Optimal Strategy Synthesis for Request-Response Games

Florian Horn[†] *, Wolfgang Thomas* and Nico Wallmeier*

† LIAFA, Université Paris 7, Case 7014, 2 place Jussieu, F-75251 Paris 5, France
∗ Lehrstuhl für Informatik 7, RWTH Aachen University, 52056 Aachen, Germany
horn@liafa.jussieu.fr,
{thomas,wallmeier}@informatik.rwth-aachen.de

Abstract. We show the solvability of an optimization problem on infinite two-player games. The winning conditions are of the "request-response" format, i.e. conjunctions of conditions of the form "if a state with property Q is visited, then later also a state with property P is visited". We ask for solutions that do not only guarantee the satisfaction of such conditions but also minimal wait times between visits to Q-states and subsequent visits to P-states. We present a natural class of valuations of infinite plays that captures this optimization problem, and with respect to this measure show the existence of an optimal winning strategy (if a winning strategy exists at all) and that it can be realized by a finite-state machine. For the latter claim we use a reduction to the solution of mean-payoff games due to Paterson and Zwick.

1 Introduction

Infinite two-player games are a natural model of reactive systems in which a controller interacts with moves of its environment and has to guarantee certain conditions on the infinite runs on the system (*i.e.*, infinite plays in the game theoretic view). A typical condition that arises in many contexts is the "request-response condition". It refers to state properties $Q_1, \ldots, Q_k$ which capture k different types of "requests", and other state properties $P_1, \ldots, P_k$ which represent the corresponding "responses", usually seen as satisfaction of the requests. The associated request-response condition is the following requirement on an infinite state sequence (play of the game): *For each i, whenever a state in Q_i is visited, then now or later a state in P_i is visited.* In linear time temporal logic the condition is formalized as $\mathrm{G}(Q_i \rightarrow \mathrm{F}P_i)$. In practice, request-response conditions often occur in the presence of safety conditions. We assume here that the state space is restricted to states that satisfy the safety condition.

In this paper we analyze request-response games over finite arenas, taking in account not only the mere satisfaction of the winning condition

but also the quantitative aspect of minimizing the delay between Q_i visits and the corresponding subsequent P_i-visits. A quantitative sharpening of liveness conditions has been studied by several authors. For instance, in their work on parameterized temporal logic, Alur et al. [1] supplement temporal operators by constants that give bounds in the semantics, e.g., F_k meaning "eventually, within k steps". A more abstract view is to ask for the existence of an unspecified bound: Kupferman, Piterman, and Vardi [10] define the "prompt" operator F_p, meaning that there is a constant k that bounds the satisfaction of an eventuality formula over all runs of a system; Chatterjee and Henzinger [3] consider finitary games, where the waiting time between a request and its satisfaction must be ultimately bounded. In the present paper we go a step further in the sense that we try a achieve "best bounds" rather than asking whether given bounds can be met or whether some bound exists. Thus we study an optimization problem rather than a decision problem. Moreover, we work in the context of open rather than closed systems.

In order to solve this optimization problem, we introduce valuations for infinite plays (by real numbers) that measure globally the delays in a play between visits to request- and subsequent response-states. The approach we take here is to associate with any time instance where a request is "open" a corresponding "penalty". In Section 2 we discuss possible conventions and then pursue a definition which stresses the requirement of avoiding long wait times, namely the case of strictly growing and diverging penalties: The penalties $v_1, v_2, v_3, \ldots$ for waiting $1, 2, 3, \ldots$ moments of time strictly increase and give a diverging sequence of real numbers. For a finite play prefix we take the sum of the occurring penalties divided by its length, and as value of a play we define the limsup of the prefix values. The corresponding notion of optimal winning strategy (of player "controller") is now obvious.

Our main result states that for a finite-state request-response game (with some given initial state) one can not only decide whether controller wins and provide a finite-state strategy (which was known), but that also – with respect to the mentioned valuation – an optimal winning strategy exists, is computable, and can be realized by a finite-state machine.

The paper is structured as follows: In the subsequent Section 2 we introduce the technical preliminaries on game arenas, request-response games, the valuation of plays, and optimality of strategies. In Section 3 we state the main result. Section 4 is devoted to the key lemma which states that an optimal strategy realizes a uniform bound on the delays. This allows to study the optimization problem over a finite game graph

(resulting from the given one by attaching all possible tuples of delays up to the mentioned bound) and to invoke (in Section 5) known results on mean-payoff games for computing an optimal strategy. In Section 6, we conclude with a discussion and some open problems.

The paper extends (and gives a more streamlined exposition of) results of the third author's dissertation [12, Chapter 4].

2 Definitions

2.1 Games

We recall here some background on infinite games used in specification and verification. We refer the reader to [12,6] for more details.

Arenas. A *game arena* $\mathcal{A}$ consists of a directed finite graph $(\mathcal{S}, \mathcal{T})$, a partition $(\mathcal{S}_E, \mathcal{S}_A)$ of $\mathcal{S}$, and an initial state s_0. The states in $\mathcal{S}_E$ (resp. $\mathcal{S}_A$) are *Eve's states, or Controller's states* (resp. *Adam's states, or Environment's states*), and are graphically represented as $\bigcirc$ (resp. $\Box$) in figures.

Plays. A *play of* $\mathcal{A}$ is a finite or infinite sequence $\rho = \rho_0\rho_1\ldots$ of states such that $\rho_0 = s_0$ and $(\rho_i, \rho_{i+1}) \in \mathcal{T}$ for all $i <$ length of ρ. The set of *occurring states* is $\mathrm{Occ}(\rho) = \{s \in \mathcal{S} \mid \exists i \in \mathbb{N}, \rho_i = s\}$.

Strategies. A *strategy for Eve* is a function σ from $\mathcal{S}^*\mathcal{S}_E$ to $\mathcal{S}$ such that for any path w ending in state q, there is a transition in $\mathcal{T}$ between q and $\sigma(w)$. Intuitively, it is a "recipe" to extend a finite play ending in one of Eve's states.

A popular and equivalent definition uses memory states: a *strategy with memory* M for Eve is a transducer $\sigma = (M, \nu, \mu)$, where ν is the "next-move" function from $(\mathcal{S}_E \times M)$ to $\mathcal{S}$ such that $\nu(s, m) \subseteq \mathcal{T}(s)$ and μ is the "memory-update" function, from $(\mathcal{S} \times M)$ to M. A strategy is *finite-memory, or finite-state* if M is a finite set, and *memoryless* if M is a singleton.

For two given strategies σ and τ, the outcome of the game is an infinite play $\rho^{\sigma,\tau}$. A play ρ is *consistent with* σ (resp. τ) if there is a strategy τ (resp. σ) such that ρ is a prefix of, or equal to $\rho^{\sigma,\tau}$.

Winning Conditions. A *winning condition* is a function way to grade a play. Usually, this is a boolean function, used to partition the plays between those *winning for Eve* and those *winning for Adam*. The main problem is then to find *winning strategies*: A strategy $\sigma \in \Sigma$ is winning for Eve if for every strategy τ of Adam, the play $\rho^{\sigma,\tau}$ is winning for Eve. When games are *determined* (as are request-response games), there is always a winning strategy for one of the players.

2.2 Request-Response Games

Request-Response games were introduced in [14]. They are a special case of ω-regular games. Let $\mathcal{G} = (\mathcal{S}, \mathcal{S}_E, \mathcal{S}_A, \mathcal{T})$ be an arena. A play is winning for Eve under the *request-response condition* $\mathcal{C} = (Q_j, P_j)_{j=1...k}$ (where $Q_j, P_j \subseteq \mathcal{S}$) iff $\bigwedge_{j=1}^{k} \forall n(\rho_n \in Q_j \Rightarrow \exists m \geq n\ \rho_m \in P_j)$.

Rather than the mere satisfaction of the request-response condition, we study the question *how well* the controller (Eve) can satisfy it. Consider for example the following game, which represents a controller for traffic lights in a north-south one-lane road. In a single round, up to three cars (c_1, c_2, c_3) can come to the one-lane section from either side — Adam's choice. For example, $\overline{c_1}$ means that the first car comes from the north, while $\underline{c_2}$ means that the second car comes from the south. Eve can then choose which side has the green light for the current round: $\overline{g}$ for southbound traffic, and $\underline{g}$ for northbound traffic. A simple winning strategy

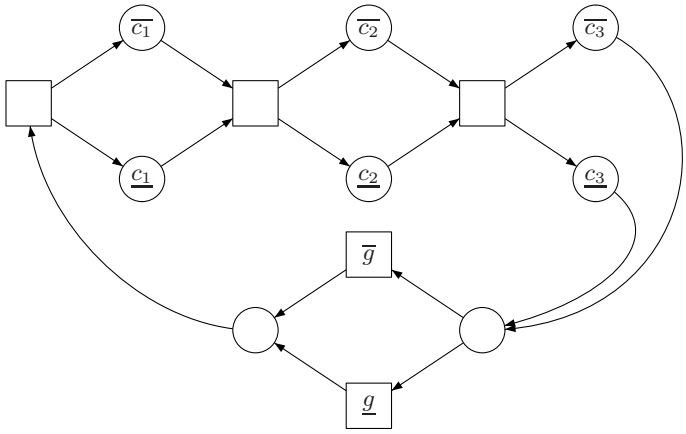

Fig. 1. Example of request-response game (traffic controller)

would be to alternate between north and south. Clearly a "better" solution would be to give, at each moment of choice, the green light to the side with the majority of cars.

In order to capture the quality of strategies, we define the *value* of a play as a real number. This is done in two steps: to each position in a play, we associate a *penalty*, depending on the time since when the open requests are waiting for a response; then, we define the value of a play as the limsup over the averages for all finite play prefixes. The penalties (paid by Eve) count only steps for as yet unanswered requests. In particular,

the value of a play without any request is zero, while a play winning for Adam has an infinite value.

Formally, we use the following definitions in order to define the value. The wait time for the condition (Q_j, P_j) at the end of the (finite) play prefix w, denoted by $t_j(w)$, is the number moves since the earliest unanswered visit to Q_j in w. If all visits to Q_j have been answered, its value is 0. The value $t_j(w)$ is defined recursively. We set $t_j(\epsilon) = 0$ and let

- if $t_j(w) = 0$
  - $t_j(ws) := 1$ if $s \in Q_j \setminus P_j$
  - $t_j(ws) := 0$ otherwise
- if $t_j(w) > 0$
  - $t_j(ws) := 0$ if $s \in P_j$
  - $t_j(ws) := t_j(w) + 1$ otherwise.

By definition of the wait times, we can immediately derive the following remark about their evolution:

$$\forall v, w \in \mathcal{S}^*, s \in \mathcal{S} : t_j(v) < t_j(w) \Rightarrow t_j(vs) < t_j(ws) \tag{1}$$

Based on the wait time $t_j(w)$ for condition (Q_j, P_j) at play prefix w we can introduce different kinds of "penalties" (for Eve). A simple choice would be to count one unit of penalty for each time instance of waiting; this leads to a linear increase of the accumulated penalties while time elapses. In this measure, an extra moment of waiting costs the same regardless of the past. In realistic scenarios it seems more appropriate to let the penalties increase, reflecting the idea that longer wait times pose higher pressure for delivering immediate response. A natural implementation of this idea is to associate penalty p for the p-th moment of (successive) waiting, which leads to a quadratic increase of accumulated penalties over time. In order to capture the different options, we introduce (for condition (Q_j, P_j)) a *penalty function* $f_j : \mathbb{N} \to \mathbb{R}_{\geq 0}$ (which was the constant 1, respectively the identity on $\mathbb{N}$ in the examples above), and we define the *penalty* $p_j(w)$ *of play prefix* w as $f_j(t_j(w))$. The *global penalty* over all conditions (Q_j, P_j) is the sum $p(w)$ over all $p_j(w)$, and the *value of an infinite play* ρ, denoted $v(\rho)$, is the mean value of the costs $p(w)$ of its prefixes w. – Accordingly, the *value of a strategy* σ, denoted $v(\sigma)$, is the limsup of the values of the plays consistent with this strategy. To sum up:

- $p_j(w) = f_j(t_j(w))$
- $p(w) = \sum_{j=1}^{k} p_j(w)$
- $v(\rho) = \limsup_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v(\rho_0 \dots \rho_{i-1})$

– $v(\sigma) = \limsup_\tau v(\rho^{\sigma,\tau})$

Our main result below is stated for the case of *strictly increasing and unbounded penalty functions* f_j. We have argued for this restriction from a pragmatic point of view. Let us also observe that – for example – for the case of constant penalties one cannot hope to construct optimal winning strategies.

Example 1. In the game of Figure 2, two requests are made each time the token reaches the central state. Eve can choose to satisfy either request, by going left or right. It is more costly to satisfy the second condition since the loop is longer. The request-response game is won by Eve with a strategy σ_k which goes to the left only each k-th turn. It is easy to check that — with constant penalties — no optimal winning strategy exists; note that the value of σ_{k+1} is an improvement over the value of σ_k. Our main result shows that for increasing and diverging penalty functions optimal strategies exist and can be computed.

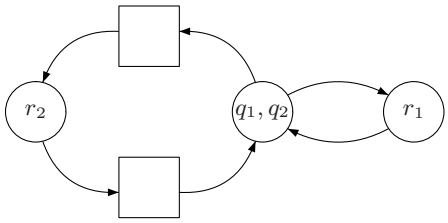

Fig. 2. No optimal solution for constant penalties

3 Main Result

In the sequel we consider request-response games with k conditions (Q_j, P_j) $(j = 1, \ldots, k)$, and *we assume strictly increasing and unbounded penalty functions* f_j for $j = 1, \ldots, k$. We state the main result:

Theorem 2. *If a request-response game is won by Eve, then Eve in fact has (with respect to strictly increasing and unbounded penalty functions) an optimal winning strategy, which moreover is finite-state and effectively computable.*

As a preparation we recall the solution given in [14] for classical ("Boolean") request-response games and deduce a bound for the value of a canonical winning strategy.

Proposition 3. *If Eve wins in a request-response game over G with n states and k conditions, she has a winning strategy of value $\leq M := \sum_{j=1}^{k} f_j(n * k)$.*

Let us sketch the proof, which involves an easy reduction to Büchi games. Starting from a game arena $G = (\mathcal{S}, \mathcal{S}_E, \mathcal{S}_A, \mathcal{T})$ and k conditions, an expanded graph is constructed over the vertex set $\mathcal{S}' := \mathcal{S} \times \{0,1\}^k \times [1, \ldots, k]$; the bit vector signals which of the k conditions have an open request, and the final index from $[1, k]$ signals which request to serve next. The index 0 is assumed everytime after index k is reached, signalling a successful cycle of satisfaction of requests. (The Büchi condition then requires to visit vertices in $\mathcal{S}'$ with final component 0 again and again.) Using well-known fixed point computations (see [12,6]) one can decide whether Eve wins this Büchi game. Moreover, it is easy to see that the "round robin" winning strategy σ_0 derived from the game reduction will guarantee uniformly bounded wait times for each condition. A "next response" is reached after $\leq n$ steps (where n is the size of the arena), and it can take k such responses until the currently considered request is served. Thus, the reduction to Büchi games yields a bound of $n*k$ for the wait times, and the value of the corresponding strategy σ_0 is $v(\sigma_0) \leq M := \sum_{i=1}^{k} f_i(n * k)$.

We often refer to the value $f_j^{-1}(M)$, i.e. the smallest number s of steps (of waiting) such that $f_j(s) \geq M$. We shall show that for any strategy with value $\leq M$, there exists a strategy with smaller or equal value, which uniformly bounds the wait times for each pair (Q_j, P_j). Let us state the main technical lemma.

Lemma 4. *There is a function $d : \mathbb{N}^2 \to \mathbb{N}$ such that the following holds for request-response games over graphs with n vertices and with k conditions: From any strategy σ with value $v(\sigma) \leq M$, we can compute a strategy σ' that bounds, for $j = 1, \ldots, k$, all wait times for the pair (Q_j, P_j) to $b_j = f_j^{-1}(M) + n * d(n, k)$, and such that $v(\sigma') \leq v(\sigma)$.*

This result is the essential step in our analysis of request-response games. It says that in the domain of strategies of value at most M, we can achieve a uniform bound on the wait times and still keep or improve the value of the strategy. Intuitively this means that it is not possible to improve the value of a strategy by deferring the responses for some condition more and more (namely, beyond the bound M) while preferring

to "serve" another condition. In other words, the effect of Example 1 is excluded.

To obtain the main result, we shall show how these "bounded" strategies can be interpreted as strategies in finite-state mean-payoff games.

4 Bounding Wait Times

This section gives the proof of Lemma 4.

The main problem in the construction of optimal strategies is that *a priori*, the wait times for the pairs can be unbounded if the long spells where a condition remain open occur rarely. The point of this section is to show that we can restrict our study to strategies where the wait times are bounded. Furthermore, the bounds can be computed beforehand, directly from the parameters of the game.

In the proof of Lemma 4, we modify a strategy σ_0 k times, thus constructing strategies $\sigma_1, \ldots, \sigma_k$. At step j, the strategy σ_{j-1} is "improved" into one respecting the desired wait time bound for condition (Q_j, P_j), with the following constraints:

- The value of σ_j is less or equal to the value of σ_{j-1}.
- For any pair (Q_i, P_i), if σ_{j-1} bounds t_i to B, then σ_j bounds t_i to B.

The function d occurring in Lemma 4 comes from a special version of Dickson's Lemma [4]. We consider here play prefixes w by their last state and their wait time vector $t(w) := (t_1(w), \ldots, t_k(w))$. Note that in a play the components $t_j(w)$ can only increase by 1 or be reset to 0.

This allow us to bound recursively the length of the longest "non-Dickson" sequence. It is clear that for n states and 0 pairs, this length is n, so $d(n, 0) = n + 1$. Let us consider now a non-Dickson sequence for n states and $k + 1$ pairs. For any pair (Q_i, P_i), the length between two positions such that $t_i = 0$ is at most $d(n, k) - 1$: as t_i is non-decreasing, removing it must yield a non-Dickson sequence for n states and k pairs. The waiting times can only increase by 1, so their values cannot exceed $d(n, k) - 1$, and the number of possible configurations in a non-Dickson sequence is at most $n \cdot (k + 1)^{d(n,k)-1}$. We define the recursive function d as follows:

- $d(n, 0) = n + 1$
- $d(n, k + 1) = n \cdot (k + 1)^{d(n,k)-1} + 1$

Thus, we have shown the following lemma:

Lemma 5. *In a request-response game with n states and k conditions, in any play w of length at least $d(n,k)$, there are two prefixes u and v such that u is a prefix of v, u and v end in the same state of the game, and $t(u) \leq t(v)$.*

Let us turn to the improvement of a strategy $\sigma(= \sigma_{j-1})$, aiming at bounding the wait times for the condition (Q_j, P_j). A path segment v, prefixed by w, is considered undesirable and will be removed if the following holds:

1. $P_j \notin \mathrm{Occ}(v)$
2. $t_j(w) \geq f_j^{-1}(M+1)$
3. $t(w) \leq t(wv)$

The value of the final bound — $f_j^{-1}(M) + d(n,k)$ — comes directly from the points 2 and 3. In order to guarantee that we remove only costly factors, we wait until the price from condition (Q_j, P_j) alone is more than the worst average penalty of the original strategy. And, in order to get a better strategy, we remove factors only when everything is worse at the end of the factor: Lemma 5 guarantees that we cannot wait more than $d(n,k)$ steps before finding one.

We describe now the improvement for pair (Q_j, P_j). To ease notation, we denote the initial strategy (σ_{j-1}) by σ, and the new one (σ_j) by τ. This strategy τ is defined as a strategy with memory: the memory states of τ are words over the states of $\mathcal{S}$ (more specifically, prefixes of plays consistent with σ). Let us explain the notation to avoid confusion between the four functions involved in the definition:

$\nu : (\mathcal{S}^*, \mathcal{S}_E) \to \mathcal{S}$. The "next-move" function. Notice that the first component of the argument is a memory state (*i.e.* a play consistent with σ), not the history of the play.

$\mu : (\mathcal{S}^*, \mathcal{S}_E) \to \mathcal{S}^*$. The "memory-update" function. Again, the first component of the argument is a memory state.

$\Omega : \mathcal{S}^* \to \mathcal{S}^*$. Ω extends μ to words: $\Omega(\epsilon) = \epsilon$, and $\Omega(ws) = \mu(w,s)$. Here, the argument is the history of the play (thus consistent with τ).

$\tau : \mathcal{S}^* \to \mathcal{S}^*$. The resulting strategy, in standard form. It can be defined from Ω: $\tau(ws) = \nu(\Omega(w), s)$. Again, the argument is the history of the play.

We define only the functions ν and μ, as the two others derive from them. Let y and s be the current memory and control states. We consider only the case where ys is a finite play of G consistent with σ, as the

update guarantees that this holds during the whole play. There are two cases, depending on whether the immediate penalty for condition (Q_j, P_j) in ys is greater than M or not. If it is not so, the update is done by adding the current state at the end of the memory: $\mu(s, y) = ys$. If it is Eve's turn to play, her move mimics σ's one: $\nu(s, y) = \sigma(ys)$.

On the other hand, if $t_j(ys) > f_j^{-1}(M)$, we first check if there is an "undesirable factor" to remove. We consider the game tree G^σ, rooted in ys and limited to the path segments up to the next visit to P_j. This tree is finite, as an infinite branch would be a loosing play – in the boolean sense. We consider the nodes of this tree such that the last visited state is s, and the wait times for *all* conditions (Q_i, P_i) are greater than the corresponding wait time in ys. Notice that ys itself is one of these nodes. Let zs be the path to a maximal node satisfying these conditions. Then, $\mu(y, s) := zs$. Likewise, if it is Eve's turn to play, she mimics the behavior of σ after the prefix zs: $\nu(y, s) := \sigma(zs)$.

Notice that in both cases, the contents of the memory remain to be plays consistent with σ. We claim that with such a definition, τ behaves as we wanted. We prove separately the Propositions 7, 8 and 9 that describe the different attributes we want to ensure in τ. Proposition 6 is an useful step in the proofs of Properties 7 and 8:

Proposition 6. *For any finite play w consistent with τ, we have $t(w) \leq t(\Omega(w))$.*

Proof. We prove by induction on the length of v that (2) holds for any finite play v consistent with τ:

$$t(v) \leq t(\Omega(v)) \tag{2}$$

By definition of Ω, $\Omega(\epsilon) = \epsilon$. Thus, (2) holds for ϵ. The induction step supposes that (2) holds for w, and proves that it also holds for ws, providing that ws is consistent with τ. From (2) applied to w and (1), we get:

$$t(ws) \leq t(\Omega(w)s) \tag{3}$$

We need now to check the two possible cases:

- $t_j(\Omega(w)) \leq f_j^{-1}(M)$**:** By definition of μ, $\Omega(ws) = \Omega(w)s$. Thus, the fact that (2) holds for ws follows from (3).
- $t_j(\Omega(w)) > f_j^{-1}(M)$**:** In this case, the update of memory guarantees that $t(\Omega(ws)) \geq t(\Omega(w)s)$. From this and (3), we derive that (2) holds for ws.

Thus, (2) holds for any finite play v consistent with τ, and Proposition 6 holds. □

We proceed now to the proof of Proposition 7.

Proposition 7. *If σ uniformly bounds the waiting time for pair (Q_i, P_i) to B, then so does τ.*

Proof. Let us suppose that σ bounds uniformly the waiting time for pair i to B. Let w be a finite play consistent with τ. The corresponding memory state $\Omega(w)$ is a finite prefix consistent with σ. Thus, $t_i(\Omega(w)) \leq B$. By Proposition 6, $t_i(w) \leq t_i(\Omega(w))$. Thus, $t_i(w) \leq B$. □

Proposition 8. *$v(\tau) \leq v(\sigma)$ (i.e., $v(\sigma_j) \leq v(\sigma_{j-1})$).*

Proof. In order to prove this proposition, we extend the function Ω to infinite plays consistent with τ: $\Omega(\rho)$ is the limit of the $(\Omega(w_i))_{i\in\mathbb{N}}$, for w_i the prefix of ρ of size i. This definition is sound, as $\Omega(w_{i+1})$ is always a strict suffix of $\Omega(w_i)$. Furthermore, as all the $\Omega(w_i)$ are consistent with σ, $\Omega(\rho)$ is an infinite play consistent with σ.

We show that for any ρ consistent with τ, $v(\rho) \leq v(\Omega(\rho))$. We do so by considering separately two sets of prefixes of $\Omega(\rho)$: w prefix of $\Omega(\rho)$ belongs to H if there is a prefix v of ρ such that $\Omega(v) = w$. Otherwise, it belongs to K.

We consider first the case of the prefixes in K. The corresponding conditions have been added to the memory during a "leap". By definition, all the positions in such a leap have a penalty for pair (Q_j, P_j) that is greater than $M + 1$. Thus, for all $w \in K, p(w) \geq M + 1$. As the mean value for all the positions is $v(\rho) \leq v(\sigma) \leq M$, it follows that the mean value for the prefixes in H is less than $v(\rho)$.

Let v be a prefix of ρ. By Proposition 6, we have $\forall i \in 1 \dots k, t_i(v) \leq t_i(\Omega(v))$, and thus $p(v) \leq p(\Omega(v))$. Thus, the average penalty for the prefixes of ρ is less or equal than the average penalty for the prefixes in H. This yields $v(\rho) \leq v(\Omega(\rho))$, and Proposition 8 follows. □

Proposition 9 expresses the main interest of strategy τ:

Proposition 9. *The strategy τ uniformly bounds the waiting time for (Q_j, P_j) to $f_j^{-1}(M) + d(n, k-1)$.*

Proof. This proof is done by contradiction and derives directly from the definition of the strategy and Lemma 5. We suppose that there is a path w consistent with σ_j such that $t_j(w) > f_j^{-1}(M) + n \cdot d(k)$. Let $w = uv$,

where $|v| = d(n,k)$. Thus, for any word between u and w, the wait time for pair (Q_j, P_j) is greater than $f_j^{-1}(M+1)$, and the memory is updated to a maximal branch in the tree. More formally, the definition of μ imposes that there cannot be two words x and y such that:

- $u < x < y < uv$
- $last(x) = last(y)$
- $\forall i \in 1 \dots k, t_i(\Omega(x)) < t_i(\Omega(y))$

This contradicts Lemma 5, which states that there cannot be such a sequence of length $d(n, k-1)$. Thus, Proposition 9 holds. □

It is now easy to complete the proof of Lemma 4. From a strategy σ, one can derive strategies $\sigma_1, \sigma_2, \dots, \sigma_k$ by successively applying the improvement scheme with respect to each pair (Q_j, P_j). Each time, a new pair is bounded (Proposition 9); the bounds hold through any improvement (Proposition 7); and the value of the strategy never increases (Proposition 8). Thus, the resulting strategy bounds each pair to the desired bound, and its value is less than the one of the original strategy. Lemma 4 follows.

5 From Request-Response Games to Mean-Payoff Games

By Lemma 4, we know that we can restrict our search for optimal strategies to strategies in which the wait times for each condition (Q_j, P_j) are bounded by

$$b_j := f_j^{-1}(M) + d(n, k-1).$$

In this section, we show how to interpret such strategies in a reduced mean-payoff game, and derive from this interpretation optimal strategies with finite memory. We assume here familiarity with mean-payoff games ([5,16]).

From a real-valued request-response game $\mathcal{G} = (\mathcal{S}, \mathcal{S}_E, \mathcal{S}_A, \mathcal{T})$ with conditions (Q_j, P_j) and increasing unbounded penalty functions f_j $(j = 1, \dots, k)$ we derive the mean-payoff game $\mathcal{G}^+ = (\mathcal{S}^+, \mathcal{S}_E^+, \mathcal{S}_A^+, \mathcal{T}^+)$ with a weight function w on the edges as follows:

- $\mathcal{S}^+ = \mathcal{S} \times \prod_{i \in 1 \dots k} [0, b_j]$,
- $\mathcal{S}_E^+$ and $\mathcal{S}_A^+$ are defined accordingly,
- $(q, n_1, \dots, n_k) \to (q', n_1', \dots, n_k') \in \mathcal{T}^+ \iff q \to q' \in \mathcal{T}$, and $\forall j$ from 1 to k
  - if $n_j = 0$ then $n_j' = 1$ if $q' \in Q_j \setminus P_j$ and $n_j' = 0$ otherwise,

  - if $n_j > 0$ then $n'_j = 0$ if $q' \in P_j$ and $n'_j = n_j + 1$ otherwise,
- $w((q, n_1, \ldots, n_k), (q', n'_1, \ldots, n'_k)) = \sum_{i=1}^{k} f_i(n_i)$.

This construction consists essentially in adding wait times (and penalties) to the graph, as long as they keep to the bounds we defined. Notice that the evolution of the "wait times" part depends directly on the first component. There is thus a natural bijection between the plays of the original game and the plays of the corresponding mean-payoff game.

We now apply a classical result on mean-payoff games:

Theorem 10 ([16]). *In a finite-state mean-payoff game (with designated initial vertex), there are (computable) optimal memoryless strategies for the two players.*

The counterpart of a memoryless strategy in the mean-payoff game is a finite-state strategy in the original request-response game:

- The set of memory states is $\prod_{i \in 1 \ldots k} [0, b_i]$.
- The strategy mimics the moves of the mean-payoff game:
  If $\sigma(q, m_1, \ldots, m_k) = (q', m'_1, \ldots, m'_k)$, then the next-move function μ and the memory update function ν are defined as follows:
  - $\mu(q, m_1, \ldots, m_k) = (m'_1, \ldots, m'_k)$
  - $\nu(q, (m_1, \ldots, m_k)) = q'$

  Theorem 2 follows.

6 Conclusion

We have studied request-response games in a setting where optimal strategies are to be constructed – in the present paper with respect to increasing and unbounded penalty functions. The main results says that if Eve (controller) wins a request-response game, then also an optimal winning strategy exists and can be effectively constructed as a finite-state strategy. This result fits into current research on valued games, such as studies on optimality of (mostly memoryless) strategies in stochastic and mean-payoff games [7,8,15]. In our case, however, we start from a discrete game, introduce a real-valued game, and after a reduction to mean-payoff games arrive again at a discrete entity (a finite-state controller) as optimal solution. Another methodological aspect is the necessity to find a balance between possibly conflicting aims when different request-response conditions (Q_j, P_j) have to be satisfied, a feature which often is only attached to multiplayer games.

Let us finally state some open problems:

1. The existence result of this paper involves very high complexity bounds (notably, in the present version of Dickson's Lemma). A more efficient synthesis procedure to optimal strategy synthesis should be found.
2. One may ask for more efficient solutions when the value of the constructed strategy only has to be an approximate of the optimal value.
3. The idea of approximation can also be invoked when bounded penalty functions are considered (see the example of Figure 1): The task of synthesis is then to find a strategy which realizes the limit of strategy values up to a certain factor.
4. Heuristic solutions should be developed, with an emphasis on games where the penalties of the request-response conditions are different.
5. More general ω-regular winning conditions might be considered ([2]). For example, "eager strategies" where Adam and Eve complete their cycles as fast as possible ([11]), or finitary games where requests must be answered in bounded time ([3,9]).

References

1. R. Alur, K. Etessami, S. La Torre, D. Peled, Parametric temporal logic for "model measuring", *ACM Trans. Comput. Logic* 2 (2001), 388-407.
2. J.R. Büchi, L.H. Landweber. Solving sequential conditions by finite-state strategies, *Trans. Amer. Math. Soc.* 138 (1969), 367-378.
3. K. Chatterjee, Th. A. Henzinger, Finitary Winning in omega-Regular Games, in: *Proc. 12th TACAS 2006* (H. Hermanns, J. Palsberg, eds.), Springer LNCS 3920, pp. 257–271.
4. L.E. Dickson, Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors, *Amer. J. Math.* 35, 413–422.
5. A. Ehrenfeucht and J. Mycielski, Positional strategies for mean payoff games", *Int. J. of Game Theory* 8, 109–113.
6. E. Grädel, W. Thomas, Th. Wilke (eds.), *Automata, Logics, and Infinite Games*, Springer LNCS 2500, Springer-Verlag 2002.
7. H. Gimbert, W. Zielonka, Games Where You Can Play Optimally Without Any Memory, in: *Proc. 16th CONCUR 2005* (M. Abadi, L. de Alfaro, eds.), Springer LNCS 3653, Springer-Verlag 2005.
8. H. Gimbert, W. Zielonka, Deterministic Priority Mean-Payoff Games as Limits of Discounted Games, in: *Proc. 33rd ICALP 2006* (M. Bugliesi, et al., eds.) Springer LNCS 4052, pp. 312–323.
9. F. Horn, Faster Algorithms for Finitary Games, in: *Proc. 13th TACAS 2007* (O. Grumberg, M. Huth, eds.), Springer LNCS 4424, pp. 472–484.
10. O. Kupferman, N. Piterman, M.Y. Vardi, From Liveness to Promptness, in: *Proc. 19th CAV 2007* (W. Damm, H. Hermanns, eds.) Springer LNCS 4590, pp. 406–419.
11. R. McNaughton, Playing infinite games in finite time. in: *A Half-Century of Automata Theory* (A. Salomaa, D. Wood, S. Yu, eds.) World Scientific 2000, pp. 73–91.

12. W. Thomas. On the synthesis of strategies in infinite games, *Proc. STACS 1995*, Springer LNCS 900 (1995), 1-13.
13. N. Wallmeier. *Strategien in unendlichen Spielen mit Liveness-Gewinnbedingungen: Syntheseverfahren, Optimierung und Implementierung*, Dissertation, RWTH Aachen 2007.
14. N. Wallmeier, P. Hütten, W. Thomas. Symbolic synthesis of finite-state controllers for request-response specifications, Proc. 8th CIAA, Springer LNCS 2759 (2003), 11-22.
15. W. Zielonka, An Invitation to Play. in: *Proc. 30th MFCS 2005* (J. Jedrzejowicz, A. Szepietowski, eds.), Springer LNCS 3618, Springer-Verlag, pp. 58–70.
16. U. Zwick, M. Paterson. The complexity of mean payoff games on graphs, *Theor. Comput. Sci.* 158 (1996), 343-259.