Improved Lower Bounds for Request-response and Finitary Streett Games

Krishnendu Chatterjee, Thomas A. Henzinger, and Florian Horn

IST Austria (Institute of Science and Technology Austria)

Am Campus 1

A-3400 Klosterneuburg

Technical Report No. IST-2009-0002

http://pub.ist.ac.at/Pubs/TechRpts/2009/IST-2009-0002.pdf

September 9, 2009

Improved Lower Bounds for Request-response and Finitary Streett Games

Krishnendu Chatterjee[†] Thomas A. Henzinger[†,‡] Florian Horn[†]

[†] IST Austria (Institute of Science and Technology Austria)

[‡] EPFL, Switzerland

{krish.chat,tah}@ist.ac.at, f.horn@cwi.nl

Abstract

We consider two-player games played on graphs with request-response and finitary Streett objectives. We show these games are PSPACE-hard, improving the previous known NP-hardness. We also improve the lower bounds on memory required by the winning strategies for the players.

1 Introduction

Games played on graphs are suitable models for multi-component systems: vertices represent states; edges represent transitions; players represent components; and objectives represent specifications. The specification of a component is typically given as an ω-regular condition [7], and the resulting ω-regular games have been used for solving control and verification problems (see, e.g., [3, 9, 10]).

Every ω-regular specification (indeed, every specification) can be decomposed into a safety part and a liveness part [1]. The safety part ensures that the component will not do anything "bad" (such as violate an invariant) within any finite number of transitions. The liveness part ensures that the component will do something "good" (such as proceed, or respond, or terminate) within some finite number of transitions. Liveness can be violated only in the limit, by infinite sequences of transitions, as no bound is stipulated on when the "good" thing must happen. This infinitary, classical formulation of liveness has both strengths and weaknesses. A main strength is robustness, in particular, independence from the chosen granularity of transitions. Another main strength is simplicity, allowing liveness to serve as an abstraction for complicated safety conditions. For example, a component may always respond in a number of transitions that depends, in some complicated manner, on the exact size of the stimulus. Yet for correctness, we may be interested only that the component will respond "eventually." However, these strengths also point to a weakness of the classical definition of liveness: it can be satisfied by components that in practice are quite unsatisfactory because no bound can be put on their response time. It is for this reason that alternative, stronger formulations of liveness have been proposed. One of these is *finitary* liveness [2, 5]: finitary liveness does not insist on response within a known bound b (i.e., every stimulus is followed by a response within b transitions), but on response within some unknown bound (i.e., there exists b such that every stimulus is followed by a response within b transitions). Note that in the finitary case, the bound b may be arbitrarily large, but the response time must not grow forever from one stimulus to the next. In this way, finitary liveness still maintains the robustness (independence of step granularity) and simplicity (abstraction of complicated safety) of traditional liveness, while removing unsatisfactory implementations.

The classical infinitary notion of fairness is given by the Streett objective: a Streett objective consists of a set of d pairs of requests and corresponding responses and the objective requires that every request that appears infinitely often must be responsed infinitely often. The two finitary formulation of fairness are the request-response objective and finitary Streett objective. The request-response objective requires that there is a bound b such that every request is responsed with in b steps; and the finitary Streett objective requires that there is a bound b such that in the limit every request is responsed with in b steps.

Previous results. Games with infinitary Streett objectives with d request-response pairs is coNP-complete [6]. The memory bound for winning strategies is as follows: there is an optimal (matching lower and upper) bound of $d!$ for the size of memory for the player with the Streett objective and the opposing player has memoryless winning strategy (a strategy that is independent of the history and depends on the current state). Games with request-response objective can be solved in EXPTIME [11]. The winning strategies for the player with request-response objective require a memory of size at least $2^{\lfloor d/3 \rfloor}$ and memory of size $d \cdot 2^d$ suffices for winning strategies; memory of size 2^d suffices for the winning strategies for the opposing player. Games with finitary Streett objectives can be solved in EXPTIME and is NP-hard [4]. The winning strategies for the player with finitary Streett objective require a memory of size at least $2^{\lfloor d/2 \rfloor}$ and memory of size $d \cdot 2^d$ suffices for winning strategies; the winning strategies for the opposing player require infinite memory in general.

Our results. In this work we present improved lower bounds for complexity and memory required by winning strategies. We first show that games with request-response and finitary Streett objectives are PSPACE-hard (improving the NP-hardness lower bound). We also study the complexity of one player game graphs: if there is only one player with request-response or finitary Streett objectives, then the problem is NP-complete; and if there is only the opposing player, then the problem can be solved in polynomial time. We improve the lower bound for memory required for winning strategies in games with request-response objectives: we show that in games with request-response objectives both players require at least $2^{\lfloor d/2 \rfloor}$ memory (improving the lower bound of $2^{\lfloor d/3 \rfloor}$ for the player with request-response objective, and no bound was known for the opposing player).

2 Request-response and Finitary Streett Games

In this section we first present the definitions of game graphs, plays, strategies, and then define the request-response and finitary Streett objectives.

2.1 Game graphs

Game graphs. A *game graph* $G = ((S, E), (S_1, S_2))$ consists of a directed graph (S, E) with a finite state space S and a set E of edges, and a partition (S_1, S_2) of the state space S into two sets. The states in S_1 are player 1 states, and the states in S_2 are player 2 states. For a state $s \in S$, we write $E(s) = \{t \in S \mid (s, t) \in E\}$ for the set of successor states of s. We assume that every state has at least one out-going edge, i.e., $E(s)$ is non-empty for all states $s \in S$. A game graph is a player-1 graph if $S_2 = \emptyset$, and is a player-2 graph if $S_1 = \emptyset$.

Plays. A game is played by two players: player 1 and player 2, who form an infinite path in the game graph by moving a token along edges. They start by placing the token on an initial state, and then they take moves indefinitely in the following way. If the token is on a state in S_1, then player 1 moves the token along one of the edges going out of the state. If the token is on a state in S_2, then player 2 does likewise. The result is an infinite path in the game graph; we refer to such infinite paths as plays. Formally, a *play* is an infinite sequence $\langle s_0, s_1, s_2, \ldots \rangle$ of states such that $(s_k, s_{k+1}) \in E$ for all $k \geq 0$. We write Π for the set of all plays.

Strategies. A strategy for a player is a recipe that specifies how to extend plays. Formally, a *strategy* σ for player 1 is a function σ: $S^* \cdot S_1 \to S$ that, given a finite sequence of states (representing the history of the play so far) which ends in a player 1 state, chooses the next state. The strategy must choose only available successors, i.e., for all $w \in S^*$ and $s \in S_1$, if $\sigma(w \cdot s) = t$, then $t \in E(s)$. The strategies for player 2 are defined analogously. We write Σ and Γ for the sets of all strategies for player 1 and player 2, respectively.

An equivalent definition of strategies is as follows. Let M be a set called *memory*. A strategy with memory can be described as a pair of functions: (a) a *memory-update* function σ_u: $S \times M \to M$ that, given the memory and the current state, updates the memory; and (b) a *next-state* function σ_n: $S \times M \to S$ that, given the memory and the current state, specifies the successor state. The strategy is *finite-memory* if the memory M is finite and for a finite-memory strategy σ we write $|\sigma|$ to denote the size of its memory, i.e., $|M|$. The strategy is *memoryless* if the memory M is a singleton set. The memoryless strategies do not depend on the history of a play, but only on the current state. Each memoryless strategy for player 1 can be specified as a function σ: $S_1 \to S$ such that $\sigma(s) \in E(s)$ for all $s \in S_1$, and analogously for memoryless player 2 strategies. Given a starting state $s \in S$, a strategy $\sigma \in \Sigma$ for player 1, and a strategy $\tau \in \Gamma$ for player 2, there is a unique play, denoted $\pi(s, \sigma, \tau) = \langle s_0, s_1, s_2, \ldots \rangle$, which is defined as follows: $s_0 = s$ and for all $k \geq 0$, if $s_k \in S_1$, then $\sigma(s_0, s_1, \ldots, s_k) = s_{k+1}$, and if $s_k \in S_2$, then $\tau(s_0, s_1, \ldots, s_k) = s_{k+1}$.

2.2 Request-response and Finitary Streett objectives

An objective $\Psi \subseteq \Pi$ for player 1 in a game graph is a subset of plays. We will consider *request-response* and *finitary Streett* objectives, and to define the objectives we need to define the notion of distance sequence.

Distance sequences for Streett objectives. Let $P = \{(\mathsf{Rq}_1, \mathsf{Rp}_1), (\mathsf{Rq}_2, \mathsf{Rp}_2), \ldots, (\mathsf{Rq}_d, \mathsf{Rp}_d)\}$ be a set of d requests and the corresponding responses; for all $1 \le i \le d$ we have $\mathsf{Rq}_i \subseteq S$ and $\mathsf{Rp}_i \subseteq S$. Given a play $\pi = \langle s_0, s_1, s_2, \ldots \rangle$ and P, the d sequences of distances $dist_k^j(\pi, P)$, for all $k \ge 0$ and $1 \le j \le d$, are defined as follows:

$$dist_k^j(\pi, P) = \begin{cases} 0 & \text{if } s_k \notin \mathsf{Rq}_j; \\ \inf\{k' - k \mid k' \ge k, s_{k'} \in \mathsf{Rp}_j\} & \text{if } s_k \in \mathsf{Rq}_j. \end{cases}$$

Let $dist_k(\pi, P) = \max\{dist_k^j(\pi, P) \mid 1 \le j \le d\}$ for all $k \ge 0$.

Request-response objective. The request-response objective requires the distance sequence to be bounded. Formally, given $P = \{(\mathsf{Rq}_1, \mathsf{Rp}_1), \ldots, (\mathsf{Rq}_d, \mathsf{Rp}_d)\}$, the request-response objective is defined as follows:

$$\begin{aligned} \mathrm{ReqRep}(P) &= \{\pi \in \Pi \mid \exists j \in \mathbb{N}.\ \forall k \ge 0.\ dist_k(\pi, P) \le j\} \\ &= \{\pi \in \Pi \mid \sup\{dist_k(\pi, P) \mid k \ge 0\} < \infty\} \\ &= \{\langle s_0, s_1, s_2, \ldots \rangle \in \Pi \mid \exists j \in \mathbb{N}. \forall i \ge 0.\ \text{if } s_i \in \mathsf{Rq}_\ell,\ \text{for } 1 \le \ell \le d, \\ &\qquad \text{then exists } i \le k \le i + j \text{ such that } s_k \in \mathsf{Rp}_\ell\} \end{aligned}$$

In other words the request-response objective requires that every request is responsed with in a bounded number (i.e., within the number j) of steps. We use the following notations for the complementary objective: $\mathrm{coReqRep}(P) = \Pi \setminus \mathrm{ReqRep}(P)$.

Finitary Streett objectives. The finitary Streett objective $\mathrm{finStreett}(P)$ for a set P of request-response pairs requires that the distance sequence be bounded in the limit, i.e., the winning plays are $\mathrm{finStreett}(P) = \{\pi \in \Pi \mid \limsup_{k \to \infty} dist_k(\pi, P) < \infty\}$. We use the following notations for the complementary objective: $\mathrm{cofinStreett}(P) = \Pi \setminus \mathrm{finStreett}(P)$.

Winning. Given an objective $\Psi \subseteq \Pi$ for player 1, a strategy $\sigma \in \Sigma$ is a *winning strategy* for player 1 from a set $U \subseteq S$ of states if for all player 2 strategies $\tau \in \Gamma$ and all states $s \in U$, the play $\pi(s, \sigma, \tau)$ is winning, i.e., $\pi(s, \sigma, \tau) \in \Psi$. The winning strategies for player 2 are defined analogously. A state $s \in S$ is winning for player 1 with respect to the objective Ψ if player 1 has a winning strategy from $\{s\}$. Formally, the set of *winning states* for player 1 with respect to the objective Ψ is $W_1(\Psi) = \{s \in S \mid \exists \sigma \in \Sigma.\ \forall \tau \in \Gamma.\ \pi(s, \sigma, \tau) \in \Psi\}$. Analogously, the set of winning states for player 2 with respect to an objective $\Psi \subseteq \Pi$ is $W_2(\Psi) = \{s \in S \mid \exists \tau \in \Gamma.\ \forall \sigma \in \Sigma.\ \pi(s, \sigma, \tau) \in \Psi\}$. We say that there exists a (memoryless; finite-memory) winning strategy for player 1 with respect to the objective Ψ if there exists such a strategy from the set $W_1(\Psi)$; and similarly for player 2.

Remark 2.1 *The request-response objectives were introduced in [11], and the following alternative definition was used:*

$$\begin{aligned} \widehat{\mathrm{ReqRep}}(P) &= \{\langle s_0, s_1, s_2, \ldots \rangle \in \Pi \mid \exists j \in \mathbb{N}. \forall i \ge 0.\ \textit{if } s_i \in \mathsf{Rq}_\ell,\ \textit{for } 1 \le \ell \le d, \\ &\qquad \textit{then exists } k \ge i.\ \textit{such that } s_k \in \mathsf{Rp}_\ell\}. \end{aligned}$$

From the result of existence of finite-memory winning strategies for $\widehat{\mathrm{ReqRep}}(P)$ *[11], it follows that for all game graphs we have* $W_1(\mathrm{ReqRep}(P)) = W_1(\widehat{\mathrm{ReqRep}}(P))$.

3 Improved Complexity Bounds

In this section we first present improved complexity lower bound for request-response and finitary Streett games, and then present the complexity results for game graphs with only one player.

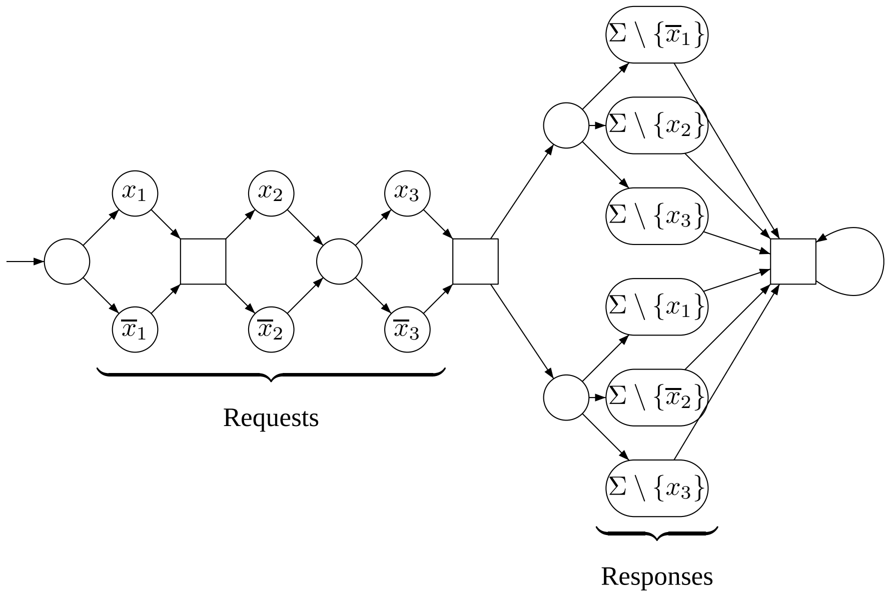

Figure 1: Request-response games are PSPACE-hard. The game for the QBF $\exists x_1 \forall x_2 \exists x_3.(x_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$.

3.1 Improved complexity lower bound for games

It was shown in [11] that request-response games can be solved in EXPTIME, and it was shown in [4] that finitary Streett games can be solved in EXPTIME. It was also shown in [4] that finitary Streett games are NP-hard. Below we improve the lower bound showing that the problems are PSPACE-hard.

Theorem 3.1 *Let G be a game graph with a request-response or a finitary Streett objective Ψ. Given a state s, the decision problem of whether $s \in W_1(\Psi)$ is PSPACE-hard.*

Proof. We present a reduction from the QBF (quantified boolean formula). Consider a QBF

$$\Phi = \exists x_1.\forall x_2.\exists x_3 \ldots \forall x_n.\, c_1 \wedge c_2 \wedge \ldots \wedge c_k;$$

over the set $X = \{x_1, x_2, \ldots, x_n\}$ of variables and the set $C = \{c_1, c_2, \ldots, c_k\}$ of clauses where each clause c_i consists of exactly three literals c_{i1}, c_{i2}, and c_{i3}; (a literal is a variable x_i or its complement $\overline{x}_i$). Given a QBF Φ, the problem of deciding the truth of Φ is PSPACE-complete [8]. We present a reduction of deciding the truth of QBFs to determining winner in a game graph with request-response objective. Given a QBF Φ we construct a game graph G_Φ as follows:

1. *(State space).* The set of states S is as follows

$$X \cup \{x_{iq}, \overline{x}_{iq} \mid i \in \{1,2,\ldots,n\}\} \cup C \cup \{c_{ij} \mid i \in \{1,2,\ldots,k\}, j \in \{1,2,3\}\} \cup \{x_{n+1}\} \cup \{c_{k+1}\}.$$

   There is a state for every variable $x_i \in X$ and there is a state for every clause $c_i \in C$, and there are two additional states x_{n+1} and c_{k+1}. For the literal x_i we have the state x_{iq} and for the literal $\overline{x}_i$ we have the state $\overline{x}_{iq}$. For a clause c_i we have states for the literals c_{i1}, c_{i2}, and c_{i3} that appear in c_i.

2. *(State space partition).* For a variable x_i, if it is universally quantified in Φ, then the state x_i belongs to player 2. The state x_{n+1} belongs to player 2. All other states belong to player 1.

3. *(Edges).* The set of edges E is as follows

$$\begin{aligned}&\{(x_i, x_{iq}), (x_i, \overline{x}_{iq}) \mid i \in \{1,2,\ldots,n\}\} \cup \{(x_{iq}, x_{i+1}), (\overline{x}_{iq}, x_{i+1}) \mid i \in \{1,2,\ldots,n\}\}\\ \cup&\{(c_i, c_{ij}) \mid i \in \{1,2,\ldots,k\}, j \in \{1,2,3\}\} \cup \{(c_{ij}, c_{k+1}) \mid i \in \{1,2,\ldots,k\}, j \in \{1,2,3\}\}\\ \cup&\{(x_{n+1}, c_i) \mid i \in \{1,2,\ldots,k\}\} \cup \{(c_{k+1}, c_{k+1})\}.\end{aligned}$$

For an existentially quantified variable x_i, player 1 can choose between two successors x_{iq} and $\overline{x}_{iq}$ that corresponds to choosing either x_i as true or x_i as false. For universally quantified variable, player 2 has similar choices. The next state of x_{iq} and $\overline{x}_{iq}$ is the state x_{i+1}. From the state x_{n+1}, player 2 can choose any clause c_i; and in a clause c_i, player 1 has the choice of the literals c_{ij} that appear in c_i. From a state c_{ij} the next state is c_{k+1} and the state c_{k+1} is an *absorbing state* (a state with only self-loop as the outgoing transition).

4. *(Request-response labeling).* The request-response labeling is as follows: (a) there is a request-response pair for every literal; (b) a state x_{iq} is labeled with the request x_i and a state $\overline{x}_{iq}$ is labeled with the request $\overline{x}_i$; and (c) for a state c_{ij}, if c_{ij} is the literal x_ℓ, then c_{ij} is labeled with all responses other than the response for the complement of x_ℓ (formally, let $\overline{X}$ be the set of all complementary variables of X and let $\Sigma = X \cup \overline{X}$, then c_{ij} is labeled by all responses $\Sigma \setminus \{\overline{x}_\ell\}$); and if c_{ij} is the literal $\overline{x}_\ell$, then it is labeled by all responses $\Sigma \setminus \{x_\ell\}$.

Figure 1 presents a pictorial description of the reduction on an example. We now present the two directions of the correctness argument.

1. *Truth implies winning.* We first show that if Φ is true, then player 1 has a winning strategy from the state x_1. If Φ is true, then there is a witness assignment function A that satisfies the following condition: given an existentially quantified variable x_i and a truth assignment to all variables x_j before x_i (i.e., $j < i$), the assignment function assigns a truth value x_i, and the assignment function ensures that against all truth value assignments to the universally quantified variables, all the clauses $c_i \in C$ are satisfied. A witness strategy σ for player 1 to ensure winning for the request-response objective is as follows: for an existentially quantified variable x_i and a history w that leads to x_i, if the assignment function A assigns true to x_i given the truth value assignment that corresponds to w, then the strategy σ chooses x_{iq}, otherwise chooses $\overline{x}_{iq}$. Consider a strategy τ for player 2: let w be the path that lead to x_{n+1} given σ and τ, and then let the choice of player 2 at state x_{n+1} be a clause c_i. Since A is a witness truth assignment, it follows that if we consider the truth assignment to universally quantified variables that corresponds to the choices in w, then clause c_i must be satisfied. Since σ is constructed from A it follows that there must be a literal c_{ij} in c_i such that the complement variable of c_{ij} was not chosen in the path given σ and τ, and hence by choosing the successor state c_{ij} from c_i player 1 ensures that the request-response objective is satisfied.

2. *Winning implies truth.* We now show that if there is a winning strategy for player 1 from x_1, then Φ is true. Consider a witness winning strategy σ for player 1. A witness truth assignment function A to show Φ is true is constructed as follows. Consider an existentially quantified variable x_i, and a truth assignment to all variables x_j before x_i (i.e., $j < i$). Let w be the history in the game graph that corresponds to the given truth assignment values that lead to x_i; if σ chooses x_{iq}, then x_i is set to true, otherwise to false. Consider a truth assignment to the universally quantified variables, and a clause c_i, and let us consider the path in the game graph that corresponds to all the truth assignments and chooses the clause c_i. Since σ is winning it follows that there is a literal c_{ij} in c_i that is a response to all the requests of the path (i.e., the complement of the variable of c_{ij} was not chosen in the path), and hence it follows that the assignment function ensures that clause c_i is satisfied since c_{ij} is set to true.

The result follows for request-response objective. The result for finitary Streett objective follows from a similar construction. The above construction is modified as follows: the state c_{k+1} is made a player 2 state, and along with the self-loop, an edge is added to the starting state x_1. If Φ is true, then a strategy σ constructed from the witness assignment function A ensures that every request is responsed with in $2n + 2$-steps. If Φ is not true, then there is a strategy τ for player 2 such that against all player 1 strategies in the path from x_1 to c_{k+1} there is a request that is not responsed: the strategy for player 2 to ensure that the finitary Streett objective is violated plays τ in rounds and in round i it stays in the self-loop at c_{k+1} for i-steps, then goes to round $i + 1$ choosing the edge to state x_1 and repeats the strategy τ. This shows that player 1 has a winning strategy from x_1 iff Φ is true. Hence the hardness result follows for finitary Streett objectives, and we have the desired result. ∎

3.2 Complexity bound for player-1 and player-2 graphs

We now present the complexity bounds for player-1 and player-2 graphs with request-response and finitary Streett objectives.

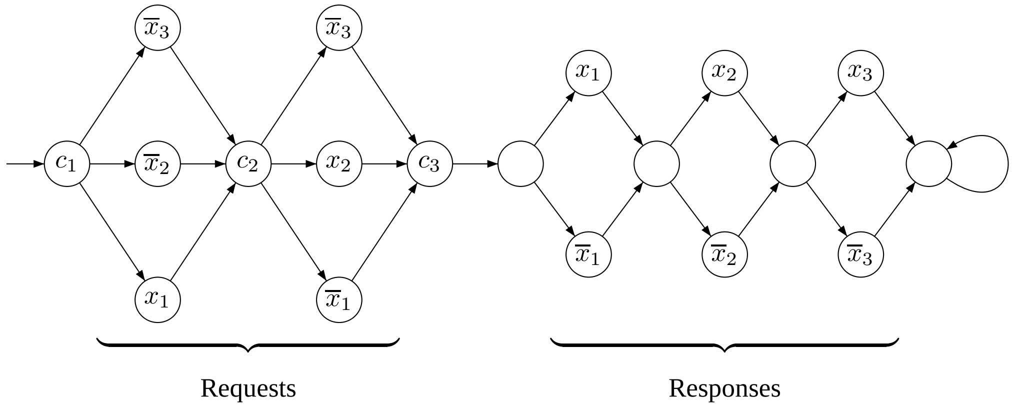

Figure 2: Player-1 graphs with request-response objective is NP-hard: the graph for the 3SAT formula $(x_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$

Theorem 3.2 *Let G be a player-1 graph with a request-response or a finitary Streett objective Ψ. Given a state s, the decision problem of whether $s \in W_1(\Psi)$ is NP-complete.*

Proof. The NP-hardness for finitary Streett objective was shown in [4]. The correctness argument for inclusion in NP is simple and as follows: if the graph has n states, and the request-response or the finitary Streett objective consists of d-pairs, then there is a path of length at most $n \cdot d$ followed by visiting infinitely often every state of a subset C of states that is strongly connected (and every state in C is visited within $|C|^2$ steps). The guess of the path of length $n \cdot d$ and the subset C of states is polynomial and can be verified in polynomial time. To complete the proof we show the NP-hardness for request-response objective. We present a reduction from the 3-SAT problem. Consider a 3-SAT formula

$$\Phi = c_1 \wedge c_2 \ldots \wedge c_k;$$

over the set X of variables with the such that every clause c_i has exactly three literals c_{ij}, $j \in \{1,2,3\}$. We denote by C the set $\{c_1, c_2, \ldots, c_k\}$ of clauses. Given a 3-SAT formula Φ we construct a player-1 graph G_Φ as follows:

1. *(State space).* The set of states S is as follows

$$X \cup \{x_{iq}, \overline{x}_{iq} \mid i \in \{1,2,\ldots,n\}\} \cup C \cup \{c_{ij} \mid i \in \{1,2,\ldots,k\}, j \in \{1,2,3\}\} \cup \{x_{n+1}\} \cup \{c_{k+1}\}.$$

   There is a state for every variable $x_i \in X$ and there is a state for every clause $c_i \in C$, and there are two additional states x_{n+1} and c_{k+1}. For the literal x_i we have the state x_{iq} and for the literal $\overline{x}_i$ we have the state $\overline{x}_{iq}$. For a clause c_i we have states for the literals c_{i1}, c_{i2}, and c_{i3} that appear in c_i.

2. *(Edges).* The set of edges E is as follows

$$\begin{array}{l} \{(c_i, c_{ij}) \mid i \in \{1,2,\ldots,k\}, j \in \{1,2,3\}\} \cup \{(c_{ij}, c_{i+1}) \mid i \in \{1,2,\ldots,k\}, j \in \{1,2,3\}\} \\ \cup\{(x_i, x_{iq}), (x_i, \overline{x}_{iq}) \mid i \in \{1,2,\ldots,n\}\} \cup \{(x_{iq}, x_{i+1}), (\overline{x}_{iq}, x_{i+1}) \mid i \in \{1,2,\ldots,n\}\} \\ \cup\{(c_{k+1}, x_1) \mid i \in \{1,2,\ldots,k\}\} \cup \{(x_{n+1}, x_{n+1})\}. \end{array}$$

   For a clause c_i, player 1 has the choice of the literals c_{ij} that appear in c_i, and the next state of a state c_{ij} is c_{i+1}. For a state x_i, player 1 can choose between two successors x_{iq} and $\overline{x}_{iq}$ that corresponds to choosing either x_i as true or x_i as false. The next state of x_{iq} and $\overline{x}_{iq}$ is the state x_{i+1}. The next state of c_{k+1} is the state x_1 and the state x_{n+1} is absorbing.

3. *(Request-response labeling).* The request-response labeling is as follows: (a) there is a request-response pair for every literal; (b) a state c_{ij} is labeled by the request of the literal that it represents; and (c) a state x_{iq} is labeled by the response literal, i.e., x_{iq} is labeled with the response for literal x_i, and a state $\overline{x}_{iq}$ is labeled with the response for literal $\overline{x}_i$.

Figure 2 gives a pictorial description on an example. If Φ is true, then there is a truth assignment A to the variables such that every clause is satisfied (i.e., for every clause c_i there is a choice of literal c_{ij} in c_i such that c_{ij} is set as true by A). The strategy to choose the c_{ij} and the successor at x_i as given by the truth assignment A ensures that the request-response objective is satisfied. If player 1 can satisfy the request-response objective, then consider the choice of literal c_{ij} at states c_i, and the choice of successor at states x_i; the corresponding truth assignment and the choice of literal is a witness that Φ can be satisfied. This completes the proof and the result follows. ∎

Theorem 3.3 *Let G be a player-2 graph with a request-response or a finitary Streett objective Ψ. Given a state s, the decision problem of whether $s \in W_1(\Psi)$ can be solved in PTIME.*

Proof. We present a polynomial time algorithm to solve player-2 graphs with request-response objectives. An algorithm was proposed in [4] to solve finitary Streett games that required $O(n)$ iterations of an algorithm that solves request-response objectives. Hence the result would follow. To present the result we need two notations: for a set $U \subseteq S$ we denote by $\mathrm{Safe}(U) = \{\langle s_0, s_1, s_2, \ldots\rangle \mid \forall i \geq 0.\ s_i \in U\}$ the set of paths that avoids visiting states outside U; and by $\mathrm{Reach}(U) = \{\langle s_0, s_1, s_2, \ldots\rangle \mid \exists i \geq 0.\ s_i \in U\}$ the set of paths that visits a state in U. The polynomial time algorithm to solve request-response objectives in player-2 graphs is as follows:

1. for $1 \leq i \leq d$, let $X_i = \mathsf{Rq}_i \cap W_2(\mathrm{Safe}(S \setminus \mathsf{Rp}_i))$ be the set of states that correspond to a request Rq_i and player 2 can ensure to stay safe avoiding any state of the corresponding response Rp_i. Hence any state in X_i is loosing for the request-response objective.

2. Let $X = \bigcup X_i$, and let $Z = W_2(\mathrm{Reach}(X))$. From Z player 2 can play a strategy to reach X, and if a state in X_i is reached, then player 2 can play the strategy to avoid Rp_i, and ensure that the request-response objective is violated.

Hence we have $Z \subseteq W_2(\Pi \setminus \Psi)$. Let $\overline{Z} = S \setminus Z$. From every state $s \in \overline{Z}$, for all states t, if $(s,t) \in E$, then $t \in \overline{Z}$; as otherwise s could reach Z, and from Z the set X can be reached. Moreover, from every state in $\overline{Z}$ for a state $s \in \mathsf{Rq}_i$, for any strategy for player 2 a state in Rp_i is reached and within $|S|$-steps (as otherwise player 2 could have ensured $\mathrm{Safe}(S \setminus \mathsf{Rq}_i)$). Hence it follows that $\overline{Z} \subseteq W_1(\Psi)$, and thus we have $\overline{Z} = W_1(\Psi)$. Since the safety and reachability objectives can be solved in polynomial time in graphs, the desired result follows. ∎

4 Improved Lower Bound for Memory

In this section we present improved lower bound for memory required by winning strategies for both players. Given a request-response objective with d-pairs, a lower bound of $2^{\lfloor d/3 \rfloor}$ memory requirement for player 1 strategies was shown in [11], and no lower bound was presented for player 2. We now improve the bounds showing that in general winning strategies may require at least $2^{\lfloor d/2 \rfloor}$ memory for both players.

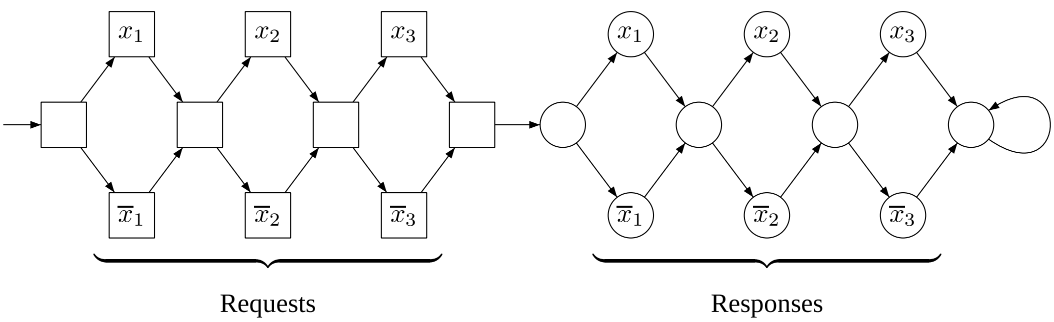

Figure 3: Player 1 needs $2^{\lfloor \frac{k}{2} \rfloor}$ memory in games with request-response objectives

Theorem 4.1 *Given a game graph with a request-response objective with d-pairs, in general winning strategies for player 1 and player 2 require at least $2^{\lfloor d/2 \rfloor}$ memory*

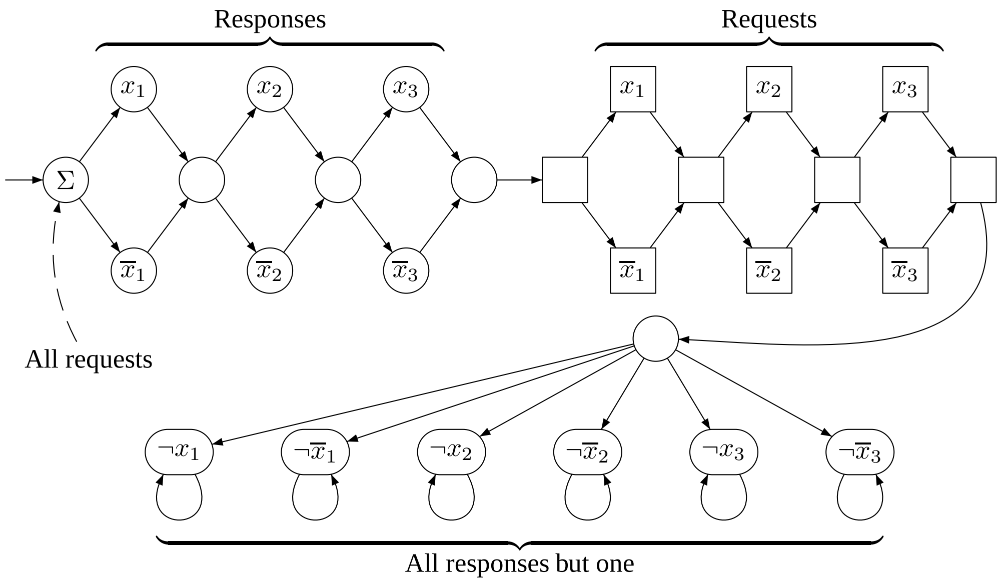

Figure 4: Player 2 needs $2^{\lfloor \frac{k}{2} \rfloor}$ memory in games with request-response objectives

Proof. We first present the family of examples for player 1, and then present the result for player 2. Let $X = \{x_1, x_2, \ldots, x_d\}$, $\overline{X} = \{\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_d\}$, $\widehat{X} = \{\widehat{x}_1, \widehat{x}_2, \ldots, \widehat{x}_{d+1}\}$, $Y = \{y_1, y_2, \ldots, y_d\}$, $\overline{Y} = \{\overline{y}_1, \overline{y}_2, \ldots, \overline{y}_d\}$, and $\widehat{Y} = \{\widehat{y}_1, \widehat{y}_2, \ldots, \widehat{y}_{d+1}\}$.

1. *Lower-bound for player 1.* Consider a game graph G as follows:

   (a) *State space and partition.* The state space is $X \cup \overline{X} \cup \widehat{X} \cup Y \cup \overline{Y} \cup \widehat{Y}$. The state space partition is as follows: $S_1 = Y \cup \overline{Y} \cup \widehat{Y}$ and $S_2 = X \cup \overline{X} \cup \widehat{X}$. We will consider x_1 as the starting state.

   (b) *Request-response labeling.* Every state x_i is labeled by the request x_i, every state $\overline{x}_i$ is labeled by the request $\overline{x}_i$; every state y_i is labeled by the response for x_i and every state $\overline{y}_i$ is labeled by the response for $\overline{x}_i$.

   (c) *Edges.* The set of edges is as follows:

   $$\begin{array}{l} \{(\widehat{x}_i, x_i), (\widehat{x}_i, \overline{x}_i) \mid 1 \le i \le d\} \cup \{(x_i, \widehat{x}_{i+1}), (\overline{x}_i, \widehat{x}_{i+1}) \mid 1 \le i \le d\} \\ \cup \{(\widehat{y}_i, y_i), (\widehat{y}_i, \overline{y}_i) \mid 1 \le i \le d\} \cup \{(y_i, \widehat{y}_{i+1}), (\overline{y}_i, \widehat{y}_{i+1}) \mid 1 \le i \le d\} \\ \cup \{(\widehat{x}_{d+1}, \widehat{y}_1), (\widehat{x}_{d+1}, \widehat{x}_{d+1})\} \end{array}$$

   At every state $\widehat{x}_i$ player 2 chooses between x_i and $\overline{x}_i$ and then proceeds to $\widehat{x}_{i+1}$. At every state $\widehat{y}_i$ player 1 chooses between y_i and $\overline{y}_i$ and then proceeds to $\widehat{y}_{i+1}$. The next state of $\widehat{x}_{d+1}$ is $\widehat{y}_1$, and $\widehat{y}_{d+1}$ is an absorbing state.

   See Figure 3 for a pictorial description. In other words, player 2 initially chooses a sequence of requests of length d such that the i-th request is either x_i or $\overline{x}_i$. A winning strategy for player 1 matches the sequence by the corresponding responses. Consider a strategy for player 1 that uses less than 2^d memory. Then there must exist two sequences of requests for which player 1 plays in the same way, and hence for one of the sequence there is a request that is not answered. Hence any strategy with less than 2^d memory cannot be winning. Hence we have a game graph with $2d$ request-response pairs such that every winning strategy for player 1 requires 2^d memory.

2. *Lower-bound for player 2.* Consider a game graph G as follows:

   (a) *State space and partition.* Let $\widetilde{X} = \{\widetilde{x}_1, \widetilde{x}_2, \ldots, \widetilde{x}_{2d}\}$. The state space is $X \cup \overline{X} \cup \widehat{X} \cup Y \cup \overline{Y} \cup \widehat{Y} \cup \widetilde{X}$. The state space partition is as follows: $S_2 = X \cup \overline{X} \cup (\widehat{X} \setminus \{\widehat{x}_{d+1}\}) \cup \widetilde{X}$ and $S_1 = Y \cup \overline{Y} \cup \widehat{Y} \cup \{\widehat{x}_{d+1}\}$. We will consider y_1 as the starting state.

(b) *Request-response labeling.* The initial state y_1 is labeled with all requests; every state y_i is labeled by the response for x_i and every state $\overline{y}_i$ is labeled by the response for $\overline{x}_i$; every state x_i is labeled by the request x_i, every state $\overline{x}_i$ is labeled by the request $\overline{x}_i$; and for $1 \leq i \leq d$, a state $\widetilde{x}_{2i-1}$ is labeled with every response other than $\overline{x}_i$, and a state $\widetilde{x}_{2i}$ is labeled with every response other than x_i.

(c) *Edges.* The set of edges is as follows:

$$\begin{array}{l}\{(\widehat{y}_i, y_i), (\widehat{y}_i, \overline{y}_i) \mid 1 \leq i \leq d\} \cup \{(y_i, \widehat{y}_{i+1}), (\overline{y}_i, \widehat{y}_{i+1}) \mid 1 \leq i \leq d\} \\ \cup \{(\widehat{x}_i, x_i), (\widehat{x}_i, \overline{x}_i) \mid 1 \leq i \leq d\} \cup \{(x_i, \widehat{x}_{i+1}), (\overline{x}_i, \widehat{x}_{i+1}) \mid 1 \leq i \leq d\} \\ \cup \{(\widehat{y}_{d+1}, \widehat{x}_1)\} \cup \{(\widehat{x}_{d+1}, \widetilde{x}_i) \mid 1 \leq i \leq 2d\} \cup \{(\widetilde{x}_i, \widetilde{x}_i) \mid 1 \leq i \leq 2d\}\end{array}$$

At every state $\widehat{x}_i$ player 2 chooses between x_i and $\overline{x}_i$ and then proceeds to $\widehat{x}_{i+1}$. At every state $\widehat{y}_i$ player 1 chooses between y_i and $\overline{y}_i$ and then proceeds to $\widehat{y}_{i+1}$. The next state of $\widehat{y}_{d+1}$ is $\widehat{x}_1$, from the state $\widehat{x}_{d+1}$ player 1 can choose any state in $\widetilde{X}$, and every state in $\widetilde{X}$ is absorbing.

See Figure 4 for a pictorial description. In other words, the game starts by generating all requests at y_1, and then player 1 answers by a sequence of d responses. Then player 2 can again generate a sequence of d requests, and then player 1 can choose to answer all but one request. A winning strategy for player 2 exactly generates the sequence of requests that have been previously answered by player 1. Thus in the end player 1 must answer all requests, but can answer all but one requests, and hence player 2 wins. If player 2 plays a strategy that uses less than 2^d memory, then there exist two sequences of responses for player 1 for which player 2 plays in a similar fashion. In one of the sequences there is one response that player 1 has previously answered and player 2 have not generated the corresponding request. Hence player 1 can choose to answer all but one request and satisfy the request-response objective. Hence it follows any winning strategy for player 2 requires 2^d memory.

The desired result follows. ∎

Concluding remarks. In this work we improve the lower bound for complexity for games with request-response and finitary Streett objectives from NP-hardness to PSPACE-hardness. The upper bound is EXPTIME, and whether these games can be solved in PSPACE or whether they are EXPTIME-hard remain open. We also improve the lower bound on memory required for the winning strategies for both players. However an optimal bound (matching upper and lower bound) for memory for the winning strategies is still open.

References

[1] B. Alpern and F.B. Schneider. Defining liveness. *IPL*, 21:181–185, 1985.

[2] R. Alur and T.A. Henzinger. Finitary fairness. In *LICS*, pages 52–61. IEEE, 1994.

[3] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *JACM*, 49:672–713, 2002.

[4] K. Chatterjee, T.A. Henzinger, and F. Horn. Finitary winning in ω-regular games. *ACM ToCL*.

[5] N. Dershowitz, D.N. Jayasimha, and S. Park. Bounded fairness. In *Verification: Theory and Practice*, pages 304–317. LNCS 2772, Springer, 2003.

[6] E.A. Emerson and C. Jutla. The complexity of tree automata and logics of programs. In *Foundations of Computer Science*, pages 328–337. IEEE Computer Society, 1988.

[7] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer, 1992.

[8] C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1993.

[9] A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *POPL*, pages 179–190. ACM, 1989.

[10] P.J. Ramadge and W.M. Wonham. Supervisory control of a class of discrete-event processes. *SIAM Journal of Control and Optimization*, 25:206–230, 1987.

[11] N. Wallmeier, P. Hutten, and W. Thomas. Symbolic synthesis of finite-state controllers for request-response specifications. In *CIAA*, pages 11–22. LNCS 2759, Springer, 2003.