Playing Against Opponents With Limited Memory

Dhananjay Raju  
The University of Texas at Austin, USA  
draju@cs.utexas.edu

Rüdiger Ehlers  
Clausthal University of Technology, Germany  
ruediger.ehlers@tu-clausthal.de

Ufuk Topcu  
The University of Texas at Austin, USA  
utopcu@utexas.edu

--- Abstract ---

We study partial-information two-player turn-based games on graphs with omega-regular objectives, when the partial-information player has limited memory. Such games are a natural formalization for reactive synthesis when the environment player is not genuinely adversarial to the system player. The environment player has goals of its own, but the exact goal of the environment player is unknown to the system player. We prove that the problem of determining the existence of a winning strategy for the system player is PSPACE-hard for reachability, safety, and parity objectives. Moreover, when the environment player is memoryless, the problem is PSPACE-complete. However, it is simpler to decide if the environment player has a winning strategy; it is only NP-complete. Additionally, we construct a game where the partial-information player needs at least $O(\sqrt{n})$ bits of memory to retain winning strategies in a game of size $O(n)$.

2012 ACM Subject Classification Theory of computation → Formal languages and automata theory; Theory of computation → Complexity theory and logic

Keywords and phrases Graph Games, Partial Information, Synthesis, Limited Memory

Category Submitted to ICALP 2020
1 Introduction

Reactive synthesis is the process of computing correct-by-construction implementations for reactive systems from their specifications. The attractiveness of leaving the implementation of a system to a synthesis algorithm led to the application of reactive synthesis in a diverse set of domains such as hardware circuits, graphical user interfaces, and high-level robotic mission planning [10, 7, 2].

The core difficulty in reactive synthesis is the computation of how parts of a specification can be satisfied without the knowledge of the future input to the system. The reactive synthesis problem is typically reduced to solving a game between two players: a player corresponding to the system and a player corresponding to the environment. The specification in the synthesis problem is encoded through the winning condition for the system player. There exists a solution to the synthesis problem if and only if the system player has a strategy to win this game.

Most specifications are realizable only under certain assumptions on the behavior of the environment. These assumptions have to also be captured in the specification. For instance, an elevator can only move between floors if its doors are closed. If the doors are blocked all of the time, a specification to eventually reach another floor cannot be fulfilled. Correspondingly, making assumptions about the environment player into the games constructed for synthesis is necessary to solve the synthesis problem. In practice, specifying these assumptions precisely is difficult. Furthermore, requiring the system player to satisfy the specification only if the assumptions hold creates an incentive for the system player to violate the assumptions actively. Raskin et al. [12] have for instance addressed this problem by computing strategy profiles for the environment and system players so that none of them has an incentive to deviate from their strategy.

Synthesis based on strategy profiles relies on the often unreasonable assumption that the precise goals of the environment player is available to the system player. Typically, only an approximate understanding of the potential environment behavior is available. As such these assumptions are frequently insufficient to guarantee the existence of a winning strategy for the system player. Consider an example from the robotics domain, where a common problem is that a robot should operate in a workspace shared with a human without collision. The goal of the human, i.e. the environment player, is usually not known, and the human could indeed always move in a way that the robot’s path is blocked. In this case, by using the conventional game setting, we cannot compute any winning strategy for the system player, and hence synthesizing a controller for the robot fails. For successful synthesis, we need to restrict the environment’s behavior without making assumptions about the environment’s goals.

In this paper, we present a new class of games in order to address this seemingly self-contradicting requirement: we restrict the abilities of the environment player without any prior knowledge on its goals. Our starting point is the common idea that, while an environment may behave arbitrarily, it is not its goal to prevent the system from satisfying its requirements. Hence, its behavior should not depend on the internal state of the system (which is encoded into the state space of the game). From the perspective of the environment player, we are thus dealing with a partial-information setting. Simultaneously, from the system’s perspective, the behavior of the environment can be arbitrary, but not arbitrarily complex. We restrict the complexity of the environment’s behavior by requiring that the environment uses only limited memory.

Interestingly, partial-information games against limited memory opponents have not been
Table 1 The complexity of determining if there exists a winning strategy.

| Player | WIN_P | WIN_F |
|--------|-------|-------|
| Memoryless | NP-complete | PSPACE-complete |
| Limited-memory | NP-complete | PSPACE-hard |

studied in the past, even though they are a natural formalization for unknown environment behavior in reactive synthesis. In the present paper, we not only define this new class of games and discuss its basic properties, but also analyze the complexity of solving such games. In this first work on such games, we assume that the environment player does not obtain an additional stimulus from outside of the modeled scenario, and thus it only reacts to the system player’s actions. We show that, surprisingly, even for memoryless environment players, checking if the system player has a winning strategy is PSPACE-complete for safety, reachability and parity objectives. Moreover, for a unary encoding of the memory, the problem can be solved in PSPACE. However, checking if the environment player has a winning strategy is NP-complete. This difference in the complexity highlights the asymmetry between the two players in this new game setting. Note that for a binary encoding of the memory size, the upper bound on the amount of memory needed by a winning partial-information player can be concisely encoded, which means that the EXPTIME-hardness of incomplete-information game solving directly carries over to our setting [1].

We explore a new type of information asymmetry between the two players of a game: the environment player has limited information on the current state of the game, and the system player has no initial information on the goal of the environment player. The system player needs to learn from the environment’s behavior while avoiding situations in which the unknown behavior of the environment player could lead to the loss of the game for the system player [4]. As such, we believe that our work prepares the ground for more formal approaches to the design of self-adaptive systems that are able to achieve reasonable performance in completely unknown environments. In the future, we plan to research how to efficiently computes strategies for the system player in this new class of games.

Results, outline and related work.

We study the problem of determining whether the partial-information player $P$ or the full-information player $F$ have a winning strategy in a game $G$. For $P \in \{P, F\}$, let

$$\text{WIN}_P = \{ (G) \mid G \text{ is a game arena and player } P \text{ has a winning strategy } \}.$$  

The games in the limited-memory setting are not determined, witnessed by the fact that the complexity of determining the existence of a winning strategy for a given game is different for each player as shown in Table 1. The case in which both players have unlimited memory has been analyzed in [15, 5, 1] for different objectives. We study the WIN_P problem under reachability, safety, and parity objectives. The complexity results are summarized in Table 1. Additionally, we establish a lower bound on the memory required by the partial-information player to retain its winning strategies (an upper bound can be obtained from [5]). More specifically, we construct a game where the partial-information player needs at least $O(\sqrt{n})$ bits of memory to preserve a winning strategy in a game of size $O(n)$.

Next, we outline the main ideas in the case of a memoryless partial-information player with a reachability objective. To show the NP-completeness result of the WIN_P problem, we first observe that, given a strategy for the partial-information player, it is easy to check if it is
winning. For proving hardness, we reduce the SAT problem to \( \text{WIN}_F \). It is harder for the full-information player to determine the existence of strategies. More specifically, this problem is PSPACE-complete. To prove that \( \text{WIN}_F \in \text{PSPACE} \), we reduce an arbitrary instance of a game in our setting to a QBF formula with at most \( O(n^2) \) quantifier alternations, such that the full-information player has a winning strategy if and only if the formula is satisfiable. For the hardness result, we define a game with an exit action, which makes the game stop abruptly. If the exit action is played in the initial stage, the full-information player directly wins. If that happens after the initial stage, the partial-information player wins. Moreover, the partial-information player can play the exit action any step. Therefore, the full-information player has to ensure that, in the memoryless strategy played by the partial-information player, no state previously not visited is visited after the initial stage, since the partial-information player can choose the exit action in such a case.

We provide a reduction from QBF to this game, to prove PSPACE-hardness. Thus, we can encode the satisfaction of a QBF instance into the game graph. The two players initially build an assignment to the QBF variables sequentially, which the partial-information player can see. Afterward, the full-information player loses if the partial-information player takes the exit move. The only way for the full-information player to avoid losing is if it does not output a variable assignment part for which it has not previously observed that the partial-information player will not respond with exit. By this manner, the same assignment has to be played repeatedly by both players. By taking the product of this game with an encoding of all clauses in a QBF formula, we obtain the required reduction from a QBF formula, as we know that the partial-information player will always choose the same values if the other player does, and the other player will need to do so to avoid an exit.

The idea behind the construction is that the full-information player cannot know that the assignment selected does not happen to be the only one not causing an exit; hence it has to assume that the partial-information player can play the exit move and hence is subsequently forced to solve the QBF problem to find an initial assignment for which it will not lose the game. Additionally, we show that the hardness carries over to all interesting objectives and also to the case when the partial-information player is forgetful.

For obtaining the lower bound on the memory required by the partial-information player to retain winning strategies, we use the idea of succinct counters from [1] to encode prime numbers \( \leq 2^n \) in a game graph of size \( O(n^2) \). The full-information player chooses a prime number randomly, and then the partial-information player has to guess this number correctly to win the game. When the partial-information player has unlimited memory, the constructed game is always winning for him. However, when we consider the limited memory scenario, the partial information player requires at least \( n\# \) (primorial) distinct states in the memory.

Related work. In most straightforward setting of two-player games on graphs, the players are assumed to have perfect information about the state of the game. This setting has been thoroughly researched in \([14, 8, 17, 11]\). Most recently, quasi-polynomial algorithms for parity are given in \([3, 13]\). The imperfect information games, where the players have asymmetric knowledge about the state of the game has been studied in \([15, 5, 1]\). This setting is particularly useful in the synthesis of controllers that gather information about the state of the game using sensors. In \([6]\) a class of semiperfect-information games, where one player has imperfect information and the other player has perfect information, is studied. This class is more straightforward than the games studied in \([6]\), and it can be solved in \( \text{NP} \cap \text{coNP} \) for parity objectives. However, in this setting, the ambiguity on the current state
is not carried over.

## 2 Definitions

### Arenas.

We study two-player games in which each player chooses, in turn, a symbol from an alphabet. An infinite word is obtained as a result of the play. The game is played in an arena $A = \langle V, \Sigma_P, \Sigma_F, \Delta, v_0 \rangle$ between a partial-information player $P$ and a full-information player $F$, where

- $V = V_{\text{public}} \times V_{\text{private}}$ is the state space;
- $\Sigma_P$ and $\Sigma_F$ are the sets of actions available to the respective players. Moreover, the actions of player $F$ have two components i) a public component $\Sigma_{F_{\text{public}}}$, and ii) a private component $\Sigma_{F_{\text{private}}}$, i.e., $\Sigma_F = \Sigma_{F_{\text{public}}} \times \Sigma_{F_{\text{private}}}$;
- $\Delta : V \times \Sigma_P \times \Sigma_F \to V$ is the transition function; and
- $v_0$ is the initial state of the game.

The two players have asymmetric capabilities. The public component of the state space $V_{\text{public}}$ is visible to both players. The private component $V_{\text{private}}$ is not visible to player $P$. The actions of the full-information player are only partially visible to the partial-information player.

### Strategies.

In every step of the game Player $P$ first makes a move from $\Sigma_P$. Player $F$ responds to player $P$’s action by making a move from $\Sigma_F$. A strategy function for a player depends on the memory available to it and the information it has about the state of the game. If the next move of a player depends only on the current state then the corresponding strategy is memoryless. Memoryless strategies are a special case of limited-memory strategies. Formally, the strategy for the partial-information player $P$ is a function $S_P : V_{\text{public}} \times M \to \Sigma_P \times M$, where $M = \{1, \ldots, k\}$. Player $P$ is memoryless if $k = 1$. The strategy $S_F$ for the full-information player is a function $S_F : V^+ \to \Sigma_F$.

### Plays and Outcomes.

A play $\pi = v_0(\sigma_1^P, \sigma_1^F)v_1(\sigma_2^P, \sigma_2^F)v_2 \ldots$ is an alternating sequence of positions and action pairs, such that for all $i$, there is a valid transition between $v_i$ and $v_{i+1}$. The prefix up to $v_n$ of $\pi$ is denoted $\pi(n)$. Its length denoted $|\pi(n)|$ is $n$ and its last element is $\text{last}(\pi(n)) = v_n$. The set of infinite plays in $A$ is denoted $\text{plays}(A)$, and the set of corresponding finite prefixes is denoted $\text{pref}(A)$. The outcome of two strategies $S_P$ (for player $P$) and $S_F$ (for player $F$) in $A$ is the play $\pi = v_0(\sigma_1^P, \sigma_1^F)v_1(\sigma_2^P, \sigma_2^F)v_2 \ldots \in \text{plays}(A)$, such that for all $i$, a) $\sigma_i^P = S_P(v_{i-1}, M)$, b) $\sigma_i^F = S_F(v_0, \ldots, v_{i-1}, \sigma_i^P)$, and c) $\Delta(v_i, (\sigma_i^P, \sigma_i^F)) = v_{i+1}$. This play is denoted $\text{outcome}(A, S_P, S_F)$.

---

$\downarrow$ \text{i} is the projection to the $i$th component
### Objectives.

An objective $\phi \subseteq V^\omega$ for arena $A$ is a set of infinite sequences of states. A play $\pi = v_0(\sigma_1^P, \sigma_1^F)v_1(\sigma_2^P, \sigma_2^F)v_2$ satisfies the objective $\phi$, denoted $\pi \models \phi$, if $v_0v_1v_2 \cdots \in \phi$. We specifically consider reachability, safety, and parity objectives. For a play $\pi = v_0(\sigma_1^P, \sigma_1^F)v_1(\sigma_2^P, \sigma_2^F)\ldots$, we write $\text{Inf}(\pi)$ for the set of states that appear infinitely often in $\pi$.

- **Reachability**: Given a set $T \subseteq V$ of target states, the *reachability* objective $\text{Reach}(T)$ requires that a state in $T$ is visited at least once, i.e., $\text{Reach}(T) = \{v_0v_1v_2 \cdots \in V^\omega | \exists k \geq 0 : v_k \in T\}$.
- **Safety**: Dually, the *safety* objective $\text{Safe}(T)$ requires that only states in $T$ are visited. Formally, $\text{Safe}(T) = \{v_0v_1v_2 \cdots \in V^\omega | \forall k \in \mathbb{N} : v_k \notin T\}$.
- **Parity**: For $d \in \mathbb{N}$, let $p : V \rightarrow \{0,1,\ldots,d\}$ be a priority function, which maps each state to a non-negative integer priority. The *parity objective* $\text{Parity}(p)$ requires that the minimum priority that appears infinitely often is even. Formally, $\text{Parity}(p) = \{\pi \in V^\omega | \min\{p(v) | v \in \text{Inf}(\pi)\} \text{ is even}\}$.

### Games.

A strategy $\alpha$ for a player with objective $\phi$ is *winning* in $A$, if for every strategy $\beta$ of its opponent, if $\pi \in \text{outcome}(A, \alpha, \beta)$, then $\pi \models \phi$. A *game* $G$ is a pair $(A, \phi)$, where $A$ is an arena as above, and $\phi$ is an objective. An objective $\phi$ for player $i$ is denoted $\phi_i$.

## Memoryless partial-information player

First, we consider the scenario when the partial-information player is memoryless. The results proved for a memoryless partial-information player are then extended to the forgetful case in the next section. We prove that the problem of determining the existence of winning strategies for the partial-information player is NP-complete irrespective of the objective.

**WIN$_P$ is NP-complete.**

Given a memoryless strategy for $P$, we can verify in polynomial time if it is winning. The following lemma is a direct consequence of this fact and is stated without proof.

**Lemma 1.** In a partial-information game with a memoryless partial-information player with parity objective, $\text{WIN}_P \in \text{NP}$.

To show the NP-hardness of the $\text{WIN}_P$ problem, we only need to consider reachability and safety goals for player $P$. We first construct an arena corresponding to any boolean formula in $\text{CNF}$. Following this construction, we show how to encode this CNF formula using the acceptance conditions.

For any propositional formula $\phi = C_1 \land C_2 \land \ldots \land C_m$ in CNF over a set $V_1 = \{x_1, \ldots, x_n\}$ of propositional variables, we construct an arena $A_\phi$ of size at most $O(mn)$. The game arena is constructed in a way such that Player $P$ wins if the game reaches a state corresponding to satisfaction of all the clauses. Simultaneously, the structure of the arena also ensures that the only way for player $P$ to ensure safety (defined later) is by producing a satisfying assignment for $\phi$. Player $P$ chooses assignments for the propositional variables once for every clause. Since $P$ is memoryless, its assignments are the same for every clause. It has to find an assignment such that all the clauses are satisfied. The arena $A_\phi$ is presented in Figure 1. Formally, $A_\phi = (V, \Sigma_P, \Sigma_F, \Delta, v_0)$ is an arena of imperfect-information, where
Lemma 4. See Appendix.

Corollary 3. See Appendix.

Lemma 5. See Appendix.

The set of states of the arena \( V = V^{\text{public}} \times V^{\text{private}} \), where \( V^{\text{public}} = \{\infty\} \cup \{x_1^+, x_2^+, x_3^+, \ldots, x_n^+\} \) and \( V^{\text{private}} = \{C_1, \ldots, C_m\} \times \{\top, \bot\} \). \( V^{\text{public}} \) records the assignment to the propositional variables. The second component \( V^{\text{private}} \) is used by player \( F \) to check the satisfaction of any clause under a partial assignment revealed in the public state.

\[ \Sigma_P = \{\top, \bot\}, \Sigma_F = \{d, n\}. \] The actions of player \( P \) correspond to assignments to the propositional variables. The action \( n \) of player \( F \) corresponds to it proclaiming that a clause is not satisfied. Player \( F \) loses the game if it incorrectly uses \( n \). Otherwise, it has to use the action \( d \).

The initial state \( v_0 = (\infty, (C_1, \bot)) \).

The set of transitions \( \Delta \) is as described below. In each of the transitions below, \( a \) is either \( \top \) or \( \bot \).

1. \((\infty, (C_k, \bot)) \xrightarrow{(a,d)} (x_1^a, (C_k, [C_k]_{x_1 = a}))\)
2. \((x_j^a, (C_k, \alpha)) \xrightarrow{(e,d)} (x_{j+1}^a, (C_k, \alpha \lor [C_k]_{x_j = a}))\), where \((j < n - 1)\)
3. \((x_n^a, (C_k, \top)) \xrightarrow{(a,d)} (\infty, (C_{k+1}, \bot))\), where \((k < m)\)
4. \((x_n^a, (C_k, \bot)) \xrightarrow{(a,n)} (x_n^a, (C_m, \bot))\), where \((k < m)\)
5. \((v, (C_m, \top)) \xrightarrow{(\ast, \ast)} (v, (C_m, \top))\)
6. \((v, (C_m, \bot)) \xrightarrow{(\ast, \ast)} (v, (C_m, \bot))\)

For proving the hardness of the reachability objective, we consider the reachability game \( G^R_\phi = (A_\phi, \text{Reach}(R)) \), where \( R = \{(v, (C_m, \top)) \mid v \in V^{\text{public}}\} \). The following lemma is a direct consequence of the construction of the arena.

Lemma 2. \( \phi \) is satisfiable if and only if memoryless player \( P \) has a winning strategy for winning \( G^R_\phi = (A_\phi, \text{Reach}(R)). \)

Proof. See Appendix.

Corollary 3. The \( \text{WIN}_P \) problem with parity objective for player \( P \) is NP-hard.

To show that \( \text{WIN}_P \) with safety objective for player \( P \) is NP-hard, we consider the game \( G^S_\phi = (A_\phi, \text{Safe}(S)) \), where \( S = V \setminus \{(x_n^+, (C_m, \bot))\} \).

Lemma 4. \( \phi \) is satisfiable if and only if memoryless player \( P \) has a winning strategy for winning \( G^S_\phi = (A_\phi, \text{Safe}(S)) \).

Proof. See Appendix.

\( \text{WIN}_F \) is PSPACE-complete.

Player \( F \) has more ways to defeat player \( P \) when player \( P \)'s strategies are restricted. In this case we prove that determining if \( F \) has a winning strategy is harder. First we show that \( \text{WIN}_F \in \text{PSPACE} \). Consider a partial-information game \( G \) played on arena \( A \), with a parity objective for player \( P \). We bound the maximum number of rounds needed by the full-information player \( F \) to win this game.

Lemma 5. If player \( F \) can win this game, then he can win this game in \( O(n^2) \) rounds, where \( n = |V| \).

\(^2\) \([C^a_{\bot x = e}] \) resolves to \textbf{true} if setting \( x \) to \( e \) in \( C \) satisfies the clause, and resolves to \textbf{false} otherwise.
Playing Against Opponents With Limited Memory

### Figure 1
Games $G^R_\phi$ and $G^S_\phi$ are played on the arena $A_\phi$. $\phi$ is satisfiable if and only if a memoryless player $P$ has a winning strategy.

**Proof.** We prove a more general lemma, and this lemma is a direct consequence.

► **Lemma 6.** Suppose $G = V_3 \prod (V^{public} \times V^{private})$ such that player $P$‘s moves on $V_3$ is fixed. Then, if player $F$ can win, it can win in $(|V_3| + |V^{public}||V^{private}|)(|V^{public}| + 1)$ rounds.

**Proof.** See Appendix.

The next result is now direct.

► **Lemma 7.** In a partial-information game with a memoryless partial-information player with parity objective, $WIN_F \in \text{PSPACE}$. 

---

The diagram shows the arenas $A_\phi$ with states $x_i$ for $i = 1$ to $n$, where $\phi$ is satisfiable. The rounds of the game are depicted, with $G^R_\phi$ and $G^S_\phi$ indicating the winning strategies for the players.
Any two-player game which ends in a bounded number of rounds in the size of the game can be encoded as a QBF formula (formula game) \[16\].

Next, for any QBF formula \( \psi \) of the form \( \forall x_1 \exists y_1 \forall x_2 \exists y_2 \ldots \forall x_n \exists y_n \cdot (C_1 \land C_2 \land \cdots \land C_m) \), we construct an arena \( A_\psi \) of size \( O(m \cdot n) \). The arena \( A_\psi \) is presented in Figure 2. Player \( P \) has partial information and can use only positional strategies. The idea is that player \( P \) has to ensure that its opponent makes the same assignment to its corresponding variables across different rounds (clauses).

Player \( P \) cannot assign different values as he can only play positional strategies. He has a exit move which he can use to push the game to a losing state for player \( F \). By construction, he cannot use this move in the 1st round (1st clause) or else he will lose. As long as player \( F \) makes the play come to the same public component when it comes to player \( P \)’s turn, player \( P \) cannot use a exit move as he is forced to play positional strategies. But if player \( F \) doesn’t know player \( P \’s \) move for this visible component, where player \( P \) can play the exit move which will cause player \( F \) to lose. This ensures that player \( F \) cannot change the assignments as well. Now if there is a strategy for player \( F \) to win this game, then the formula is satisfiable. If there is no strategy for player \( F \) to win, then the formula is unsatisfiable. Formally, \( A_\psi = (V, \Sigma_P, \Sigma_F, \Delta, v_0) \), where

- The set of states of the arena \( V = V_{\text{public}} \times V_{\text{private}} \), where
  \( V_{\text{public}} = \{ \top \} \cup \{ x_1^T, x_1^x, x_2^T, \ldots, x_n^T \} \cup \{ y_1^T, y_1^y, y_2^T, \ldots, y_n^T \} \) and
  \( V_{\text{private}} = \{ C_1, \ldots, C_m \} \times \{ \top, \bot \} \). \( V_{\text{public}} \) records the assignment to the propositional variables that the 2 players jointly decide. The second component \( V_{\text{private}} \) is used by player \( F \) (the full information player) to check the satisfaction of any clause under a partial assignment revealed in the visible state.

- \( \Sigma_1 = \{ \top, \bot, e \} \) and \( \Sigma_2 = \{ \top, \bot \} \). \( e \) is a special move used by player \( P \) (partial-information player) to proclaim that player \( F \) (full-information player) has changed his assignments of variables after using a different assignment to satisfy some clause before.

- The initial state \( v_{\text{init}} = (\top, (C_1, \top)) \).

- The set of transitions \( \Delta \) is as described below. In each of the transitions below, \( a \) is either \( \top \) or \( \bot \).
  1. \( (\ast, (C_1, \ast)) \xrightarrow{(\ast, \ast)} (y_1^T, (C_m, \top)) \)
  2. \( (y_j^T, (C_1, \ast)) \xrightarrow{(\ast, \ast)} (y_j^T, (C_m, \bot)) \) if \( j < n \) and \( m \geq k > 1 \)
  3. \( (\infty, (C_k, \ast)) \xrightarrow{(a, \ast)} (x_1^T, (C_k, \alpha \lor [C_k]x_1 = a)) \)
  4. \( (x_j^a, (C_k, \bot)) \xrightarrow{(\ast, a)} (y_j^a, (C_k, [C_k]y_j = a)) \)
  5. \( (x_j^a, (C_k, \top)) \xrightarrow{(\ast, a)} (y_j^a, (C_k, \top)) \)
  6. \( (y_n^a, (C_k, \top)) \xrightarrow{(\ast, a)} (\infty, (C_{k+1}, \bot)) \) where \( k < m \)
  7. \( (y_n^a, (C_k, \bot)) \xrightarrow{(\ast, a)} (y_n^T, (C_m, \bot)) \) where \( k < m \)
  8. \( (y_j^a, (C_k, \bot)) \xrightarrow{(\ast, a)} (x_j^a, (C_k, [C_k]y_j = a)) \) where \( j < n \)
  9. \( (y_j^a, (C_k, \top)) \xrightarrow{(\ast, a)} (x_j^a, (C_k, \top)) \) where \( j < n \).

To prove PSPACE-hardness of the \( \text{WIN}_F \) problem for different objectives, we only need to consider reachability and safety objectives. For safety objective for \( P \), we consider the game \( G^3_\psi = (A_\psi, Safe(V \setminus R)) \), where \( R = \{(v, (C_m, \top))| v \in V_1 \} \). Player \( F \) wins if the
Games $G^R_\psi$ and $G^S_\psi$ are played on the arena $A_\psi$. $\psi$ is satisfiable if and only if a memoryless player $P$ has a winning strategy. If player $P$ plays an $e$ move in $C_1$, then it loses the game.

The following lemma is a direct consequence of the definition of the game.

**Lemma 8.** $\psi$ is satisfiable if and only if player $F$ has a winning strategy for winning $G^S_\psi = (A_\psi, \text{safe}(V \setminus R))$.

**Proof.** See Appendix.

**Corollary 9.** The $\text{WIN}_F$ problem with parity objective for player $P$ is PSPACE-hard.

To show that $\text{WIN}_F$ with reachability objective for player $P$ is PSPACE-hard, we consider the game $G^R_\psi = (A_\psi, \text{Reach}(R))$, where $R = \{(y^T_i, (C_m, \bot)), (y^\bot_i, (C_m, \bot))\}$. The proof of
Lemma 10. \( \psi \) is satisfiable if and only if player \( F \) has a winning strategy for winning \( G^k_\psi = (A_\psi, \text{Reach}(R)) \).

Proof. The argument from the proof of Lemma [x]

\section{Limited-memory partial-information player}

In this section, the partial-information player \( P \) can use a memory object \( M = \{1, 2, \ldots, k\} \) for his strategies. Therefore, its strategy \( \sigma_P \) is a function \( \sigma_P : V_{\text{public}} \times M \rightarrow \Sigma_P \times M \).

Consider a game \( G = (A, \text{Parity}(p)) \) played on the arena \( A = (V, \Sigma_P, \Sigma_F, \Delta, v_0) \). We construct an equivalent game \( G^k = (A^k, \text{Parity}(p^k)) \) played between a memoryless player \( P \) and player \( F \). In the arena \( A^k \), all the uses of memory by player \( P \) is encoded in the states. The main idea is that player \( F \) forces player \( P \) to exhaust all possible uses of memory for every public component, further this can be encoded in the game. Formally \( A^k = (V', \Sigma'_P, \Sigma'_F, \Delta', v'_0) \) where,

- The set of states of the arena \( V' = (V_{\text{public}}) \times (V_{\text{private}}) \times M \).
- \( \Sigma'_P = \Sigma_P \times M \) and \( \Sigma'_F = \Sigma_F \).
- The initial state \( v'_0 = (v^p_0, v^p_0, k) \).
- The set of transitions \( \Delta' \) is,
  
- \((u, v, i) \xrightarrow{(x, j), y} (u', v, j)\), where \( \Delta' = ((u, v, j)) \) played between a memoryless player \( P \).

We only consider parity objectives for the partial information player, as they can encode both reachability and safety. Let \( \text{Parity}(p) \) be the parity objective of the partial-information player. \( p^k((u, v, i)) = p((u, v)) \) is the priority function \( p^k \) for the new arena. In the new arena, the priority of a state is the corresponding priority in the original arena. The following lemma is straightforward.

Lemma 11. A bounded-memory player \( P \) has a winning strategy against player \( F \) in the game \( G \) if and only if a memoryless \( P \) has a winning strategy against player \( F \) in the game \( G^k \) defined above.

Now as a consequence of Corollary [x] the existence of a winning strategy for player \( P \) is also NP-complete.

Corollary 12. If player \( P \) can use bounded memory, \( \text{WIN}_P \) is NP-complete.

Next, we show that \( \text{WIN}_P \) remains PSPACE-hard even when player \( P \) can use bounded memory. For every game \( G = (A, \text{Parity}(p)) \) played between a bounded-memory player \( P \) and \( F \), we construct an equivalent game \( G_k = (A_k, \text{Parity}(p_k)) \) played between a bounded-memory player \( P \) and \( F \). In the game \( G_k \), player \( P \) has \( k \) unique memory values, therefore for every \( u \in V_{\text{public}} \) it can make at most \( k \) different choices from its actions. However, it cannot make different choices since the game will reach a winning sink for player \( F \). The arena \( A_k = (V_k, \Sigma'_P, \Sigma'_F, \Delta', v'_0) \) where,

- The set of states \( V_k = V_{\text{public}} \times (V_{\text{private}}) \times C \times \{W\} \), where \( V_{\text{private}}' = V_{\text{private}} \times C \times R \).
- \( C = \{0, \ldots, k\} \). It is a counter used by player \( F \) to exhaust all the options of memory that player \( P \) can use at any state. Further \( R = \Sigma_P \). Player \( F \) uses \( R \) to record the move used by player \( P \) when the counter \( C \) has the value 0. \( W \) is a state that is winning for player \( F \), once the game reaches \( W \).
- The initial state \( v'_0 = (u_0, (v_0, 0, \bar{x})) \), where \( (u_0, v_0) \) is the initial state of \( A \) and \( \bar{x} \) is some fixed move in \( R \).

the next lemma is similar to the previous one.
Theorem 15. The actions $\Sigma'_P = \Sigma_P \times \{1, \ldots, k\}$ and $\Sigma'_F = \Sigma_F$.

In the below set of transitions, $\Delta((u, v), (x, y)) = (u', v')$. The set of transitions $\Delta'$ is,

1. $(u, (v, j, x)) \xrightarrow{((x,j), y)} (u, (v, j + 1, x))$, if $1 \leq j < k - 1$.
2. $(u, (v, 0, *)) \xrightarrow{((x,1), y)} (u, (v, 1, x))$.
3. $(u, (v, k - 1, x)) \xrightarrow{((x,0), y)} (u', (v', 0, x))$.
4. $(u, (v, j, x)) \xrightarrow{(e,x)} W$ if $1 \leq j < k - 1$ and $e \neq (x, j + 1)$
5. $(u, (v, k - 1, x)) \xrightarrow{(e,*)} W$ if $e \neq (x, 0)$

The priority function for the arena $p_k(u, (v, *, *)) = p(u, v)$ and $p_k(W) = 1$.

Lemma 13. In the game $G_k$, player $F$ has a winning strategy if and only if it also has winning strategy in $G$ against a memoryless $P$.

Corollary 14. $WIN_F$ is PSPACE-hard.

5 Lower bound on memory for the partial-information player

In this section, we prove a lower bound on the memory required for the partial-information player to retain winning strategies. As the order of play does not matter without loss of generality, we will assume that the full-information player $F$ plays first. In this case, strategy for player $F$ is a function $\pi : V^+ \rightarrow \Sigma_F$. Strategy for player $P$ is a function $\sigma : V^{\text{public}} \times M \times \Sigma_F \rightarrow \Sigma_P \times M$. $M$ is the memory object available to player $P$ and $|M| = k$. $p_n$ denotes the $n^{th}$ prime number.

Remark: $p_n < n^2$.

The prime-remainder game.

Player $P$ and player $F$ play a game in two stages. In the first stage, player $F$ chooses a prime number number $p \in \{p_1, \ldots, p_n\}$, by playing an invisible move $p$ and number $N \in \{0, 1, \ldots, \prod^n_{i=1} p_i\}$. A number $N$ can be chosen by player $F$ by making the move $s N$ times. After this player $F$ reveals the prime number $p$ to player $P$ by playing the move $p$ publicly. The remainder $r = N \mod p$ is stored in the invisible component $V^{\text{private}}$. In the second stage, player $P$ has to guess the remainder $r$ correctly by playing the move $f$ correct number of times then playing an $s$. The arena is presented in Figure 3. Player $F$ wins the game if the game reaches a state (corresponding to player $P$ incorrectly guessing the remainder) in $F = \{(S, v_2) \mid v_2 \text{ is colored blue}\} \cup \{(F, X)\}$.

Analysis of the prime-remainder game.

Theorem 15. Player $P$ always has a winning strategy when it can use unlimited memory.

Proof. Player $P$ can record the number $N$ itself by using a memory structure with $\prod^n_{i=1} p_i$ distinct states. Then player $P$ has a winning strategy against all strategies of player $F$.

Remark: $n^# = \prod^n_{i=1} p_i = e^{(1+o(1))n}$.

Lemma 16. Player $P$ needs a memory object with at least $n^#$ distinct states to win.

Proof. The Chinese-remainder theorem implies that $N$ is uniquely determined by its remainders against the set of primes $\{p_1, p_2, \ldots, p_n\}$. Player $F$ can choose any prime in $p_1, \ldots, p_n$ and this prime is revealed only after the number has been chosen. Since player $P$ cannot know a priori the choice of the prime number $p$, he has to record all the remainders. Therefore, he requires at least $(p_1 - 1)(p_2 - 1) \ldots (p_n - 1)$ states.
Corollary 17. To record \((p_1 - 1)(p_2 - 1)\ldots(p_n - 1)\) states we need \(O(n)\) bits of memory. This game has \(O(n^2)\) states as \(p_n < n^2\). To win in a game of size \(O(n)\), we need a memory with \(O(\sqrt{n})\) bits of memory.

6 Conclusion

To our best knowledge, this paper offers the first study where limited memory is used to capture the non-adversarial nature of the opponent. Moreover, the opponent is also forgetful. We show that the existence of winning strategies for the partial-information player is an NP-complete problem. On the other hand, the existence of winning strategies for the full-information player is a PSPACE-hard problem. Such games played against a weak opponent are asymmetric for the two players, as observed from the complexity results. As expected, limiting the memory available to the opponent introduces new winning strategies. However, these strategies are harder to find when compared to the traditional setting. Surprisingly, the worst-case complexity for determining the existence of winning strategies is independent of the type of the winning condition. However, it is not clear if it is PSPACE-complete.
References

1. Dietmar Berwanger and Laurent Doyen. On the power of imperfect information. In Ramesh Hariharan, Madhavan Mukund, and V. Vinay, editors, IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2008, December 9-11, 2008, Bangalore, India, volume 2 of LIPIcs, pages 73–82. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2008. URL: https://doi.org/10.4230/LIPIcs.FSTTCS.2008.1742

2. Roderick Bloem, Krishnendu Chatterjee, and Barbara Jobstmann. Graph Games and Reactive Synthesis, pages 921–962. Springer International Publishing, Cham, 2018. URL: https://doi.org/10.1007/978-3-319-10575-8_27

3. Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li, and Frank Stephan. Deciding parity games in quasipolynomial time. ACM, Jun 2017. doi:10.1145/3055399.3055409

4. Doran Chakraborty and Peter Stone. Multiagent learning in the presence of memory-bounded agents. Autonomous Agents and Multi-Agent Systems, 28(2):182–213, Mar 2014. doi:10.1007/s10458-013-9222-4

5. Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, and Jean-François Raskin. Algorithms for omega-regular games with imperfect information. In Zoltán Ésik, editor, Computer Science Logic, pages 287–302, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

6. Krishnendu Chatterjee and Thomas A. Henzinger. Semiperfect-Information Games. Springer-Link, pages 1–18, Dec 2005. doi:10.1007/11590156_1

7. Laurent Doyen and Jean-François Raskin. Games with Imperfect Information: Theory and Algorithms, page 185–212. Cambridge University Press, 2011. doi:10.1017/CBO9780511973468.007

8. E. A. Emerson, C. S. Jutla, and A. P. Sistla. On model-checking for fragments of μ-calculus. SpringerLink, pages 385–396, Jun 1993. doi:10.1007/3-540-56922-7_32

9. Erich Grädel, Wolfgang Thomas, and Thomas Wilke. Automata, Logics, and Infinite Games - A Guide to Current Research. Springer-Verlag Berlin Heidelberg, 2002. doi:10.1007/978-3-540-36387-3

10. D. Harel and A. Pnueli. On the Development of Reactive Systems. SpringerLink, pages 477–498, 1985. doi:10.1007/978-3-642-82453-1_17

11. Thomas A. Henzinger. Games, Time, and Probability: Graph Models for System Design and Analysis. SpringerLink, pages 103–110, Jan 2007. doi:10.1007/978-3-540-69507-3_7

12. Paul Hunter, Guillermo A. Pérez, and Jean-François Raskin. Reactive synthesis without regret. Acta Informatica, 54(1):3–39, Feb 2017. doi:10.1007/s00236-016-0268-z

13. Karoliina Lehtinen. A modal μ perspective on solving parity games in quasi-polynomial time. ACM, Jul 2018. doi:10.1145/3209108.3209115

14. Donald A. Martin. Borel Determinacy. Annals of Mathematics, 102(2):363–371, Sep 1975. doi:10.2307/1971035

15. John H. Reif. Universal games of incomplete information. In Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing, STOC ’79, pages 288–308, New York, NY, USA, 1979. ACM. URL: http://doi.acm.org/10.1145/800135.804422

16. Michael Sipser. Introduction to the Theory of Computation. International Thomson Publishing, 1st edition, 1996.

17. Wolfgang Thomas. Infinite Games and Verification. SpringerLink, pages 58–65, Jul 2002. doi:10.1007/3-540-45657-0_5
A Proofs

Proof of Lemma 1 (⇒) If ψ is satisfiable, then there is an assignment to all the variables that satisfies every clause, player P simply chooses this assignment. Player F can never play the n-move as it will take him to a losing state.

(⇐) If ψ is unsatisfiable, then no matter the assignment chosen by player P, there is a clause k that is not satisfied. Player F can play the n-move in this clause to win the game. □

Proof of Lemma 2 When the formula ψ is not satisfiable, the game reaches a state of the form (∗, (C_k, ⊥)). In this state, player F can play the n move to defeat player P. The next lemma formalizes this.

Proof of Lemma 3 Proof is by induction on |V_{public} × V_{private}|. Suppose V_{public} = V_{private} = ∅, then all of player P’s moves are fixed. Hence if player P can win, it can win in |V_3| moves. The start state of the game can be in either

1. V_{public} × V_{private}: Say the game starts in (v_1, v_2). After the first round player P’s moves with respect to {v_1} × V_{private} are fixed for the rest of the game. Hence, we set V_3 = V_3 ∪ ({v_1} × V_{private}) and V_{public} = V_{public} \ {v_1}. Thus, player F wins in 1 + (|V_3| + |V_{private}| + (|V_{public}| − 1)|V_{private}|)(|V_{public}| − 1 + 1) ≤ (|V_3| + |V_{public}||V_{private}|)(|V_{public}| + 1).

2. V_3: If player F can win the game by staying inside V_3, then it can win the game in |V_3| moves. Otherwise player F has to escape the V_3 sub-graph in at most |V_3| moves as player P’s moves in V_3 are known, else it remains stuck in V_3. Say it escapes to (v_1, v_2), using the previous case player F wins in ≤ |V_3| + (|V_3| + |V_{private}| + (|V_{public}| − 1)|V_{private}|)(|V_{public}| − 1 + 1) ≤ (|V_3| + |V_{public}||V_{private}|)(|V_{public}| + 1) moves.

□

Proof of Lemma 4 (⇒) If the formula ψ is satisfiable, for any assignment to a variable x_i by player P, player F has a suitable response. The model for the formula is also a winning strategy for player F. Observe that the partial-information player cannot perform the exit move c as this will directly take him to a winning state of player F (by construction rules 1 and 2 of the transition).

(⇐) We prove the contrapositive. We show that if ψ is not satisfiable, then player F does not have a winning strategy. Since ψ is not satisfiable, the player P has a winning strategy in the standard game interpretation. This implies in any path traversed in the decision tree, there is a clause in ψ that is not satisfied. Let p be one such path.

Suppose if C_l evaluates to ⊥ in p, then some state (y^*_l, (C_k, ⊥)) is reached. Further by rule 7 this always leads to a state where there is no transition out for player F (it does not have a path to any winning state). If this is not the case, there is some clause C_k that is not satisfiable under p. If player F plays the same assignments for every clause under p, by the same argument above it would reach a dead end. Hence player F has to necessarily make a different choice for some 3 variable y_j. Let c_l (1 < l ≤ k) be the clause where it deviates from p for the 1st time. If player P responds by making the exit move (Note that it can do this as per a memoryless strategy since it is the first time it is encountering this vertex in the visible component of the game) irrespective of player F’s response the game hits a dead end as per rule 2.

□