Time-Optimal Winning Strategies for Poset Games

Martin Zimmermann*

Lehrstuhl Informatik 7, RWTH Aachen University, Germany
zimmermann@automata.rwth-aachen.de

Abstract. We introduce a novel winning condition for infinite two-player games on graphs which extends the request-response condition and better matches concrete applications in scheduling or project planning. In a poset game, a request has to be responded by multiple events in an ordering over time that is compatible with a given partial ordering of the events. Poset games are zero-sum, but there are plays that are more desirable than others, i.e., those in which the requests are served quickly. We show that optimal strategies (with respect to long term average accumulated waiting times) exist. These strategies are implementable with finite memory and are effectively computable.

1 Introduction

The use of two-player games of infinite duration has a long history in the synthesis of controllers for reactive systems (see [3] for an overview). Classically, the quality of a winning strategy is measured in the size of the memory needed to implement it. But often there are other natural quality measures: in many games (even if they are zero-sum) there are winning plays for Player 0 that are more desirable than others, often given by notions of waiting that reflect periods of waiting in the modeled system. In reachability games, this can be the number of steps before the play reaches one of the designated vertices, in Büchi games the number of steps between visits of the designated vertices, and in parity games the number of steps between visits of the minimal even color seen infinitely often.

Another winning condition with a natural notion of waiting is the request-response condition [6]. It is given by pairs (Q_j, P_j) of subsets of the graph's vertices. Player 0 wins a play if every visit of Q_j is eventually responded by a visit of P_j. The waiting time is given by the number of steps between a request and the next response. As there might be several request-response pairs, there is a trade-off between the pairs: it can be favorable to delay the response of a pair to answer another request more quickly. Wallmeier [5] defined the value of a play to be the long-term average accumulated waiting time and the value of a strategy to be the worst-case outcome. He then proved that optimal winning strategies exist and are effectively computable (see also [4, 7]).

* This work was supported by the ESF project GASICS

However, request-response winning conditions are often too weak to express real-life requirements concisely, because a request is responded by a single event. Imagine an intersection with a level crossing: if a train approaches the crossing (a request), then all traffic lights have to be switched to red, then the boom barriers are lowered, the train gets an all-clear signal and crosses the intersection. Afterwards, the barriers are raised and the lights are switched to green. It would be rather cumbersome to model this requirement using request-response conditions with a single event as response. Another example is motivated by project planning: a project consists of several subtasks (and their durations) and a partial ordering of the subtasks, e.g., specifying that the roof of a house cannot be constructed before the walls are built. A plan is then a linearization of this partial ordering.

These examples motivate to replace a response by a partially ordered set of events and require Player 0 to answer every request by an embedding of these events in time. This generalization of request-response games retains the natural definition of waiting times. Hence, the framework for request-response games can be adapted to the new type of games, called poset games.

We prove that optimal winning strategies for poset games exist, which are again finite-state and effectively computable. To this end, we adapt the proof presented in [4] for request-response games. However, the increased expressiveness of poset games requires substantial changes. As a request is no longer responded by a single event, there can be different requests that are answered to a different degree at a given position, i.e., the embeddings can overlap. This requires additional bookkeeping of the events that still have to be embedded and changes to the definition of waiting times. Informally, in request-response games, there is a single clock for every pair (Q_j, P_j) that is started when Q_j is visited and stopped as soon as P_j is visited afterwards; requests that are encountered, while the clock is already active, are ignored. This is no longer possible in poset games: here, we need a clock for every request, due to the overlapping of embeddings. Hence, we do not only have to bound the waiting times to obtain our result, but also the number of open requests, i.e., the number of active clocks.

This paper is structured as follows: Section 2 fixes our notation and introduces poset games, which are solved by a reduction to Büchi games in Section 3. Finally, in Section 4 the existence of optimal strategies is proven. All proofs omitted due to space restrictions can be found in [8].

2 Definitions

Throughout this paper let P be a set of *events*. The power set of a set S is denoted by 2^S, $\mathbb{N}$ is the set of non-negative integers, and let $[n] := \{1, \ldots, n\}$. The prefix-ordering on words is denoted by $\sqsubseteq$, its strict version by $\sqsubset$. Given a sequence $(w_n)_{n \in \mathbb{N}}$ of finite words such that $w_n \sqsubset w_{n+1}$ for all n, $\lim_{n\to\infty} w_n$ denotes the unique ω-word induced by the w_n. Let $(f_n)_{n\in\mathbb{N}}$ be a sequence of functions $f_n : A \to B$ and $f : A \to B$. We say that $(f_n)_{n\in\mathbb{N}}$ *converges* to f, $\lim_{n\to\infty} f_n = f$, if $\forall a \in A\, \exists n_a \in \mathbb{N}\, \forall n \geq n_a : f_n(a) = f(a)$.

Infinite Games An *(initialized and labeled) arena* $G = (V, V_0, V_1, E, s_0, l_G)$ consists of a finite directed graph (V, E), a partition $\{V_0, V_1\}$ of V denoting the positions of *Player* 0 and *Player* 1, an *initial vertex* $s_0 \in V$, and a *labeling function* $l_G : V \to 2^P$. It is assumed that every vertex has at least one outgoing edge. A *play* $\rho = \rho_0\rho_1\rho_2\ldots$ is an infinite path starting in s_0. A *strategy for Player* i is a (partial) mapping $\sigma : V^*V_i \to V$ such that $(s, \sigma(ws)) \in E$ for all $w \in V^*$ and all $s \in V_i$. A play ρ is *consistent with* σ if $\rho_{n+1} = \sigma(\rho_0, \ldots \rho_n)$ for all $\rho_n \in V_i$. The unique play consistent with the strategies σ for Player 0 and τ for Player 1 is denoted by $\rho(\sigma, \tau)$.

A *game* $\mathcal{G} = (G, \varphi)$ consists of an arena G and a *winning condition* φ specifying the set of *winning plays* for Player 0. All other plays are won by Player 1. A strategy σ is a *winning strategy for Player* i if every play consistent with σ is won by Player i. Player i *wins* $\mathcal{G}$ (and Player $1 - i$ loses $\mathcal{G}$) if she has a winning strategy for $\mathcal{G}$. A game is *determined* if one of the Players has a winning strategy.

Game Reductions A *memory structure* $\mathfrak{M} = (M, m_0, \text{update})$ *for* G consists of a set M of *memory states*, an *initial memory state* $m_0 \in M$, and an *update function* $\text{update} : M \times V \to M$. The update function can be extended to a function $\text{update}^* : V^* \to M$ by defining $\text{update}^*(s_0) = m_0$ and $\text{update}^*(ws) = \text{update}(\text{update}^*(w), s)$. A *next-move function for Player* i $\text{next} : V_i \times M \to S$ has to satisfy $(s, \text{next}(s, m)) \in E$ for all $s \in V_i$ and all $m \in M$. It induces a *strategy* σ *with memory* $\mathfrak{M}$ via $\sigma(ws) = \text{next}(s, \text{update}^*(ws))$. A strategy is called *finite-state* if it can be implemented with finite memory, and *positional* if it can be implemented with a single memory state.

An arena G and a memory structure $\mathfrak{M}$ for G induce the *expanded arena* $G \times \mathfrak{M} = (V \times M, V_0 \times M, V_1 \times M, E', (s_0, m_0), l_{G\times\mathfrak{M}})$ where $((s, m), (s', m')) \in E'$ iff $(s, s') \in E$ and $\text{update}(m, s') = m'$, and $l_{G\times\mathfrak{M}}(s, m) = l_G(s)$. Every play $\rho' = (\rho_0, m_0)(\rho_1, m_1)(\rho_2, m_2)\ldots$ in $G \times \mathfrak{M}$ has a unique *projected play* $\rho = \rho_0\rho_1\rho_2\ldots$ in G. Conversely, every play $\rho = \rho_0\rho_1\rho_2\ldots$ in G has a unique *expanded play* $\rho' = (\rho_0, m_0)(\rho_1, m_1)(\rho_2, m_2)\ldots$ in $G\times\mathfrak{M}$ defined by $m_{n+1} = \text{update}(m_n, \rho_{n+1})$. A game $\mathcal{G} = (G, \varphi)$ is *reducible* to $\mathcal{G}' = (G', \varphi')$ via $\mathfrak{M}$, written $\mathcal{G} \leq_{\mathfrak{M}} \mathcal{G}'$, if $G' = G \times \mathfrak{M}$ and every play in $\mathcal{G}'$ is won by the player who wins the projected play in $\mathcal{G}$.

Remark 1. If $\mathcal{G} \leq_{\mathfrak{M}} \mathcal{G}'$ and Player i has a positional winning strategy for $\mathcal{G}'$, then she also has a finite-state winning strategy with memory $\mathfrak{M}$ for $\mathcal{G}$.

Poset Games A *(labeled) partially ordered set* (*poset* for short) $\mathcal{P} = (D, \preceq, l_{\mathcal{P}})$ consists of a domain D, a reflexive, antisymmetric and transitive relation $\preceq$ over D, and a labeling function $l_{\mathcal{P}} : D \to P$. The set of non-empty upwards-closed subsets of $\mathcal{P}$ is denoted by $\text{Up}(\mathcal{P})$; its size can be bounded by $|D| \leq |\text{Up}(\mathcal{P})| \leq 2^{|D|} - 1$.

Let ρ be an infinite path in an arena G with labeling function l_G. An *embedding in time*, *embedding* for short, of $\mathcal{P}$ in ρ is a function $f : D \to \mathbb{N}$ such that $l_{\mathcal{P}}(d) \in l_G(\rho_{f(d)})$ and $d \preceq d'$ implies $f(d) \leq f(d')$. An embedding of $\mathcal{P}$ in a finite path w is defined analogously.

A *poset game* $\mathcal{G} = (G, (q_j, \mathcal{P}_j)_{j\in[k]})$ consists of an arena G as above and a finite collection of *(request-poset) conditions* $(q_j, \mathcal{P}_j)$ where $q_j \in P$ is a *request (of condition j)* and $\mathcal{P}_j = (D_j, \preceq_j, l_j)$ is a finite labeled poset. Player 0 wins a play ρ iff $q_j \in l_G(\rho_n)$ implies that $\mathcal{P}_j$ can be embedded in $\rho_n\rho_{n+1}\rho_{n+2}\ldots$ for all $j \in [k]$ and all $n \in \mathbb{N}$.

To define the waiting times we need to keep track of the unanswered requests. For $j \in [k]$, $D \subseteq D_j$ and $s \in V$ let $\mathrm{New}_j(s) = D_j$ if $q_j \in l_G(s)$ and $\mathrm{New}_j(s) = \emptyset$ otherwise, and $\mathrm{Emb}_j(D, s) = \{d \in D \mid \exists d' \in D : d' \preceq_j d \text{ and } l_j(d') \notin l_G(s)\}$. The set $\mathrm{Emb}_j(D, s)$ contains the elements of D that cannot be embedded into s since a smaller element $d' \in D$ cannot be mapped to s. The *set of open requests of condition j after the finite play w* is defined inductively by $\mathrm{Open}_j(\varepsilon) = \emptyset$ and

$$\mathrm{Open}_j(ws) = \{(\mathrm{Emb}_j(D, s), t{+}1) \mid (D, t) \in \mathrm{Open}_j(w) \cup \{(\mathrm{New}_j(s), 0)\}\} \setminus \{\emptyset\} \times \mathbb{N}.$$

That is, $\mathrm{Open}_j(ws)$ deletes all those elements from the open requests D in $\mathrm{Open}_j(w)$ that can be embedded into s, adds a tick to the clock t of every request that is not yet responded completely, checks for new requests, and deletes responded requests. If $(D, t + 1) \in \mathrm{Open}_j(\rho_0 \ldots \rho_n)$, then there was a request of condition j at position $n - t$, the elements of $D_j \setminus D$ can be embedded into $\rho_{n-t} \ldots \rho_n$, and Player 0 has to embed all elements of D in the future to respond to this request.

Note that $\mathrm{Open}_j(w)$ contains only upwards-closed subsets of $\mathcal{P}_j$. The *number of open requests $D \in \mathrm{Up}(\mathcal{P}_j)$ of condition j after w* is $s_{j,D}(w) = |\{t \mid (D, t) \in \mathrm{Open}_j(w)\}|$. A set $D \in \mathrm{Up}(\mathcal{P}_j)$ is *open indefinitely* in $\rho_0\rho_1\rho_2\ldots$, if there exists a position n such that $(D, t) \in \mathrm{Open}_j(\rho_0 \ldots \rho_{n+t})$ for all $t > 1$.

Lemma 1. *Let $\rho = \rho_0\rho_1\rho_2\ldots$ be a play. For all $j \in [k]$:*

(i) If Player 0 wins ρ, then $(\mathrm{Open}_j(\rho_0 \ldots \rho_n))_{n\in\mathbb{N}}$ induces an embedding f_m of $\mathcal{P}_j$ in $\rho_m\rho_{m+1}\rho_{m+2}\ldots$ for every position m such that $q_j \in l_G(\rho_m)$.
(ii) ρ is won by Player 0 iff there is no $D \in \mathrm{Up}(\mathcal{P}_j)$ that is open indefinitely.

For the remainder of this paper, let $(G, (q_j, \mathcal{P}_j)_{j\in[k]})$ be a poset game, where $\mathcal{P}_j = (D_j, \preceq_j, l_j)$. Furthermore, let $c_j := |\mathrm{Up}(\mathcal{P}_j)|$ and $c := \sum_{j=1}^{k} c_j$.

3 Solving Poset Games

In this section, poset games are reduced to Büchi games. The memory stores the elements of the posets $\mathcal{P}_j$ that still have to be embedded. A cyclic counter ensures that all requests are responded by an embedding eventually.

Theorem 1. *Poset games are reducible to Büchi games and therefore determined with finite-state strategies.*

Proof. Let $h = \sum_{j=1}^{k} |D_j|$ and fix an enumeration $e : [h] \to \bigcup_{j=1}^{k} \{j\} \times D_j$. We assume $h > 1$ (without loss of generality) to obtain a nontrivial counter. The memory structure $\mathfrak{M} = (M, m_0, \mathrm{update})$ consists of $M = \prod_{j=1}^{k} \mathrm{Up}(\mathcal{P}_j) \times [h] \times \{0, 1\}$, $m_0 = (\mathrm{Emb}_1(\mathrm{New}_1(s_0), s_0), \ldots, \mathrm{Emb}_k(\mathrm{New}_k(s_0), s_0), 1, 0)$, and we define $\mathrm{update}((O_1, \ldots, O_k, m, f), s) = (O'_1, \ldots, O'_k, m', f')$ with

- $O'_j = \begin{cases} \mathrm{Emb}_j(D_j, s) & \text{if } q_j \in l_G(s) \\ \mathrm{Emb}_j(O_j, s) & \text{if } q_j \notin l_G(s) \end{cases}$,
- $m' = \begin{cases} (m \mod h) + 1 & \text{if } e(m) = (j, d) \text{ and } d \notin O'_j \text{ or } l_j(d) \in l_G(s) \\ m & \text{if } e(m) = (j, d) \text{ and } d \in O'_j \text{ and } l_j(d) \notin l_G(s) \end{cases}$,
- $f' = \begin{cases} 1 & \text{if } m \neq m' \\ 0 & \text{otherwise} \end{cases}$.

Finally, let $F = V \times \prod_{j=1}^{k} \mathrm{Up}(\mathcal{P}_j) \times [h] \times \{1\}$ and let $\mathcal{G}' = (G \times \mathfrak{M}, F)$ be a Büchi game in the expanded arena. Verifying $\mathcal{G} \leq_{\mathfrak{M}} \mathcal{G}'$ is now straightforward. Positional determinacy of Büchi games [3] and Remark 1 finish the proof.

If e is defined such that the elements of each domain D_j are enumerated consecutively and such that $d \preceq_j d'$ implies $e^{-1}(j, d) \leq e^{-1}(j, d')$, then it takes at most $h + |D_j|$ visits to vertices in F after a request of condition j to complete an embedding of $\mathcal{P}_j$ in the projected play.

The size of $\mathfrak{M}$ can be bounded by $|M| \leq h \cdot 2^{h+1}$, which is asymptotically optimal. This can be shown by transforming the family of request-response games presented in Theorem 2 of [6] into poset games.

4 Time-optimal Strategies for Poset Games

Waiting times for poset games are defined employing the information given by the open requests in $\mathrm{Open}_j(w)$. Define the

- *totalized waiting time for* $D \in \mathrm{Up}(\mathcal{P}_j)$ *after* w: $t_{j,D}(w) = \sum_{(D,t) \in \mathrm{Open}_j(w)} t$,
- *penalty after* w: $p(w) = \sum_{j=1}^{k} \sum_{D \in \mathrm{Up}(\mathcal{P}_j)} t_{j,D}(w)$,
- *value of a play* ρ: $v(\rho) = \limsup_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n-1} p(\rho_0 \dots \rho_i)$,
- *value of a strategy* σ: $v(\sigma) = \sup\{v(\rho(\sigma, \tau)) \mid \tau \text{ strategy for Player 1}\}$.

Hence, the influence of an open request on the value of a play grows quadratically in the waiting time, which penalizes longer waiting times more severely. A strategy σ for Player 0 is *optimal* if $v(\sigma) \leq v(\sigma')$ for all strategies σ' for Player 0. The following lemma is a simple consequence of Lemma 1.

Lemma 2. *Let* ρ *be a play and* σ *a strategy for Player* 0*.*

(i) If $v(\rho) < \infty$*, then Player* 0 *wins* ρ*.*
(ii) If $v(\sigma) < \infty$*, then* σ *is a winning strategy for Player* 0*.*

Note that the other directions of the statements are false: there are plays of infinite value that are won by Player 0.

Theorem 1 implies an upper bound on the value of an optimal strategy.

Corollary 1. *Let* $h = \sum_{j=1}^{k} |D_j|$*. If Player* 0 *wins* $\mathcal{G}$*, then she also has a winning strategy* σ *with*

$$v(\sigma) \leq \sum_{j=1}^{k} \left(c_j \cdot \frac{|M| \cdot |G|(h + |D_j|)(|M| \cdot |G|(h + |D_j|) + 1)}{2} \right) =: b_{\mathcal{G}} \ .$$

Let σ be a strategy for Player 0 and $D \in \mathrm{Up}(\mathcal{P}_j)$ for some condition j. We say that σ *uniformly bounds the waiting time for* D *to* b, if for all finite plays w consistent with σ it holds that $t \leq b$ for all $(D, t) \in \mathrm{Open}_j(w)$. Analogously, σ *uniformly bounds the totalized waiting time for* D *to* b, if $t_{j,D}(w) \leq b$ for all finite plays w consistent with σ. If the (totalized) waiting time for all $D \in \mathrm{Up}(\mathcal{P}_j)$ is bounded, then the length of the embeddings that respond to a request is also bounded.

Remark 2. Let σ be a strategy for Player 0. If σ uniformly bounds the waiting time for D to b, then σ also uniformly bounds the totalized waiting time for D to $\frac{1}{2}b(b+1)$.

We are now able to state the main theorem of this paper, which will be proved in the remainder of this section.

Theorem 2. *If Player* 0 *wins a poset game* $\mathcal{G}$, *then she also has an optimal winning strategy which is finite-state and effectively computable. The value of an optimal strategy is effectively computable as well.*

4.1 Strategy Improvement for Poset Games

We begin by defining a *strategy improvement operator* $I_{j,D}$ for every $D \in \mathrm{Up}(\mathcal{P}_j)$. It deletes loops of plays, consistent with the given strategy, that are spent waiting for a position into which an element from D has been embedded. Hence, the intervals in which D is an open request will be shorter if Player 0 plays according to the improved strategy. Doing this repeatedly will uniformly bound the waiting time $t_{j,D}$. However, the improved strategy has to ensure that no other responses get incomplete by deleting loops, i.e., the improved strategy is still winning for Player 0. Also, we do not want the value of the improved strategy to be greater than the value of the original strategy. We begin by defining the operator and then prove that it has the desired properties. Afterwards we show how to obtain uniform bounds on the waiting time by applying each $I_{j,D}$ infinitely often.

Let σ be a winning strategy (not necessarily finite-state) for Player 0 such that $v(\sigma) \leq b_{\mathcal{G}}$. The strategy $I_{j,D}(\sigma)$ is implemented with memory structure $\mathfrak{M} = (M, m_0, \mathrm{update})$ where M is a subset of the finite plays consistent with σ and defined implicitly. The initial memory state is $m_0 = s_0$ and $\mathrm{update}(w, s)$ is defined by a case distinction:

Player 0 only skips loops if the totalized waiting time for D is guaranteed to be higher than the value of the strategy, i.e., at least $b_{\mathcal{G}}$. Then, the value of the play does not increase from taking a shortcut. Thus, if $t_{j,D}(ws) \leq b_{\mathcal{G}}$, let $\mathrm{update}(w, s) = ws$. Hence, if the totalized waiting time is small, then she copies the original play according to σ.

Otherwise, if $t_{j,D}(ws) > b_{\mathcal{G}}$ consider the tree $\mathfrak{T}^{\sigma}_{ws}$ containing all finite continuations of ws that are consistent with σ restricted to those paths wsx such that $\mathrm{Open}_j(wsx') \cap (\{D\} \times \mathbb{N}) \neq \emptyset$ for all $x' \sqsubseteq x$. This tree contains all continuations of ws up to the point where the first element of the open request D can be embedded into. This tree is finite since σ is a winning strategy. The set of finite plays zs of $\mathfrak{T}^{\sigma}_{ws}$ such that $t_{j',D'}(zs) \geq t_{j',D'}(ws)$ and $s_{j',D'}(zs) \geq s_{j',D'}(ws)$ for all $j' \in [k]$ and all $D' \in \mathrm{Up}(\mathcal{P}_{j'})$ is non-empty as it contains ws. Let x be a play of maximal length in that set. Then, $\mathrm{update}(ws) = x$. So, if the totalized waiting time for D is sufficiently high, then Player 0 looks ahead whether ws is the start of a loop such that the totalized waiting times and the number of open requests for all $j' \in [k]$ and all $D' \in \mathrm{Up}(\mathcal{P}_{j'})$ are higher at the end of the loop than they were at the beginning. Then, she jumps ahead (by updating the memory to x) and continues to play as if she had finished the loop already.

The condition on $t_{j',D'}$ ensures that she does not miss a vertex that she has to visit in order to embed an element of the posets. This ensures that the improved strategy is still winning for Player 0. The condition on $s_{j',D'}$ guarantees that the value of the play does not increase from taking a shortcut by jumping ahead to a position where more requests will be open than before.

Finally, define $\mathrm{next}(s, ws) = \sigma(ws)$. Thus, Player 0's choice of the next move assumes that she has already finished the loops which were skipped by the memory update. The improved strategy $I_{j,D}(\sigma)$ is now given by $\mathfrak{M}$ and next.

Lemma 3. *Let σ be a winning strategy for Player 0, $j \in [k]$, and $D \in \mathrm{Up}(\mathcal{P}_j)$.*

(i) If σ bounds the totalized waiting time for some $D' \in \mathrm{Up}(\mathcal{P}_{j'})$ to b, then so does $I_{j,D}(\sigma)$.
(ii) $v(I_{j,D}(\sigma)) \leq v(\sigma)$.
(iii) $I_{j,D}(\sigma)$ is a winning strategy for Player 0.

In order to obtain small bounds on the waiting times, each improvement operator $I_{j,D}$ is now applied infinitely often to a given initial winning strategy. The limit of the strategies improved with $I_{j,D}$ uniformly bounds the totalized waiting time for D. Furthermore, all properties stated in Lemma 3 can be lifted to the limit strategy as well.

The order of improvement is given by enumerations $e_j : [c_j] \to \mathrm{Up}(\mathcal{P}_j)$ such that $|D| > |D'|$ implies $e_j^{-1}(D) < e_j^{-1}(D')$. Thus, the subsets are enumerated in order of decreasing size. Given a winning strategy σ_0 for Player 0 such that $v(\sigma_0) \leq b_{\mathcal{G}}$ (whose existence is guaranteed by Corollary 1), define

- $\sigma_{j,l,0} = \begin{cases} \sigma_{j-1} & \text{if } l = 1 \\ \sigma_{j,l-1} & \text{otherwise} \end{cases}$ for $j \in [k]$ and $l \in [c_j]$,
- $\sigma_{j,l,n+1} = I_{j,e_j(l)}(\sigma_{j,l,n})$ for $j \in [k]$, $l \in [c_j]$, and $n \in \mathbb{N}$,
- $\sigma_{j,l} = \lim_{n\to\infty} \sigma_{j,l,n}$ for $j \in [k]$ and $l \in [c_j]$, and
- $\sigma_j = \sigma_{j,c_j}$ for $j \in [k]$.

For notational convenience, let $\sigma_{j,0} = \sigma_{j-1}$ for $j \in [k]$. Before we discuss the properties of the strategies defined above, we need to introduce some additional notation that is used to bound the waiting times.

The strategy improvement operator $I_{j,D}$ skips a loop if the vertices at the beginning and at the end coincide and the values $s_{j',D'}$ and $t_{j',D'}$ at the end are greater than or equal to the values at the beginning. Hence, we say that two finite plays $y_1 \sqsubset y_2$ form a *Dickson pair* [1] if their last vertices coincide and $s_{j',D'}(y_1) \leq s_{j',D'}(y_2)$ and $t_{j',D'}(y_1) \leq t_{j',D'}(y_2)$ for all $j' \in [k]$ and all $D' \in \mathrm{Up}(\mathcal{P}_{j'})$. Dickson pairs are candidates for deletion by $I_{j,D}$.

The set D is in Open_j throughout a loop skipped by $I_{j,D}$. Accordingly, an infix $\rho_m \dots \rho_{m+n}$ of a play ρ is called *non-Dickson save D* if $t_{j,D}$ increases strictly monotonic throughout the infix and if there are no $m \leq g < h \leq m+n$ such that $\rho_0 \dots \rho_g$ and $\rho_0 \dots \rho_h$ are a Dickson pair. The length of such an infix can be bounded inductively by a function b in the size n of G and in $c = \sum_{j=1}^{k} |\mathrm{Up}(\mathcal{P}_j)|$.

If $c = 1$, then the single set is D, whose values increase monotonically. Hence, there is a vertex repetition after at most $|G|$ steps. Therefore, $b(n,1) = n$.

If $c+1 > 1$, then $t_{j',D'}$ (and thereby also $s_{j',D'}$) has to be reset to 0 after at most $b(n,c)$ steps for every $D' \neq D$. If not, then the initial prefix of length $b(n,c)$ contains a Dickson pair by induction hypothesis. For the same reason, for every $c' \in [c]$ there are c' sets D' such that $t_{j',D'}$ (and also $s_{j',D'}$) was reset to 0 in the last $b(n,c')$ steps. If not, then this infix would again contain a Dickson pair by induction hypothesis. Accounting for all possible combinations, we obtain $b(n,c+1) = b(n,c) + nc! \prod_{j=1}^{c} \frac{1}{2}(b(n,j))^2(b(n,j)+1)$, as we have $t_{j',D'}(xy) \leq \frac{1}{2}|y|(|y|+1)$ and $s_{j',D'}(xy) \leq |y|$ if $t_{j',D'}(x) = 0$.

Note that the same idea can be applied to request-response games, which lowers the bounds given in [4, 5].

Now, we are able to lift the properties of the strategy improvement operator to the limit of the improved strategies and to bound the waiting times.

Lemma 4. *Let $j \in [k]$, $l \in [c_j]$, and $e_j(l) = D$. Then:*

(i) $\lim_{n\to\infty} \sigma_{j,l,n}$ *exists.*
(ii) If $\sigma_{j,l-1}$ uniformly bounds the totalized waiting time for some $D' \in \mathrm{Up}(\mathcal{P}_{j'})$, then so does $\sigma_{j,l}$.
(iii) $v(\sigma_{j,l}) \leq v(\sigma_{j,l-1})$, and therefore $v(\sigma_j) \leq v(\sigma_{j-1})$.
(iv) $\sigma_{j,l}$ uniformly bounds the waiting time for D to

$$b_{j,D} := b_{\mathcal{G}} + (|D_j \backslash D| + 1) \cdot b\,(|G|, c) \quad .$$

These properties of the improved strategies can be combined to show that the waiting times can be bounded without increasing the value of a strategy.

Lemma 5. *For every winning strategy σ_0 for Player 0 with $v(\sigma_0) \leq b_{\mathcal{G}}$, there is a winning strategy σ_k for Player 0 that bounds $s_{j,D}$ to $b_{j,D}$ and $t_{j,D}$ to $tb_{j,D} := \frac{1}{2}(b_{j,D}(b_{j,D}+1))$ for all $j \in [k]$ and all $D \in \mathrm{Up}(\mathcal{P}_j)$. Furthermore, $v(\sigma_k) \leq v(\sigma_0)$.*

4.2 Reducing Poset Games to Mean-Payoff Games

In this subsection, we reduce the poset game to a mean-payoff game [2], which we will introduce in the following.

A *mean-payoff game* $\mathcal{G} = (G, d, l)$ consists of an arena $G = (V, V_0, V_1, E, s_0)$, $d \in \mathbb{N}$ and a labeling function $l : E \to \{-d, \dots, d\}$ (note that l labels the edges in this case). Let ρ be a play in G. The *gain* $v_0(\rho)$ for Player 0 is defined as $v_0(\rho) = \liminf_{n\to\infty} \frac{1}{n}\sum_{i=0}^{n-1} l(\rho_i, \rho_{i+1})$ and the *loss* $v_1(\rho)$ for Player 1 is $v_1(\rho) = \limsup_{n\to\infty} \frac{1}{n}\sum_{i=0}^{n-1} l(\rho_i, \rho_{i+1})$. Player 0's goal is to maximize $v_0(\rho)$ whereas Player 1 aims to minimize $v_1(\rho)$. A strategy σ for Player 0 guarantees a gain of v if $v_0(\rho) \geq v$ for every play ρ consistent with σ. Analogously, τ for Player 1 guarantees a loss of v if $v_1(\rho) \leq v$ for every play ρ consistent with τ.

Theorem 3 ([2, 9]). *Let $\mathcal{G}$ be a mean-payoff game. There exists a value $\nu_{\mathcal{G}}$ and positional strategies σ and τ that guarantee $\nu_{\mathcal{G}}$ for Player 0 and Player 1, respectively. These strategies are optimal, i.e., there is no strategy for Player i that guarantees a better value for her. Furthermore, σ, τ and $\nu_{\mathcal{G}}$ are computable in pseudo-polynomial time.*

Now, we explain the reduction. The memory keeps track of the totalized waiting time $t_{j,D}(w)$ for every $j \in [k]$ and every $D \in \mathrm{Up}(\mathcal{P}_j)$. To be able to compute $t_{j,D}(ws)$ from $t_{j,D}(w)$ in every update of the memory state, $s_{j,D}(w)$ has to be stored as well. Due to Lemma 5 we can bound $t_{j,D}(w)$ by $tb_{j,D}$ and $s_{j,D}(w)$ by $b_{j,D}$. If these bounds are exceeded, then the memory is updated to a sink state $m_\uparrow$. Hence, we obtain a finite memory structure $\mathfrak{M}$. The formal definition is straightforward, but technical, and can be found in the long version of this paper [8].

The arena for the mean-payoff game $\mathcal{G}'$ is $G \times \mathfrak{M}$ where an edge is labeled by the sum of the totalized waiting times at the source of the edge. The value d is defined appropriately and is also the weight of all edges originating from a vertex with memory state $m_\uparrow$. As it is Player 1's goal to minimize the limit superior of the average edge labels, we have to exchange the positions of the players. This finishes the definition of $\mathcal{G}'$.

If the totalized waiting times in play ρ of the poset game $\mathcal{G}$ are bounded by $tb_{j,D}$, then the values $v(\rho)$ and $v_1(\rho')$ are equal, where ρ' is the expanded play of the mean-payoff game $\mathcal{G}'$. Dually, if a play ρ' of $\mathcal{G}'$ avoids the vertices with memory state $m_\uparrow$, then $v_1(\rho') = v(\rho)$, where ρ is the projected play of ρ'.

Now, we are able to prove Theorem 2: let Player 0 win $\mathcal{G}$. Corollary 1 and Lemma 5 imply that there is a strategy for Player 1 in $\mathcal{G}'$ that avoids the vertices with memory state $m_\uparrow$. Hence, the value $\nu_{\mathcal{G}'}$ is smaller than d and an optimal strategy for Player 1 for $\mathcal{G}'$ avoids the vertices with memory state $m_\uparrow$, too. It is now easy to show that an optimal positional strategy for Player 1 for $\mathcal{G}'$ induces an optimal finite-state strategy for Player 0 for $\mathcal{G}$. Furthermore, the values of both optimal strategies coincide.

5 Conclusion

We have introduced a novel winning condition for infinite two-player games that extends the request-response condition while retaining a natural definition of waiting times. These games are well-suited to add aspects of planning to the synthesis of finite-state controllers for reactive systems. We proved that optimal strategies (with respect to long-term average accumulated waiting times) exist and are effectively computable. The memory size of the optimal strategy computed here is super-exponential. However, this holds already for request-response games. Thus, the increased expressiveness of the poset condition does not add too much additional complexity.

In future research, the memory size should be analyzed: determining the computational complexity of finding optimal strategies and proving tight upper and lower bounds on the memory size of an optimal strategy. The size of the mean-payoff game (and thus the memory) can be reduced by finding better bounds on the length of non-Dickson infixes. Also, one should investigate, whether the (costly, in terms of time and space) reduction to mean-payoff games is necessary: can an optimal strategy be computed without a reduction?

Another direction of further research is to consider *discounted* waiting times and to establish a reduction to *discounted payoff games* [9]. Furthermore, the reduction to Büchi games induces a uniform upper bound on the waiting times in poset games, but the (efficient) computation of optimal bounds should be addressed as well.

Acknowledgments This work presents results of the author's diploma thesis [7] prepared under the supervision of Wolfgang Thomas. I want to thank him for his advice and suggestions. Also, I want to thank the anonymous referees for their helpful remarks.

References

1. Leonard E. Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *Amer. J. Math.*, 35(4):413–422, 1913.
2. Andrzej Ehrenfeucht and Jan Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 8(2):109–113, 1979.
3. Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games*, volume 2500 of *LNCS*. Springer, 2002.
4. 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.
5. Nico Wallmeier. *Strategien in unendlichen Spielen mit Liveness-Gewinnbedingungen: Syntheseverfahren, Optimierung und Implementierung.* PhD thesis, RWTH Aachen University, 2008.
6. 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 *Lecture Notes in Computer Science*, pages 11–22. Springer, 2003.
7. Martin Zimmermann. *Time-optimal Winning Strategies in Infinite Games.* Diploma Thesis, RWTH Aachen University, 2009. `automata.rwth-aachen.de/~zimmermann`.
8. Martin Zimmermann. Time-optimal winning strategies for poset games. Technical Report AIB-2009-13, RWTH Aachen University, 2009.
9. Uri Zwick and Mike Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158(1&2):343–359, 1996.