *t' is a strategy tree.*

2. *$\overline{\mathrm{wt}}(w) \leq \overline{\mathrm{wt}}(h(w))$ for every $w \in t'$.*

3. *If $\sigma(t)$ bounds the waiting times for condition j' by b, then so does $\sigma(t')$.*

4. *$\mathrm{val}(\sigma(t'), v) \leq \mathrm{val}(\sigma(t), v)$ for every $v \in W_0(\mathcal{G})$.*

Proof. (1) Prefix-closure and the first requirement on a strategy tree can be proven by a straightforward induction over the length of $w \in t'$ while the second requirement is satisfied by construction. Now, consider $w_0 \cdots w_n \in t'$ with $w_n \in V_0$. We have $\mathrm{Last}(h(w_0 \cdots w_n)) = w_n$, i.e., there is a unique successor v of w_n with $h(w_0 \cdots w_n)v \in t$. By construction, $w_0 \cdots w_n v$ is added to t', but no $w_0 \cdots w_n v'$ for $v \neq v'$. Hence, there is a unique v such that $w_0 \cdots w_n v \in t'$, i.e., the third requirement is satisfied. The reasoning for the fourth requirement is dual.

(2) By induction over $|w|$. The claim is trivially true for $|w| \leq 1$. Thus, consider $w = w'v \in t$. If $h(w'v) = h(w')v$ then

$$\overline{\mathrm{wt}}(w) = \overline{\mathrm{wt}}(w'v) \leq \overline{\mathrm{wt}}(h(w')v) = \overline{\mathrm{wt}}(h(w'v)) \ ,$$

where the inequality follows from an application of Remark 1 to the induction hypothesis $\overline{\mathrm{wt}}(w') \leq \overline{\mathrm{wt}}(h(w'))$. On the other hand, if $h(w'v) = x$ for some x satisfying $\overline{\mathrm{wt}}(h(w')v) \leq \overline{\mathrm{wt}}(x)$ then

$$\overline{\mathrm{wt}}(w) = \overline{\mathrm{wt}}(w'v) \leq \overline{\mathrm{wt}}(h(w')v) \leq \overline{\mathrm{wt}}(x) = \overline{\mathrm{wt}}(h(w'v)) \ ,$$

where the first inequality again follows from an application of Remark 1 to the induction hypothesis.

(3) This follows directly from Item (2).

(4) Let $\rho = \rho_0 \rho_1 \rho_2 \cdots$ be consistent with $\sigma(t')$ and consider the sequence

$$h(\rho_0), h(\rho_0 \rho_1), h(\rho_0 \rho_1 \rho_2), \ldots$$

of elements from t, which is an increasing chain in the (strict) prefix relation. Hence, the sequence has a unique limit $h(\rho) = \rho'_0 \rho'_1 \rho'_2 \cdots \in V^{\omega}$ such that $h(\rho_0 \cdots \rho_n)$ is a prefix of $h(\rho)$ for every n, which is the play from which ρ is obtained by removing loops. The limit is consistent with $\sigma(t)$ as it is a path through t.

Let

$$R = \{\ell \in \mathbb{N} \mid \text{there is no } n \text{ with } h(\rho_0 \cdots \rho_n) = \rho'_0 \cdots \rho'_{\ell}\}$$

be the positions of vertices of $h(\rho)$ that are removed. Due to Remark 3(5), we have $\mathrm{wt}_j(\rho'_0 \cdots \rho'_{\ell}) > f_j^{-1}(\mathrm{val}_{\mathcal{G}})$ for every $\ell \in R$ and therefore

$$p(\rho'_0 \cdots \rho'_{\ell}) \geq f_j(\mathrm{wt}_j(\rho'_0 \cdots \rho'_{\ell})) > \mathrm{val}_{\mathcal{G}} \geq \mathrm{val}(\sigma(t), \rho'_0) \geq \mathrm{val}(h(\rho)) \ .$$

Thus,

$$\limsup_{n \to \infty} \frac{1}{n} \sum_{\ell=0}^{n-1} p(h(\rho_0 \cdots \rho_\ell)) \leq \mathrm{val}(h(\rho)) \ , \tag{1}$$

since the average (and therefore also the limit superior of the averages) only decreases when we omit summands which are larger than the limit superior of the averages, i.e., those for $\rho_0 \cdots \rho_\ell$ with $\ell \in R$. Furthermore, we have $\overline{\mathrm{wt}}(\rho_0 \cdots \rho_\ell) \leq \overline{\mathrm{wt}}(h(\rho_0 \cdots \rho_\ell))$ for every ℓ due to Item (2), and therefore

$$\frac{1}{n} \sum_{\ell=0}^{n-1} p(\rho_0 \cdots \rho_\ell) \leq \frac{1}{n} \sum_{\ell=0}^{n-1} p(h(\rho_0 \cdots \rho_\ell)) \ .$$

Thus, the value of ρ, the limit superior of the left-hand side of the inequality is smaller or equal to the limit superior of the right-hand side, which in turn is smaller or equal to the value of $h(\rho)$, as shown in Equation (1). Thus, we have $\mathrm{val}(\rho) \leq \mathrm{val}(h(\rho))$.

Now, we can lift this upper bound to the values of the strategies: we have

$$\mathrm{val}(\sigma(t'), v) = \sup_{\rho \in \mathrm{Beh}(v, \sigma(t'))} \mathrm{val}(\rho) \leq \sup_{\rho \in \mathrm{Beh}(v, \sigma(t'))} \mathrm{val}(h(\rho))$$

$$\leq \sup_{\rho \in \mathrm{Beh}(v, \sigma(t))} \mathrm{val}(\rho) = \mathrm{val}(\sigma(t), v) \ ,$$

where the first inequality is the one just proven above and the second one due to the fact that $h(\rho)$ is consistent with $\sigma(t)$. $\qquad\blacksquare$

From now on denote the tree t' by $I_j(t)$ as it is obtained by removing loops w.r.t. condition j from t. Note that we have not claimed that $I_j(t)$ bounds the waiting times for condition j. We will now apply I_j infinitely often and show that the limit of the trees obtained this way does indeed bound the waiting times. Whether applying I_j once suffices to achieve this is an open question.

Formally, given a strategy tree t we define an infinite sequence of trees via $t_0 = t$ and $t_{n+1} = I_j(t_n)$. Furthermore, for $n > 0$ let $h_n \colon t_n \to t_{n-1}$ be the function constructed in the definition of t_n. We define the limit t_ω of the t_n as follows: $w \in t_\omega$ if and only if $h_n(w) = w$ for almost all n, i.e., for all but finitely many n. Note that $h_n(w) = w$ implies $w \in t_n \cap t_{n-1}$.

Lemma 7. *Let t be a strategy tree such that $\mathrm{val}(\sigma(t), v) \leq \mathrm{val}_{\mathcal{G}}$ for every $v \in W_0(\mathcal{G})$ and let t_ω be constructed as described above.*

1. *t_ω is a strategy tree.*

2. *$\sigma(t_\omega)$ bounds the waiting times for condition j to $f_j^{-1}(\mathrm{val}_{\mathcal{G}}) + b(s, k-1)$.*

3. *If $\sigma(t)$ bounds the waiting times for condition j' by b, then so does $\sigma(t_\omega)$.*

4. *$\mathrm{val}(\sigma(t_\omega), v) \leq \mathrm{val}(\sigma(t), v)$ for every $v \in W_0(\mathcal{G})$.*

Proof. (1) Prefix-closure of t_ω follows from Remark 3.(4): let $w \in t_\omega$, i.e., we have $h_n(w) = w$ for almost all n. Then, we have $h_n(w') = w'$ for the same n and all prefixes w' of w. Hence, $w' \in t_\omega$.

Furthermore, w satisfies the first requirement on a strategy tree, since w is in some t_n, for which the first requirement holds due to Lemma 6.(1). Furthermore, the second requirement is satisfied by construction: every t_n contains $W_0(\mathcal{G})$ and we have $h_n(v) = v$ for every $v \in W_0(\mathcal{G})$. Hence, it remains to prove the last two properties.

For the third requirement, consider $w \in t_\omega$ with $\mathrm{Last}(w) \in V_0$. We have to show that there is a unique v with $wv \in t_\omega$. Let n_w be such that $h_n(w) = w$ for every $n \geq n_w$, which implies $w \in t_n$ for every $n \geq n_w - 1$.

As every t_n is a strategy tree, there is a unique v_n with $wv_n \in t_n$ for every $n \geq n_w - 1$. We claim $v_n = v_{n_w-1}$ for every $n \geq n_w - 1$. The induction start $n = n_w - 1$ is trivial, so consider some $n > n_w - 1$: wv_n is in t_n, since $h_n(w)v_n = wv_n$ is in t_{n-1}. Now, this implies $v_n = v_{n-1}$, as v_{n-1} is the unique vertex v with $wv \in t_{n-1}$. An application of the induction hypothesis $v_{n-1} = v_{n_w-1}$ yields the desired result.

From now on we denote v_{n_w-1} by v. We have to show $h_n(wv) = wv$ for almost all n. As a first case, assume we have $\mathrm{wt}_j(wv) \leq f_j^{-1}(\mathrm{val}_\mathcal{G})$. Then, we have $h_n(wv) = wv$ for every $n \geq n_w - 1$, as we are in case (1) of the definition of $h_n(wv)$. Now, assume we have $\mathrm{wt}_j(wv) > f_j^{-1}(\mathrm{val}_\mathcal{G})$. We claim

$$|S_j(t_n, wv)| \geq |S_j(t_{n+1}, wv)|$$

for every $n \geq n_w - 1$. Every element in $S_j(t_{n+1}, wv)$ is mapped by h_{n+1} to an element in $S_j(t_n, wv)$. Hence, finiteness of the sets and injectivity of h_{n+1} proves our claim. Furthermore, we have equality

$$|S_j(t_n, wv)| = |S_j(t_{n+1}, wv)|$$

only in case $h_{n+1}(wv) = wv$: if $h_{n+1}(wv) \neq wv$, then there is no element in $S_j(t_{n+1}, wv)$ that is mapped to $wv \in S_j(t_n, wv)$, due to Remark 3.(1).

Thus, the sequence $(S_j(t_n, wv))_{n \geq n_w-1}$ gets stationary and from that point onwards, we have $h_n(wv) = wv$. Thus, $wv \in t_\omega$. Furthermore, v is unique since $wv' \in t_\omega$ with $v' \neq v$ implies that there is an n with $wv \in t_n$ and $wv' \in t_n$, which contradicts the fact that t_n is a strategy tree.

The fourth and final requirement on t_ω can be proven dually: let $w \in t_\omega$ with $\mathrm{Last}(w) \in V_1$. We have to show that $wv \in t_\omega$ for every successor v of $\mathrm{Last}(w)$. Again, every such wv is in every t_n for $n \geq n_w - 1$. Now, using the same reasoning as for the third requirement, one can show $wv \in t_\omega$ for every v.

(2) Assume there is a $w = w_0 \cdots w_m \in t_\omega$ with $\mathrm{wt}_j(w) > f_j^{-1}(\mathrm{val}_\mathcal{G}) + b(s, k-1)$. Let $w' = w_0 \cdots w_{m-b(s,k-1)}$. Thus, we have $\mathrm{wt}_j(w') > f_j^{-1}(\mathrm{val}_\mathcal{G})$.

Now, consider the infix $w_{m-b(s,k-1)+1} \cdots w_m$ of length $b(s, k-1)$. It contains a dickson pair (m_0, m_1) with $m - b(s, k-1) + 1 \leq m_0 < m_1 \leq m$ by definition of $b(s, k-1)$. Here, the parameter $k - 1$ stems from the fact that the waiting times for condition j increase throughout the infix, i.e., there are only $k - 1$ conditions we have to consider to obtain a dickson pair.

18

Now, consider an index n with $h_n(w) = w$, which implies $h_n(w_0 \cdots w_{m_0}) = w_0 \cdots w_{m_0}$, too. Since we have $\mathrm{wt}_j(w_0 \cdots w_{m_0}) > f_j^{-1}(\mathrm{val}_{\mathcal{G}})$, we are in the second case of the definition of $h_n(w_0 \cdots w_{m_0})$ and the existence of $w_0 \cdots w_{m_1} \in S_j(t_{n-1}, w_0 \cdots w_{m_0})$ with $w_{m_0} = w_{m_1}$ and $\overline{\mathrm{wt}}(w_0 \cdots w_{m_0}) \leq \overline{\mathrm{wt}}(w_0 \cdots w_{m_1})$ implies $h_n(w_0 \cdots w_{m_0}) \neq w_0 \cdots w_{m_0}$, i.e., we have derived the desired contradiction.

(3) For every $w \in t_\omega$ there is an $n_w \in \mathbb{N}$ such that $h_n(w) = w$ for every $n \geq n_w$. Furthermore, due to Remark 3.(4) we can pick the n_w in way that they satisfy $n_{w'} \leq n_w$ for every w, w' such that w' is a prefix of w.

Now, define $h_\omega \colon t_\omega \to t$ via

$$h_\omega(w) = h_1(h_2(\cdots h_{n_w-2}(h_{n_w-1}(w)) \cdots)) \ .$$

Applying Lemma 6.(3) inductively yields $\overline{\mathrm{wt}}(w) \leq \overline{\mathrm{wt}}(h_\omega(w))$ for every $w \in t_\omega$. The result follows.

(4) The proof is analogous to the one for Lemma 6.(4), we just have to replace h by h_ω. $\qquad\square$

We denote the limit t_ω of the applications of I_j to t by $I_{j,\omega}(t)$. Now, we are ready to prove the main result of this subsection.

Proof of Lemma 5. Consider the strategy tree $t' = I_{k,\omega}(\cdots I_{2,\omega}(I_{1,\omega}(t(\sigma))) \cdots)$ and the resulting strategy $\sigma' = \sigma(t')$. An inductive application of Lemma 7 yields that σ' bounds the waiting times for every condition j by $f_j^{-1}(\mathrm{val}_{\mathcal{G}}) + b(s, k-1)$ and satisfies $\mathrm{val}(\sigma', v) \leq \mathrm{val}(\sigma, v)$ for every $v \in V$. $\qquad\square$

The construction presented here gives a correction to the one presented in [14] where each loop removal operator I_j is applied only once.

5 Computing Optimal Strategies for RR Games

In this section, we prove our main result: Player 0 has optimal finite-state winning strategies in RR games, which are effectively computable. To this end, we construct a mean-payoff game in an arena which keeps track of the waiting times up to the bounds $f_j^{-1}(\mathrm{val}_{\mathcal{G}}) + b(s, k-1)$ and whose weight function reflects the penalty functions.

Then, we prove that an optimal strategy for the mean-payoff game, which always exists, induces an optimal winning strategy for the RR game. This approach is complete due to the fact that in an RR game an optimal strategy can be assumed to have bounded waiting times. We begin by introducing mean-payoff games in Subsection 5.1 and then prove our main result in Subsection 5.2.

5.1 Mean-Payoff Games

A mean-payoff game $\mathcal{G} = (\mathcal{A}, w)$ consists of an arena $\mathcal{A}$ with set E of edges and a weight function $w \colon E \to \{-W, \ldots, W\}$ for some $W \in \mathbb{N}$. Given a play

$\rho = \rho_0 \rho_1 \rho_2 \cdots$ we define its value for Player 0 as

$$\nu_0(\rho) = \limsup_{n \to \infty} \frac{1}{n} \sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell) \ ,$$

and its value for Player 1 as

$$\nu_1(\rho) = \liminf_{n \to \infty} \frac{1}{n} \sum_{\ell=1}^{n} w(\rho_{\ell-1}, \rho_\ell) \ .$$

Intuitively, Player 0 wants to minimize $\nu_0(\rho)$ while Player 1 wants to maximize $\nu_1(\rho)$. Note that we always have $-W \le \nu_1(\rho) \le \nu_0(\rho) \le W$. For notational convenience we have swapped the roles of the players, i.e., classically Player 0's value is the lim inf and Player 1's value is the lim sup of the mean weights.

Theorem 2 ([11, 20]). *For every mean-payoff game there exist positional strategies σ_{opt} for Player 0 and τ_{opt} for Player 1 and values $\nu(v)$ for every vertex v such that*

1. *every play $\rho \in \mathrm{Beh}(v, \sigma_{\mathrm{opt}})$ satisfies $\nu_0(\rho) \le \nu(v)$, and*

2. *every play $\rho \in \mathrm{Beh}(v, \tau_{\mathrm{opt}})$ satisfies $\nu_1(\rho) \ge \nu(v)$.*

The strategies and values are computable in pseudo-polynomial time (i.e., in polynomial time in the size of the arena and in the maximal weight of an edge).

Especially, the unique play $\rho \in \mathrm{Beh}(v, \sigma_{\mathrm{opt}}) \cap \mathrm{Beh}(v, \tau_{\mathrm{opt}})$ satisfies $\nu_0(\rho) = \nu_1(\rho) = \nu(v)$. The strategy σ_{opt} is optimal in the sense that there is no strategy for Player 0 that guarantees a strictly smaller value than $\nu(v)$ when starting from v. The analogous statement is true for τ_{opt}.

5.2 Computing Optimal Strategies for RR Games via Mean-Payoff Games

In this subsection, we prove our main theorem: optimal strategies for RR games exist and can be effectively computed via the solution of a single mean-payoff game.

Theorem 3. *In every RR game, Player 0 has an optimal finite-state winning strategy, which is effectively computable.*

Proof. Let $\mathcal{G} = (\mathcal{A}, (Q_j, P_j)_{j \in [k]})$ be an RR game with s vertices and k RR conditions and let f_j be a strictly increasing penalty function for every j. Define

$$t_{\max_j} = f_j^{-1}(\mathrm{val}_\mathcal{G}) + b(s, k-1) \ ,$$

which satisfies $t_{\max_j} \ge 1$. Now, let $\mathfrak{M} = (M, \mathrm{Init}, \mathrm{Upd})$ where

$$M = \left(\prod_{j \in [k]} \{0, \ldots, t_{\max_j}\} \right) \cup \{\bot\}$$

20

is the set of all waiting time vectors whose values are bounded by $t_{\max_j}$ in coordinate j with an additional element $\bot$ denoting that the bound $t_{\max_j}$ is exceeded for some j. Furthermore, we define $\mathrm{Init}(v) = (t_1, \ldots, t_k)$ with

$$
t_j = \begin{cases} 1 & \text{if } v \in Q_j \setminus P_j, \\ 0 & \text{otherwise,} \end{cases}
$$

and $\mathrm{Upd}(\bot, v) = \bot$. It remains to define $\mathrm{Upd}((t_1, \ldots, t_k), v)$: if there is a j such that $t_j = t_{\max_j}$ and $v \notin P_j$, then we define $\mathrm{Upd}((t_1, \ldots, t_k), v) = \bot$. Otherwise, we have $\mathrm{Upd}((t_1, \ldots, t_k), v) = (t'_1, \ldots, t'_k)$ with (cf. the definition of the waiting time wt)

$$
t'_j = \begin{cases} 0 & \text{if } t_j = 0 \text{ and } v \notin Q_j \setminus P_j, \\ 1 & \text{if } t_j = 0 \text{ and } v \in Q_j \setminus P_j, \\ 0 & \text{if } t_j > 0 \text{ and } v \in P_j, \\ t_j + 1 & \text{if } t_j > 0 \text{ and } v \notin P_j. \end{cases}
$$

Each t'_j is again bounded by $t_{\max_j}$. Intuitively, the memory keeps track of the waiting times of play prefixes up to the thresholds $t_{\max_j}$. If a threshold is exceeded, a sink state is reached.

We define the mean-payoff game $\mathcal{G}' = (\mathcal{A} \times \mathfrak{M}, w)$ by

$$
w((v, (t_1, \ldots, t_k)), (v', m)) = \sum_{j \in [k]} f_j(t_j)
$$

for every memory state $m \in M$ and $w((v, \bot), (v', \bot)) = 1 + \sum_{j \in [k]} f_j(t_{\max_j})$. Due to f_j being strictly increasing, the maximal edge weight in $\mathcal{G}'$ is $1 + \sum_{j \in [k]} f_j(t_{\max_j})$, which appears only on the edges between vertices of the form $(v, \bot)$. We continue by stating some simple connections between plays in $\mathcal{G}$ and their extended plays in $\mathcal{G}'$.

Remark 4. *Let $\rho = \rho_0 \rho_1 \rho_2 \cdots$ be a play in $\mathcal{G}$ and $\mathrm{ext}(\rho)$ its extended play in $\mathcal{G}'$.*

1. *If $\mathrm{Upd}^*(\rho_0 \cdots \rho_n) \neq \bot$, then $\mathrm{Upd}^*(\rho_0 \cdots \rho_n) = \overline{\mathrm{wt}}(\rho_0 \cdots \rho_n)$.*

2. *If $\mathrm{Upd}^*(\rho_0 \cdots \rho_n) = \bot$, then there is a prefix $\rho_0 \cdots \rho_p$ of $\rho_0 \cdots \rho_n$ and an index j such that $\mathrm{wt}_j(\rho_0 \cdots \rho_p) > t_{\max_j}$ and every suffix $\rho_0 \cdots \rho_p \cdots \rho_s$ of $\rho_0 \cdots \rho_p$ satisfies $\mathrm{Upd}^*(\rho_0 \cdots \rho_p \cdots \rho_s) = \bot$.*

3. *If $\mathrm{ext}(\rho)$ does not visit the memory state $\bot$, then $\mathrm{val}(\rho) = \nu_0(\mathrm{ext}(\rho)) < 1 + \sum_{j \in [k]} f_j(t_{\max_j})$, i.e., the value of the play ρ in the RR game and the value of its extended play $\mathrm{ext}(\rho)$ in the mean-payoff game are equal (and smaller than the weight of the edges between the sink states with memory $\bot$) if the waiting times are bounded by $t_{\max_j}$.*

4. *If $\mathrm{ext}(\rho)$ visits the memory state $\bot$, then $\nu_0(\mathrm{ext}(\rho)) = 1 + \sum_{j \in [k]} f_j(t_{\max_j})$.*

21

Now, we can begin with the actual proof of Theorem 3, in which we have to deal with several strategies for Player 0. Throughout the proof, we denote strategies for $\mathcal{G}$ without a prime and strategies for $\mathcal{G}'$ with a prime. The strategies always come in pairs, one for the RR game $\mathcal{G}$ and one for the mean-payoff game $\mathcal{G}'$.

σ and σ' σ uniformly bounds the waiting times in $\mathcal{G}$. This strategy is turned into σ' for $\mathcal{G}'$ which never reaches the memory state $\bot$. This bounds the values $\nu(v)$ of the game $\mathcal{G}'$.

σ_{opt} and σ'_{opt} σ'_{opt} is an optimal strategy for $\mathcal{G}'$, which is turned into a strategy σ_{opt} for $\mathcal{G}$. Due to the properties of σ', we know that σ'_{opt} never reaches the memory state $\bot$, which in turn bounds the waiting times of σ_{opt}. Then, we show that σ_{opt} is indeed optimal.

$\widehat{\sigma}_{\mathrm{opt}}$ and $\widehat{\sigma}'_{\mathrm{opt}}$ To this end, we assume it is not optimal, i.e., there is a better strategy $\widehat{\sigma}_{\mathrm{opt}}$. This is turned into a strategy $\widehat{\sigma}'_{\mathrm{opt}}$ for $\mathcal{G}'$, which is strictly better than the optimal strategy σ'_{opt}. This contradiction finishes the proof.

Due to Corollary 1 and Lemma 5, there is a strategy σ for Player 0 for $\mathcal{G}$ such that $\mathrm{val}(\sigma, v) \leq \sum_{j \in [k]} f_j(sk2^k)$ for every $v \in W_0(\mathcal{G})$ and such that $\mathrm{wt}_j(w) \leq t_{\max_j}$ for every play prefix w that is consistent with σ and starts in $W_0(\mathcal{G})$. First, we turn σ into a strategy σ' for $\mathcal{G}'$ and use Remark 4 to relate their values. To this end, let

$$\sigma'((v_0, m_0) \cdots (v_n, m_n)) = (\sigma(v_0 \cdots v_n), \mathrm{Upd}(m_n, \sigma(v_0 \cdots v_n))) \ , \qquad (2)$$

i.e., we mimic the behavior of σ in the first component and update the memory state in the second component accordingly. Let $\rho' = (v_0, m_0)(v_1, m_1)(v_2, m_2) \cdots$ be consistent with σ'. A straightforward induction shows that $\mathrm{proj}(\rho') = v_0 v_1 v_2 \cdots$ is consistent with σ. Also, if $(v_0, m_0) = (v_0, \mathrm{Init}(v_0))$ then $\rho' = \mathrm{ext}(\mathrm{proj}(\rho'))$. If additionally $v_0 \in W_0(\mathcal{G})$ then an application of Remark 4(3) yields

$$\mathrm{val}(\mathrm{proj}(\rho')) = \nu_0(\rho') < 1 + \sum_{j \in [k]} f_j(t_{\max_j}) \ .$$

The following is now immediate.

Remark 5. *If $v \in W_0(\mathcal{G})$, then $\nu(v, \mathrm{Init}(v)) < 1 + \sum_{j \in [k]} f_j(t_{\max_j})$.*

Now, consider an optimal strategy σ'_{opt} for Player 0 in $\mathcal{G}'$ as guaranteed by Theorem 2. Due Remark 5, every play that starts in a vertex of the form $(v, \mathrm{Init}(v))$ for some $v \in W_0(\mathcal{G})$ never visits the memory state $\bot$. Now, let σ_{opt} be the strategy for $\mathcal{G}$ induced by σ'_{opt} with memory $\mathfrak{M}$. Formally, we define it by giving a next-move function via $\mathrm{Nxt}(v, m) = v'$ in case we have $\sigma'_{\mathrm{opt}}(v, m) = (v', m')$ for some m'. Let ρ be a play in $\mathcal{G}$ that is consistent with

σ_{opt} and starts in $W_0(\mathcal{G})$. A straightforward induction shows that $\mathrm{ext}(\rho)$ (which starts in $(v, \mathrm{Init}(v)))$ is consistent with σ'_{opt}. Thus, the memory state $\perp$ is never reached and we have $\mathrm{val}(\rho) = \nu_0(\mathrm{ext}(\rho))$.

We claim that σ_{opt} has the desired properties: it is finite-state and effectively computable. Hence, it remains to show that it is optimal. Assume it is not. Then, there exists a vertex v and a strategy $\widehat{\sigma}_{\mathrm{opt}}$ for Player 0 in $\mathcal{G}$ such that $\mathrm{val}(\widehat{\sigma}_{\mathrm{opt}}, v) < \mathrm{val}(\sigma_{\mathrm{opt}}, v) \leq \mathrm{val}_{\mathcal{G}}$. Due to Lemma 5, we can assume that $\widehat{\sigma}_{\mathrm{opt}}$ bounds the waiting times for every condition j by $t_{\max_j}$. Now, using the same definition as in (2), we turn $\widehat{\sigma}_{\mathrm{opt}}$ into a strategy $\widehat{\sigma}'_{\mathrm{opt}}$ for Player 0 in $\mathcal{G}'$.

As above, for every play $\rho' = (v_0, m_0)(v_1, m_1)(v_2, m_2)\cdots$ that is consistent with $\widehat{\sigma}'_{\mathrm{opt}}$ the projected play $v_0 v_1 v_2 \cdots$ is consistent with $\widehat{\sigma}_{\mathrm{opt}}$. Furthermore, if $(v_0, m_0) = (v_0, \mathrm{Init}(v_0))$, then $\rho' = \mathrm{ext}(\mathrm{proj}(\rho'))$ and $\nu_0(\rho') = \mathrm{val}(\mathrm{proj}(\rho'))$.

Recall that v is the vertex of $\mathcal{G}$ from which $\widehat{\sigma}_{\mathrm{opt}}$ is better than σ_{opt}. Now, consider the optimal strategy τ'_{opt} for Player 1 in $\mathcal{G}'$ (as in Theorem 2) and let ρ' be the unique play in $\mathcal{G}'$ that starts in $(v, \mathrm{Init}(v))$ and is consistent with both $\widehat{\sigma}'_{\mathrm{opt}}$ and τ'_{opt}. We have

$$\nu_0(\rho') = \mathrm{val}(\mathrm{proj}(\rho')) \leq \mathrm{val}(\widehat{\sigma}_{\mathrm{opt}}, v) < \mathrm{val}(\sigma_{\mathrm{opt}}, v) \leq \nu(v, \mathrm{Init}(v)) \leq \nu_1(\rho') \ ,$$

which yields the desired contradiction to the fact that we have $\nu_0(\rho') \geq \nu_1(\rho')$ by definition. Here, the inequality $\mathrm{val}(\sigma_{\mathrm{opt}}, v) \leq \nu(v, \mathrm{Init}(v))$ follows from the fact that every play that contributes to $\mathrm{val}(\sigma_{\mathrm{opt}}, v)$ has an extended play in $\mathcal{G}'$ that starts in $(v, \mathrm{Init}(v))$, is consistent with σ'_{opt}, and has the same value (for Player 0), which is smaller than $\nu(v, \mathrm{Init}(v))$ by Theorem 2. $\qquad\square$

6   Conclusion

We have presented an algorithm that computes optimal winning strategies for RR games in case the quality of a play is measured by the limit superior of the mean accumulated penalties on the waiting times between requests and their responses. To this end, we proved that the waiting times of strategies with small value can be assumed to be bounded by some doubly-exponential bound. Thus, the search space for an optimal winning strategy is finite and the problem of finding one can be reduced to computing an optimal strategy for a mean-payoff game.

The reduction presented here is also applicable to a more general winning condition, the so-called poset condition [18], where a request has to answered by a partially ordered set of events. In such games, the waiting times are used to measure the time between a request and the occurrence of the last event required in its response. Here, unlike in RR games, we measure the waiting time for every request, even if there is currently an open one. This is necessary since a new request might appear while an old request is already partially answered, i.e., satisfying the remaining events that answer the old request does not suffice to answer the new one. This situation cannot occur in RR games, as a response is a single event. Thus, by viewing an RR game as a poset game, we can also

compute an optimal strategy when measuring the quality by taking all requests into account.

Unfortunately, the reduction to mean-payoff games presented here is expensive in terms of running time of the algorithm and also in terms of the memory requirements of the optimal strategy: the size of the mean-payoff game is doubly-exponential and the largest weight in this game is doubly exponential (if the penalty functions are the identity function, otherwise these values are even larger). The best algorithms for mean-payoff games have a polynomial running time in these two parameters. Thus, our algorithm has a doubly-exponential running time. This has to be contrasted with the Exptime-completeness of computing an arbitrary winning strategy for an RR game [7]. Furthermore, the size of the memory structure implementing the optimal strategy for the RR game computed by our algorithm is also at least of doubly-exponential size, again larger than arbitrary winning strategies, which are of exponential size [7, 17].

As mentioned earlier, the upper bound on the waiting times is tight as shown by [9]. Hence, to obtain a faster algorithm and smaller optimal winning strategies, a different approach is necessary. The exact complexity of computing optimal strategies is an open problem. Another approach to overcome the high complexity is to consider heuristics and approximation algorithms, which compute strategies that realize the value of an optimal strategy up to a certain factor. Finally, the size of the optimal strategy computed here is much larger than the lower bounds on memory requirements in RR games. This raises the question whether there is a tradeoff between the size and the quality of a strategy.

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] 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.

[3] 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.

[4] 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.

[5] 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.

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

[7] 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.

[8] Krishnendu Chatterjee, Thomas A. Henzinger, and Marcin Jurdziński. Mean-payoff parity games. In *LICS*, pages 178–187. IEEE Computer Society, 2005.

[9] Wojciech Czerwiński, Tomasz Gogacz, and Eryk Kopczyński. Lower bound for Dickson's lemma in a special case, 2014. Under submission.

[10] Leonard E. Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *Amer. Journal Math.*, 35(4):413–422, 1913.

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

[12] 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.

[13] 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.

[14] 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.

[15] Orna Kupferman, Nir Piterman, and Moshe Y. Vardi. From liveness to promptness. *Formal Methods in System Design*, 34(2):83–103, 2009.

[16] Nico Wallmeier. *Strategien in unendlichen Spielen mit Liveness-Gewinnbedingungen: Syntheseverfahren, Optimierung und Implementierung.* PhD thesis, RWTH Aachen University, 2008.

[17] 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.

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

[19] Martin Zimmermann. Optimal bounds in parametric LTL games. *Theor. Comput. Sci.*, 493:30–45, 2013.

[20] Uri Zwick and Mike Paterson. The complexity of mean payoff games on graphs. *Theor. Comput. Sci.*, 158(1&2):343–359, 1996.