Aalborg Universitet

AALBORG UNIVERSITY

DENMARK

Strategies Resilient to Delay

Games under Delayed Control vs. Delay Games

Fränzle, Martin; Winter, Sarah; Zimmermann, Martin

Published in:
Electronic Proceedings in Theoretical Computer Science, EPTCS

DOI (link to publication from Publisher):
[10.4204/EPTCS.390.14](#)

Creative Commons License
CC BY 4.0

Publication date:
2023

Document Version
Publisher's PDF, also known as Version of record

[Link to publication from Aalborg University](#)

Strategies Resilient to Delay:
Games under Delayed Control vs. Delay Games

Martin Fränzle

Carl v. Ossietzky Universität
Oldenburg, Germany

`martin.fraenzle@uol.de`

Sarah Winter

Université libre de Bruxelles
Brussels, Belgium

`swinter@ulb.ac.be`

Martin Zimmermann

Aalborg University
Aalborg, Denmark

`mzi@cs.aau.dk`

We compare games under delayed control and delay games, two types of infinite games modelling asynchronicity in reactive synthesis. Our main result, the interreducibility of the existence of sure winning strategies for the protagonist, allows to transfer known complexity results and bounds on the delay from delay games to games under delayed control, for which no such results had been known. We furthermore analyze existence of randomized strategies that win almost surely, where this correspondence between the two types of games breaks down.

1 Introduction

Two-player zero-sum games of infinite duration are a standard model for the synthesis of reactive controllers, i.e., correct-by-construction controllers that satisfy their specification even in the presence of a malicious environment. In such games, the interaction between the controller and the environment is captured by the rules of the game and the specification on the controller induces the winning condition of the game. Then, computing a correct controller boils down to computing a winning strategy.

Often, it is convenient to express the rules in terms of a graph capturing the state-space such that moves correspond to transitions between these states. The interaction between the controller and the environment then corresponds to a path through the graph and the winning condition is a language of such paths, containing those that correspond to interactions that satisfy the specification on the controller.

In other settings, it is more convenient to consider a slightly more abstract setting without game graphs, so-called Gale-Stewart games [4]. In such games, the players alternatingly pick a sequence of letters, thereby constructing an infinite word. The winning condition is a language over infinite words, containing the winning words for one player. To capture the synthesis problem, the winning condition has to encode both the specification on the controller as well as the rules of interaction. It is straightforward to transform a graph-based game into a Gale-Stewart game and a Gale-Stewart game into a graph-based game such that the existence of winning strategies for both players is preserved.

In the most basic setting of synthesis, both the controller and the environment are fully informed about the current state of the game (complete information). However, this scenario is not always realistic. Thus, much effort has been poured into studying games under incomplete information where the players are only partially informed about the current state of the game. Here, we are concerned with a special type of partial information designed to capture delays in perception and action. Such delays either render the most recent moves of the opponent invisible to a player or induce a time lag between the selection and the implementation of an own move, respectively.

As a motivating example, consider the domain of cooperative driving: Here, the exchange of information between cars is limited (and therefore delayed) by communication protocols that have to manage the available bandwidth to transfer information between cars. Other delaying factors include, e.g., complex

signal processing chains based on computer vision to detect the locations of obstacles. Thus, decisions have to be made based on incomplete information, which only arrives after some delay.

Games under Delayed Control. Chen et al. [2] introduced (graph) games under delayed control to capture this type of incomplete information. Intuitively, assume the players so far have constructed a finite path $v_0 \cdots v_k$ through the graph. Then, the controller has to base her decision on a visible proper prefix $v_0 \cdots v_{k-\delta}$, where δ is the amount of delay. Hence, the suffix $v_{k-\delta+1} \cdots v_k$ is not yet available to base the decision on, although the decision to be made is to be applied at the last state v_k in the sequence.

They showed that solving games under delayed control with safety conditions and with respect to a given delay is decidable: They presented two algorithms, an exponential one based on a reduction to delay-free safety games using a queue of length δ, and a more practical incremental algorithm synthesizing a series of controllers handling increasing delays and reducing game-graph size in between. They showed that even a naïve implementation of this algorithm outperforms the reduction-based one, even when the latter is used with state-of-the-art solvers for delay-free games. However, the exact complexity of the incremental algorithm and that of solving games under delayed control remained open.

Note that asking whether there is some delay δ that allows controller to win reduces to solving standard, i.e., delay-free games, as they correspond to the case $\delta = 0$. The reason is monotonicity in the delay: if the controller can win for delay δ then also for any $\delta' < \delta$. More interesting is the question whether controller wins with respect to every possible delay. Chen et al. conjectured that there is some exponential δ such that if the controller wins under delay δ, then also under every δ'.

Delay Games. There is also a variant of Gale-Stewart games modelling delayed interaction between the players [7]. Here, the player representing the environment (often called Player I) has to provide a lookahead on her moves, i.e., the player representing the controller (accordingly called Player O) has access to the first $n + k$ letters picked by Player I when picking her n-th letter. So, k is the amount of lookahead that Player I has to grant Player O. Note that the lookahead benefits Player O (representing the controller) while the delay in a game under delayed control disadvantages the controller.

Only three years after the seminal Büchi-Landweber theorem showing that delay-free games with ω-regular winning conditions are decidable [1], Hosch and Landweber showed that it is decidable whether there is a k such that Player O wins a given Gale-Stewart game with lookahead k [7]. Forty years later, Holtmann, Kaiser, and Thomas [6] revisited these games (and dubbed them delay games). They proved that if Player O wins a delay game then she wins it already with at most doubly-exponential lookahead (in the size of a given deterministic parity automaton recognizing the winning condition). Thus, unbounded lookahead does not offer any advantage over doubly-exponential lookahead in games with ω-regular winning conditions. Furthermore, they presented an algorithm with doubly-exponential running time solving delay games with ω-regular winning conditions conditions, i.e., determining whether there exists a k such that Player O wins a given delay game (with its winning condition again given by a deterministic parity automaton) with lookahead k.

Both upper bounds were improved and matching lower bounds were later proven by Klein and Zimmermann [9]: Solving delay games is ExpTime-complete and exponential lookahead is both necessary to win some games and sufficient to win all games that can be won. Both lower bounds already hold for winning conditions specified by deterministic safety automata while the upper bounds hold for deterministic parity automata. The special case of solving games with conditions given as reachability automata is PSpace-complete, but exponential lookahead is still necessary and sufficient. Thus, there are tight complexity results for delay games, unlike for games under delayed control.

Our Contributions. In this work, we exhibit a tight relation between controller in a game under delayed control and Player I in a delay game (recall that these are the players that are disadvantaged by delay and lookahead, respectively). Note that winning conditions in games under delayed control are

always given from the perspective of controller (i.e., she has to avoid unsafe states in a safety game) while winning conditions in delay games are always given from the perspective of Player O. Hence, as we relate controller and Player I, we always have to complement winning conditions.

More precisely, we show that one can transform a safety game under delayed control in polynomial time into a delay game with a reachability condition for Player O (i.e., with a safety condition for Player I) such that controller wins the game under delayed control with delay δ if and only if Player I wins the resulting delay game with lookahead of size $\frac{\delta}{2}$. Dually, we show that one can transform a delay game with safety condition for Player I in polynomial time into a reachability game under delayed control such that Player I wins the delay game with lookahead of size δ if and only if controller wins the resulting game under delayed control with delay 2δ. Thus, we can transfer both upper and lower bound results on complexity and on (necessary and sufficient) lookahead from delay games to delayed control. In particular, determining whether controller wins a given safety game under delayed control for every possible delay is PSPACE-complete. Our reductions also prove the conjecture by Chen et al. on the delays that allow controller to win such games. Furthermore, we generalize our translation from games with safety conditions to games with parity conditions and games with winning conditions given by formulas of Linear Temporal Logic (LTL) [12], again allowing us to transfer known results for delay games to games under delayed control.

Note that we have only claimed that the existence of winning strategies for the controller in the game under delayed control and Player I in the delay game coincides. This is no accident! In fact, the analogous result for relating environment and Player O fails. This follows immediately from the fact that delay games are determined while games under delayed control are undetermined, even with safety conditions. The reason is that the latter games are truly incomplete information games (which are typically undetermined) while delay games are perfect information games.

We conclude by a detailed comparison between environment and Player O in both the setting with deterministic as well as in the setting with randomized strategies. The latter setting increases power for both the controller and the environment, making them win (almost surely) games under delayed control that remain undetermined in the deterministic setting, but it also breaks the correspondence between controller and Player I observed in the deterministic setting: there are games that controller wins almost surely while Player I surely looses them.

All proofs which are omitted due to space restrictions can be found in the full version [3].

2 Preliminaries

We denote the non-negative integers by $\mathbb{N}$. An alphabet Σ is a non-empty finite set of letters. A word over Σ is a finite or infinite sequence of letters of Σ: The set of finite words (non-empty finite words, infinite words) over Σ is denoted by Σ^* (Σ^+, Σ^ω). The empty word is denoted by ε, the length of a finite word w is denoted by $|w|$. Given two infinite words $\alpha \in (\Sigma_0)^\omega$ and $\beta \in (\Sigma_1)^\omega$, we define $\binom{\alpha}{\beta} = \binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\binom{\alpha(2)}{\beta(2)}\cdots \in (\Sigma_0 \times \Sigma_1)^\omega$.

2.1 Games under Delayed Control

Games under delayed control are played between two players, controller and environment. For pronomial convenience [11], we refer to controller as she and environment as he.

A game $\mathscr{G} = (S, s_0, S_c, S_e, \Sigma_c, \Sigma_e, \rightarrow, \text{Win})$ consists of a finite set S of states partitioned into the states $S_c \subseteq S$ of the controller and the states $S_e \subseteq S$ of the environment, an initial state $s_0 \in S_c$, the sets of

actions Σ_c for the controller and Σ_e for the environment, a transition function $\to\colon (S_c \times \Sigma_c) \cup (S_e \times \Sigma_e) \to S$ such that $s \in S_c$ and $\sigma \in \Sigma_c$ implies $\to(s,\sigma) \in S_e$ and vice versa, and a winning condition $\mathrm{Win} \subseteq S^\omega$. We write $s \xrightarrow{\sigma} s'$ as shorthand for $s' = \to(s,\sigma)$.

A play in $\mathscr{G}$ is an infinite sequence $\pi = \pi_0 \sigma_0 \pi_1 \sigma_1 \pi_2 \sigma_2 \cdots$ satisfying $\pi_0 = s_0$ and $\pi_n \xrightarrow{\sigma_n} \pi_{n+1}$ for all $n \geq 0$. We say that controller wins π if $\pi_0 \pi_1 \pi_2 \cdots \in \mathrm{Win}$; otherwise, we say that environment wins π. The play prefix of π of length n is defined as $\pi[n] = \pi_0 \sigma_0 \cdots \sigma_{n-1} \pi_n$, i.e., n is the number of actions (equivalently, the number of transitions). We denote by $\mathrm{Pref}(\mathscr{G})$ the set of play prefixes of all plays in $\mathscr{G}$, which is partitioned into the sets $\mathrm{Pref}_c(\mathscr{G})$ and $\mathrm{Pref}_e(\mathscr{G})$ of play prefixes ending in S_c and S_e, respectively. Due to our alternation assumption, play prefixes of even (odd) length are in $\mathrm{Pref}_c(\mathscr{G})$ ($\mathrm{Pref}_e(\mathscr{G})$).

Fix some even $\delta \geq 0$. A strategy for the controller in $\mathscr{G}$ under delay δ is a pair (α, τ_c) where $\alpha \in (\Sigma_c)^{\frac{\delta}{2}}$ and $\tau_c \colon \mathrm{Pref}_c(\mathscr{G}) \to \Sigma_c$ maps play prefixes ending in S_c to actions of the controller. A play $\pi_0 \sigma_0 \pi_1 \sigma_1 \pi_2 \sigma_2 \cdots$ is consistent with (α, τ_c) if $\sigma_0 \sigma_2 \cdots \sigma_{\delta-4} \sigma_{\delta-2} = \alpha$ and $\sigma_{2n} = \tau_c(\pi[2n-\delta])$ for all $2n > \delta - 2$, i.e., controller has access to environment's actions with a delay of δ. In particular, her first $\frac{\delta}{2} + 1$ actions are independent of environment's actions and, in general, her n-th action σ_{2n} only depends on the actions $\sigma_1, \ldots, \sigma_{(2n-\delta)-1}$ picked by environment, but not on the actions $\sigma_{(2n-\delta)+1}, \ldots, \sigma_{2n-1}$. The strategy (α, τ_c) is winning under delay δ if every play that is consistent with it is winning for controller. Controller wins $\mathscr{G}$ under delay δ if she has a winning strategy under delay δ for $\mathscr{G}$.

Remark 1.

1. *The notion of winning strategy for controller under delay 0 is the classical one for delay-free games (cf. [5]).*

2. *If controller wins $\mathscr{G}$ under delay δ, then also under every delay $\delta' < \delta$ [2].*

A strategy for environment is a mapping $\tau_e \colon \mathrm{Pref}_e(\mathscr{G}) \to \Sigma_e$. A play $\pi_0 \sigma_0 \pi_1 \sigma_1 \pi_2 \sigma_2 \cdots$ is consistent with τ_e if $\sigma_{2n+1} = \tau_e(\pi_0 \sigma_0 \cdots \sigma_{2n-1} \pi_{2n+1})$ for all $n \geq 0$, i.e., environment has access to the full play prefix when picking his next action. The strategy τ_e is winning, if every play that is consistent with it is winning for the environment (i.e., the sequence of states is not in Win). Further, we say that environment wins $\mathscr{G}$, if he has a winning strategy for $\mathscr{G}$. Note that the two definitions of strategies are in general not dual, e.g., the one for environment is not defined with respect to a delay δ.

Remark 2. *The notion of winning strategy for environment is the classical one for delay-free games (cf. [5]).*

We say that a game under delayed control $\mathscr{G}$ is determined under delay δ, if either controller wins $\mathscr{G}$ under delay δ or environment wins $\mathscr{G}$. Let us stress that determinacy is defined with respect to some fixed δ and that $\mathscr{G}$ may be determined for some δ, but undetermined for some other δ' (due to the non-dual definition of strategies). Remark 9 shows an undetermined safety (!) game under delayed control.

Example 1. *Consider the game $\mathscr{G} = (S, s_I, S_c, S_e, \Sigma_c, \Sigma_e, \to, \mathrm{Win})$ depicted in Fig. 1 where* Win *contains all plays that do not visit the black vertex. Note that this is a safety condition. In particular, if controller does not pick action b at c_2 and does not pick action a at c_3, then the vertex e_3 is never reached. This is straightforward without delay, but we claim that controller can also win $\mathscr{G}$ under delay 2.*

To gain some intuition, consider a play prefix $\pi_0 \sigma_0 \pi_1 \cdots \pi_{n-1} \sigma_{n-1} \pi_n$ with $n \geq 4$ and $\pi_n \in S_c$. Then, controller has to pick an action σ_n to continue the prefix. However, due to the delayed control, she has to do so based on the prefix $\pi_0 \sigma_0 \pi_1 \cdots \pi_{n-3} \sigma_{n-3} \pi_{n-2}$.

If π_{n-2} is c_2, then π_n is either c_3 or c_1. Hence, picking $\sigma_n = b$ is the only safe choice. Dually, if π_{n-2} is c_3, then π_n is either c_2 or c_1. Hence, picking $\sigma_n = a$ is the only safe choice.
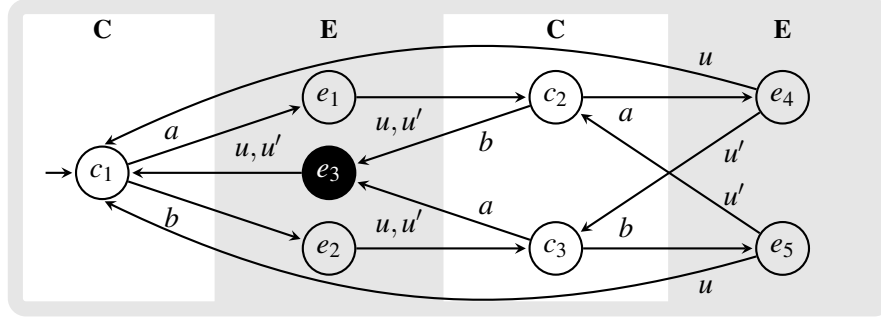
Figure 1: The game for Example 1. Controller wins all plays that never visit the black vertex. Note that we have $\Sigma_c = \{a, b\}$ and $\Sigma_e = \{u, u'\}$.

Finally, assume π_{n-2} is c_1. Then, π_n is either c_2 or c_3. In the former case, picking $\sigma_n = a$ is the only safe choice, in the latter case, picking $\sigma_n = b$ is the only safe choice. So, controller needs to distinguish these two cases, although she has no access to π_n.

But she can do so by inspecting π_{n-3} (which she has access to): As a predecessor of $\pi_{n-2} = c_1$, it can either be e_4, e_5, or e_3. In the latter case, the play is already losing. Thus, we disregard this case, as we construct a winning strategy. So, assume we have $\pi_{n-3} = e_4$ (the case $\pi_{n-3} = e_5$ is dual). Then, we must have $\pi_{n-4} = c_2$ (the only predecessor of e_4) and, by our analysis of the safe moves above, controller must have picked $\sigma_{n-2} = b$ (based, due to delay, on the prefix ending in $\sigma_{n-4} = c_2$). From this we can conclude $\pi_{n-1} = e_2$ and thus $\pi_n = c_3$ (the only successor of e_2). Thus, she can safely pick $\sigma_n = b$.

This intuition, and the necessary initialization, is implemented by the strategy (α, τ_c) with $\alpha = a$ and

$$\tau_c(\pi_0 \sigma_0 \pi_1 \cdots \pi_{n-3} \sigma_{n-3} \pi_{n-2}) = \begin{cases} a & n = 2 \text{ and } \pi_0 = c_1, \\ b & n > 2, \ \pi_{n-2} = c_1, \text{ and } \pi_{n-3} = e_4, \\ a & n > 2, \ \pi_{n-2} = c_1, \text{ and } \pi_{n-3} = e_5, \\ b & \pi_{n-2} = c_2, \\ a & \pi_{n-2} = c_3. \end{cases}$$

An induction over the play length shows that (α, τ_c) is winning for controller under delay 2.

Remark 3. *Our definition of games under delayed control differs in three aspects from the original definition of Chen et al. [2].*

- *We allow arbitrary winning conditions while Chen et al. focused on safety conditions.*

- *The original definition allows nondeterministic strategies (a strategy that returns a nonempty set of actions, each one of which can be taken), while we restrict ourselves here to deterministic strategies (a strategy that returns a single action to be taken). The motivation for their use of nondeterministic strategies is the fact that they can be refined if additional constraints are imposed, which Chen et al.'s algorithm computing a winning strategy relies on.*

   Here, on the other hand, we are just interested in the existence of winning strategies. In this context, it is sufficient to consider deterministic strategies, as controller has a nondeterministic winning strategy if and only if she has a deterministic winning strategy. Also, strategies in delay games are deterministic, so the transformation between games under delayed control and delay games can be formulated more naturally for deterministic strategies.

- *The original definition also allowed odd delays δ while we only allow even delays. As we will see in Section 3, the transformation of games under delayed control to delay games is naturally formulated for even delays. This choice also simplifies definitions, as accounting for odd delays imposes an additional notational burden.*

2.2 Delay Games

Delay games are played between two players, Player I (she) and Player O (he). A delay game $\Gamma_k(L)$ (with constant lookahead) consists of a lookahead $k \in \mathbb{N}$ and a winning condition $L \subseteq (\Sigma_I \times \Sigma_O)^\omega$ for some alphabets Σ_I and Σ_O. Such a game is played in rounds $n = 0, 1, 2, \ldots$ as follows: in round 0, first Player I picks a word $x_0 \in \Sigma_I^{k+1}$, then Player O picks a letter $y_0 \in \Sigma_O$. In round $n > 0$, Player I picks a letter $x_n \in \Sigma_I$, then Player O picks a letter $y_n \in \Sigma_O$. Player O wins a play $(x_0, y_0)(x_1, y_1)(x_2, y_2) \cdots$ if the outcome $\binom{x_0 x_1 x_2 \cdots}{y_0 y_1 y_2 \cdots}$ is in L; otherwise, Player I wins.

A strategy for Player I in $\Gamma_k(L)$ is a mapping $\tau_I \colon \Sigma_O^* \to \Sigma_I^*$ satisfying $|\tau_I(\varepsilon)| = k+1$ and $|\tau_I(w)| = 1$ for all $w \in \Sigma_O^+$. A strategy for Player O is a mapping $\tau_O \colon \Sigma_I^+ \to \Sigma_O$. A play $(x_0, y_0)(x_1, y_1)(x_2, y_2) \cdots$ is consistent with τ_I if $x_n = \tau_I(y_0 \cdots y_{n-1})$ for all $n \geq 0$, and it is consistent with τ_O if $y_n = \tau_O(x_0 \cdots x_n)$ for all $n \geq 0$. So, strategies are dual in delay games, i.e., Player I has to grant some lookahead on her moves that Player O has access to. A strategy for Player $P \in \{I, O\}$ is winning, if every play that is consistent with the strategy is won by Player P. We say that Player $P \in \{I, O\}$ wins a game $\Gamma_k(L)$ if Player P has a winning strategy in $\Gamma_k(L)$.

Remark 4.

- *If Player O wins $\Gamma_k(L)$, then he also wins $\Gamma_{k'}(L)$ for every $k' > k$.*

- *If Player I wins $\Gamma_k(L)$, then she also wins $\Gamma_{k'}(L)$ for every $k' < k$.*

Unlike games under delayed control, delay games with Borel winning conditions are determined [9], i.e., each delay game $\Gamma_k(L)$ with Borel L and fixed k is won by one of the players.

Example 2. *Consider $L = \left\{ \binom{a_0}{b_0} \binom{a_1}{b_1} \binom{a_2}{b_2} \cdots \mid b_0 \notin \{a_0, a_1, a_2\} \right\}$ over the alphabets $\Sigma_I = \Sigma_O = \{1, 2, 3, 4\}$.*

Player I wins $\Gamma_k(L)$ for $k = 1$ with the following strategy τ_I: $\tau_I(\varepsilon) = 12$ and $\tau_I(b_0) = b_0$, and $\tau_I(w)$ arbitrary for all $w \in \Sigma_O^+$ with $|w| > 1$: In round 0, after Player I has picked $a_0 a_1 = 12$, Player O has to pick some b_0. In order to not loose immediately, he has to pick $b_0 \notin \{1, 2\}$. Then, in round 1, Player I picks $a_2 = b_0$ and thereby ensures $b_0 \in \{a_0, a_1, a_2\}$. Hence, the play is not won by Player O (it's outcome is not in L), therefore it is winning for Player I.

However, Player O wins $\Gamma_k(L)$ for $k = 2$ with the following strategy τ_O: $\tau_O(a_0 a_1 a_2)$ is a letter in the nonempty set $\Sigma_O \setminus \{a_0, a_1, a_2\}$ and $\tau_O(w)$ arbitrary for all $w \in \Sigma_I^*$ with $|w| \neq 3$. In round 0, after Player I has picked $a_0 a_1 a_2$, Player O picks $b_0 \notin \{a_0, a_1, a_2\}$ and thus ensures that the outcome is in L.

Remark 5. *We restrict ourselves here to the setting of constant lookahead, i.e., in a delay game $\Gamma_k(L)$ in round n when Player O picks her n-th letter, Player I has already picked $k + n + 1$ letters (note that we start in round 0 with the zeroth letter). Delay games have also been studied with respect to growing lookahead, i.e., the lookahead increases during a play [6]. However, it is known that constant lookahead is sufficient for all ω-regular winning conditions: if Player O wins for any lookahead (no matter how fast it is growing), then she also wins with respect to constant lookahead, which can even be bounded exponentially in the size of a deterministic parity automaton recognizing the winning condition [9]. Stated differently, growing lookahead does not allow to win any more games than constant lookahead. Finally, the setting of constant lookahead in delay games considered here is the natural counterpart to games under delayed control, where the delay is fixed during a play.*

2.3 ω-Automata

A deterministic reachability automaton $\mathscr{A} = (Q, \Sigma, q_I, \delta_{\mathscr{A}}, F)$ consists of a finite set Q of states containing the initial state $q_I \in Q$ and the set of accepting states $F \subseteq Q$, an alphabet Σ, and a transition function $\delta_{\mathscr{A}} : Q \times \Sigma \to Q$. The size of $\mathscr{A}$ is defined as $|\mathscr{A}| = |Q|$. Let $w = w_0 w_1 w_2 \cdots \in \Sigma^{\omega}$. The run of $\mathscr{A}$ on w is the sequence $q_0 q_1 q_2 \cdots$ such that $q_0 = q_I$ and $q_{n+1} = \delta_{\mathscr{A}}(q_n, w_n)$ for all $n \geq 0$. A run $q_0 q_1 q_2 \cdots$ is (reachability) accepting if $q_n \in F$ for some $n \geq 0$. The language (reachability) recognized by $\mathscr{A}$, denoted by $L(\mathscr{A})$, is the set of infinite words over Σ such that the run of $\mathscr{A}$ on w is (reachability) accepting.

A deterministic safety automaton has the form $\mathscr{A} = (Q, \Sigma, q_I, \delta_{\mathscr{A}}, U)$ where $Q, \Sigma, q_I, \delta_{\mathscr{A}}$ are as in a deterministic reachability automaton and where $U \subseteq Q$ is a set of unsafe states. The notions of size and runs are defined as for reachability automata, too. A run $q_0 q_1 q_2 \cdots$ is (safety) accepting if $q_n \notin U$ for all $n \geq 0$. The language (safety) recognized by $\mathscr{A}$, again denoted by $L(\mathscr{A})$, is the set of infinite words over Σ such that the run of $\mathscr{A}$ on w is (safety) accepting.

A deterministic parity automaton has the form $\mathscr{A} = (Q, \Sigma, q_I, \delta_{\mathscr{A}}, \Omega)$ where $Q, \Sigma, q_I, \delta_{\mathscr{A}}$ are as in a deterministic reachability automaton and where $\Omega : Q \to \mathbb{N}$ is a coloring of the states. The notions of size and runs are defined as for reachability automata, too. A run $q_0 q_1 q_2 \cdots$ is (parity) accepting if the maximal color appearing infinitely often in the sequence $\Omega(q_0)\Omega(q_1)\Omega(q_2) \cdots$ is even. The language (parity) recognized by $\mathscr{A}$, again denoted by $L(\mathscr{A})$, is the set of infinite words over Σ such that the run of $\mathscr{A}$ on w is (parity) accepting.

Reachability and safety automata are dual while parity automata are self-dual.

Remark 6.

1. Let $\mathscr{A} = (Q, \Sigma, q_I, \delta_{\mathscr{A}}, F)$ be a deterministic reachability automaton and let $\overline{\mathscr{A}}$ be the deterministic safety automaton $(Q, \Sigma, q_I, \delta_{\mathscr{A}}, Q \setminus F)$. Then, $L(\overline{\mathscr{A}}) = \overline{L(\mathscr{A})}$.

2. Let $\mathscr{A} = (Q, \Sigma, q_I, \delta_{\mathscr{A}}, \Omega)$ be a deterministic parity automaton and let $\overline{\mathscr{A}}$ be the deterministic parity automaton $(Q, \Sigma, q_I, \delta_{\mathscr{A}}, q \mapsto \Omega(q) + 1)$. Then, $L(\overline{\mathscr{A}}) = \overline{L(\mathscr{A})}$.

3   From Games under Delayed Control to Delay Games and Back

In this section, we exhibit a tight correspondence between controller in games under delayed control and Player I in delay games. Recall that in a game under delayed control, it is the controller whose control is delayed, i.e., she is at a disadvantage as she only gets delayed access to the action picked by environment. In a delay game, it is Player I who is at a disadvantage as she has to grant a lookahead on her moves to Player O. Thus, when simulating a game under delayed control by a delay game, it is natural to let Player I take the role of controller and let Player O take the role of environment. Also recall that the winning condition Win in a game under delayed control is formulated from controller's point-of-view: the winning condition requires her to enforce a play in Win. On the other hand, the winning condition L of a delay game is formulated from the point-of-view of Player O: Player O has to enforce a play whose outcome is in L. Thus, as Player I takes the role of controller, we need to complement the winning condition to reflect this change in perspective: The set of winning outcomes for Player I in the simulating delay game is the complement of Win.

In the remainder of this section, we show how to simulate a game under delayed control by a delay game and then the converse, i.e., how to simulate a delay game by a game under delayed control.

Transformation 1. *First, we transform a game under delayed control into a delay game. In the resulting delay game, the players simulate a play in the game under delayed control by picking actions,*

which uniquely induce such a play. To formalize this, we need to introduce some notation. Fix a game $\mathscr{G} = (S, s_0, S_c, S_e, \Sigma_c, \Sigma_e, \rightarrow, \mathrm{Win})$. *Note that a sequence* $\sigma_0 \sigma_1 \sigma_2 \cdots \in (\Sigma_c \Sigma_e)^\omega$ *induces a unique play* $\mathrm{play}(\sigma_0 \sigma_1 \sigma_2 \cdots) = \pi_0 \sigma_0 \pi_1 \sigma_1 \pi_2 \sigma_2 \cdots$ *in* $\mathscr{G}$ *which is defined as follows:* $\pi_0 = s_0$ *and* $\pi_{n+1} = \rightarrow (\pi_n, \sigma_n)$ *for all* $n \geq 0$. *Likewise, a finite sequence* $\sigma_0 \sigma_1 \cdots \sigma_n \in (\Sigma_c \Sigma_e)^*(\Sigma_c + \varepsilon)$ *induces a unique play prefix* $\mathrm{play}(\sigma_0 \sigma_1 \cdots \sigma_n)$ *which is defined analogously.*

Now, we define the language $L(\mathscr{G}) \subseteq (\Sigma_c \times \Sigma_e)^\omega$ *such that* $\binom{\sigma_0}{\sigma_1} \binom{\sigma_2}{\sigma_3} \binom{\sigma_4}{\sigma_5} \cdots \in L(\mathscr{G})$ *if and only if* $\mathrm{play}(\sigma_0 \sigma_1 \sigma_2 \cdots)$ *is winning for controller.*

Now, we prove the correspondence between $\mathscr{G}$ and $\Gamma_k(\overline{L(\mathscr{G})})$. The winning condition of the delay game is the complement of $L(\mathscr{G})$, which implements the switch of perspective described above.

Lemma 1. *Let* $\mathscr{G}$ *be a game and* $\delta \geq 0$ *even. Controller wins* $\mathscr{G}$ *under delay* δ *if and only if Player I wins* $\Gamma_k(\overline{L(\mathscr{G})})$ *for* $k = \frac{\delta}{2}$.

Now, we consider the converse and transform a delay game into a game under delayed control.

Transformation 2. *Fix a delay game* $\Gamma_k(L)$. *We construct a game under delayed control to simulate* $\Gamma_k(L)$ *as follows: The actions of controller are the letters in* Σ_I, *and the actions of environment are the letters in* Σ_O. *Thus, by picking actions, controller and environment construct the outcome of a play of* $\Gamma_k(L)$. *As winning conditions of games under delayed control only refer to states visited by a play, but not the actions picked by the players, we reflect the action picked by a player in the state reached by picking that action. Here, we have to require without loss of generality that* Σ_I *and* Σ_O *are disjoint.*

Formally, we define $\mathscr{G}(L) = (S, s_0, S_c, S_e, \Sigma_c, \Sigma_e, \rightarrow, \mathrm{Win})$ *with* $S = S_c \cup S_e$, $S_c = \{s_0\} \cup \Sigma_O$, $S_e = \Sigma_I$, $\Sigma_c = \Sigma_I$, $\Sigma_e = \Sigma_O$, $\rightarrow(s, a) = a$ *for all* $s \in S_c$ *and* $a \in \Sigma_I$, *and* $\rightarrow(s, b) = b$ *for all* $s \in S_e$ *and* $b \in \Sigma_O$. *Finally, we define* $\mathrm{Win} = \{s_0 s_1 s_2 \cdots \mid \binom{s_0}{s_1} \binom{s_2}{s_3} \binom{s_4}{s_5} \cdots \in L\}$.

The following remark states that the two transformations are inverses of each other, which simplifies the proof of correctness of the second transformation. It follows by a careful inspection of the definitions.

Remark 7. *Let* $L \subseteq (\Sigma_I \times \Sigma_O)^\omega$. *Then,* $L = L(\mathscr{G}(L))$.

Now, we show that the second transformation is correct, again using complementation to implement the perspective switch.

Lemma 2. *Let* $L \subseteq (\Sigma_I \times \Sigma_O)^\omega$ *and* $k \geq 0$. *Player I wins* $\Gamma_k(L)$ *if and only if controller wins* $\mathscr{G}(\overline{L})$ *under delay* $2k$.

4 Results

Lemma 1 and Lemma 2 allow us to transfer results from delay games to games under delayed control. Due to the definitions of strategies in games under delayed control not being dual, we consider both players independently, controller in Section 4.1 and environment in Section 4.2.

Recall that delay that allows controller to win satisfies a monotonicity property (see Remark 1.2): if controller wins a game under delay δ, then also under every delay $\delta' < \delta$. Thus, the set of delays for which controller wins is downward-closed, i.e., it is either a finite set $\{0, 2, 4 \ldots, \delta_{\max}\}$ or it is equal to the set $2\mathbb{N}$ of even numbers. In the following, we study the complexity of determining whether controller wins under all possible delays, whether she wins under a given delay, and determine bounds on $\delta_{\max}$.

Note that winning for environment is independent of delay and boils down to the classical notion of winning delay-free games [5], which is a well-studied problem. Hence, we disregard this problem. However, we do discuss the relation between environment in a game under delayed control and Player O in the simulating delay game constructed in the previous section.

4.1   Controller's View

Before we present our results, we need to specify how to measure the size of games and delay games, especially how winning conditions are represented (recall that, so far, they are just ω-languages). In the following, we only consider ω-regular winning conditions specified by ω-automata (see Section 2.3) or formulas of Linear Temporal Logic (LTL) [12], which subsume the typical specification languages for winning conditions. Hence, the size of a game $(S, s_0, S_c, S_e, \Sigma_c, \Sigma_e, \rightarrow, \text{Win})$ under delayed control is given by the sum $|S| + |\Sigma_c| + |\Sigma_e| + |\text{Win}|$, where $|\text{Win}|$ is the size of an automaton or LTL formula (measured in the number of distinct subformulas) representing Win. Analogously, for a delay game $\Gamma_k(L)$, we define the size of L as the size of an automaton or LTL formula (measured in the number of distinct subformulas) representing L. The bound k is encoded in binary, if necessary.

Safety.   A game $\mathscr{G} = (S, s_0, S_c, S_e, \Sigma_c, \Sigma_e, \rightarrow, \text{Win})$ with winning condition Win is a safety game if Win is accepted by a deterministic safety automaton.

Remark 8.   *When Chen et al. introduced safety games under delayed control, they did not use automata to specify their winning plays, but instead equipped the game with a set of unsafe states and declared all those plays winning for controller that never visit an unsafe state. It is straightforward to see that our definition is equivalent, as their definition is captured by a deterministic safety automaton with two states. Conversely, taking the product of a game and a deterministic safety automaton yields an equivalent game with a state-based safety condition.*

Our results rely on the following two bounds on the transformations presented in Section 3, which are obtained by applying Remark 6:

1. If the winning condition Win for a game $\mathscr{G}$ under delayed control is given by a deterministic safety automaton with n states, then the winning condition $\overline{L(\mathscr{G})}$ is recognized by a deterministic reachability automaton with n states.

2. Dually, if the winning condition $L \subseteq (\Sigma_I \times \Sigma_O)^{\omega}$ of a delay game is given by a deterministic reachability automaton with n states, then the winning condition of the game $\mathscr{G}(\overline{L})$ under delayed action is recognized by a deterministic safety automaton with $\mathscr{O}(n \cdot |\Sigma_I|)$ states.

We begin by settling the complexity of determining whether controller wins a given safety game under every delay, which follows from the PSPACE-completeness of determining whether there is a lookahead that allows Player O to win a given delay game with reachability winning condition [9].

Theorem 1.   *The following problem is* PSPACE-*complete: Given a safety game $\mathscr{G}$, does controller win $\mathscr{G}$ under every delay δ?*

Next, we give a lower bound on the complexity of determining whether controller wins a given safety game under a given delay, which is derived from a lower bound for delay games with reachability winning conditions.

Theorem 2.   *The following problem is* PSPACE-*hard: Given a safety game $\mathscr{G}$ and δ (encoded in binary), does controller win $\mathscr{G}$ under delay δ.*

Note that we do not claim any upper bound on the problem considered in Theorem 2. There is a trivial 2EXPTIME upper bound obtained by hardcoding the delay into the graph of the safety game, thereby obtaining a classical delay-free safety game. It is open whether the complexity can be improved. Let us remark though that, via the correspondence to delay games presented in Section 3, improvements here would also yield improvements on the analogous problem for delay games, which is open too [15].

Next, we turn our attention to bounds on the delay for which controller wins. Recall that due to monotonicity, the set of delays for which controller wins is downward-closed, i.e., it is either a finite set $\{0, 2, 4 \ldots, \delta_{\max}\}$ or it is equal to $2\mathbb{N}$. In the following, we present tight bounds on the value $\delta_{\max}$.

As a consequence, we settle a conjecture by Chen et al.: They conjectured that there is some delay δ_t (exponential in $|\mathscr{G}|$), such that if controller wins $\mathscr{G}$ under delay δ_t, then she wins under every delay. Note that this conjecture implies that $\delta_{\max}$ is at most exponential.

The following theorem proves Chen et al.'s conjecture, while Theorem 4 shows that δ_t must necessarily be exponential. For $\delta_{\max}$ this means it is at most exponential for every game, and can be exponential for some games.

The following two results are again obtained from similar bounds for delay games with reachability winning conditions.

Theorem 3. *Let $\mathscr{G}$ be a safety game. There is a $\delta_t \in \mathscr{O}(2^{|\mathscr{G}|})$ such that if controller wins $\mathscr{G}$ under delay δ_t, then she wins $\mathscr{G}$ under every δ.*

Finally, we show that the exponential upper bound on $\delta_{\max}$ is tight.

Theorem 4. *For every $n > 1$, there is a safety game $\mathscr{G}_n$ of size $\mathscr{O}(n)$ such that controller wins $\mathscr{G}$ under delay 2^n, but not under delay $2^n + 2$.*

Parity. Next, we consider the case of ω-regular winning conditions, given by deterministic parity automata. Applying Remark 6 yields the following two bounds on the transformations from Section 3:

1. If the winning condition Win for a game $\mathscr{G}$ under delayed control is given by a deterministic parity automaton with n states, then the winning condition $\overline{L(\mathscr{G})}$ is recognized by a deterministic parity automaton with n states.

2. Dually, if the winning condition $L \subseteq (\Sigma_I \times \Sigma_O)^{\omega}$ of a delay game is given by a deterministic parity automaton with n states, then the winning condition of the game $\mathscr{G}(\overline{L})$ under delayed action is recognized by a deterministic parity automaton with $\mathscr{O}(n \cdot |\Sigma_I|)$ states.

Exponential lookahead is both sufficient to win all ω-regular delay games that can be won and required to win some of these games [9]. Furthermore, determining whether there is some lookahead that allows Player O to win a given ω-regular delay game is ExpTime-complete [9]. As in the case of safety games, we can transfer these results to games under delayed control with ω-regular winning conditions.

Theorem 5.

1. *The following problem is* ExpTime-*complete: Given a game $\mathscr{G}$ with ω-regular winning condition specified by a deterministic parity automaton, does controller win $\mathscr{G}$ under every delay δ?*

2. *Let $\mathscr{G}$ be a game with ω-regular winning condition specified by a deterministic parity automaton with n states. There is a $\delta_t \in \mathscr{O}(2^{n^2})$ such that if controller wins $\mathscr{G}$ under delay δ_t, then she wins $\mathscr{G}$ under every δ.*

3. *For every $n > 1$, there is a game $\mathscr{G}_n$ of size $\mathscr{O}(n^2)$ with ω-regular winning condition specified by a two-state deterministic parity automaton $\mathscr{A}_n$ such that controller wins $\mathscr{G}$ under delay 2^n, but not under delay $2^n + 2$.*

Note that the lower bound on δ_t is just a restatement of Theorem 4, as safety games have ω-regular winning conditions.

Linear Temporal Logic.   Finally, one can also transfer the triply-exponential upper and lower bounds on the necessary lookahead in delay games with LTL winning conditions as well as the 3EXPTIME-completeness of determining whether Player *O* wins such a delay game with respect to some lookahead [10] to games under delayed control with LTL winning conditions. Here, we exploit the following facts:

1. If the winning condition Win for a game $\mathscr{G}$ under delayed control is given by an LTL formula φ, then the winning condition $\overline{L(\mathscr{G})}$ is given by an LTL formula of size $\mathscr{O}(|\varphi|)$.

2. Dually, if the winning condition $L \subseteq (\Sigma_I \times \Sigma_O)^\omega$ of a delay game is given by an LTL formula φ, then the winning condition of the game $\mathscr{G}(\overline{L})$ under given action is given by an LTL formula of size $\mathscr{O}(|\varphi|)$.

Theorem 6.

1. *The following problem is* 3EXPTIME-*complete: Given a game $\mathscr{G}$ with winning condition specified by an LTL formula φ, does controller win $\mathscr{G}$ under every delay δ?*

2. *Let $\mathscr{G}$ be a game with ω-regular winning condition specified by an LTL formula φ. There is a $\delta_t \in \mathscr{O}(2^{2^{2^{|\varphi|+|\mathscr{G}|}}})$ such that if controller wins $\mathscr{G}$ under delay δ_t, then she wins $\mathscr{G}$ under every δ.*

3. *For every $n > 1$, there is a game $\mathscr{G}_n$ of size $\mathscr{O}(n^2)$ with winning condition specified by an LTL formula φ_n of size $\mathscr{O}(n^2)$ such that controller wins $\mathscr{G}$ under delay $2^{2^{2^n}}$, but not under delay $2^{2^{2^n}}+2$.*

To conclude, let us just remark that the results presented here also allow us to transfer results obtained for delay games with quantitative winning conditions [10, 13, 14] to games under delayed control with quantitative winning conditions. In fact, our result works for any winning condition, as long as the two transformations described in Section 3 are effective.

4.2   Environment's View

In Section 3, we proved a tight correspondence between controller in a game under delayed control and Player *I* in a delay game. Thus, it is natural to ask whether environment and Player *O* also share such a tight correspondence. A first indication that this is not the case can be obtained by considering the determinacy of these games: While delay games with Borel winning conditions are determined [8], even safety games under delayed action are not necessarily determined [2].

Upon closer inspection, this is not surprising, as the strategies in games under delayed control are not dual between the players: controller is at a disadvantage as she only gets delayed access to the actions picked by environment while environment does not benefit from this disadvantage. He does not get access to the actions picked by controller in advance. In a delay game however, the strategy definitions are completely dual: Player *I* has to grant lookahead on her moves which Player *O* gets access to. Thus, environment is in a weaker position than Player *O*.[1]

In this section, we study the correspondence between environment and Player *O* in detail by formally proving that environment is weaker than Player *O*.

Lemma 3. *Let $\mathscr{G}$ be a safety game. If environment wins $\mathscr{G}$ then Player O wins $\Gamma_k(\overline{L(\mathscr{G})})$ for every k.*

Now, we show that the converse direction fails.

---

[1]The difference can be formalized in terms of the information the players have access to: safety games under delay are incomplete-information games while delay games are complete-information games. Although interesting, we do not pursue this angle any further.
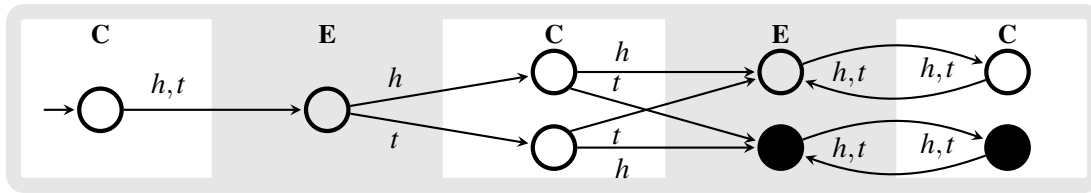
Figure 2: A safety game that environment does not win, but Player O wins the associated delay game. The initial state is marked by an arrow and the unsafe vertices are black. Note that both players have the actions h and t available.

Lemma 4. *There is a safety game $\mathcal{G}$ such that Player O wins $\Gamma_k(\overline{L(\mathcal{G})})$ for some k, but environment does not win $\mathcal{G}$.*

Proof. Let $\mathcal{G}$ be the safety game depicted in Fig. 2. With each move, the players place a coin (by either picking heads or tails) and environment wins a play by correctly predicting the second action of controller with his first action. Clearly, environment has no winning strategy in $\mathcal{G}$ because he has no access to future moves of controller. Stated differently, if environment picks h (t) in his first move, then the play in which the second action of controller is t (h) is winning for controller.[2]

Now, we consider the delay game $\Gamma_k(\overline{L(\mathcal{G})})$ for $k = 1$. Recall that the winning condition $\overline{L(\mathcal{G})}$ contains the winning plays for Player O, i.e., we have $\left(\begin{smallmatrix}\sigma_0\sigma_2\sigma_4\cdots\\\sigma_1\sigma_3\sigma_5\cdots\end{smallmatrix}\right) \in \overline{L(\mathcal{G})}$ if and only if $\sigma_1 \neq \sigma_2$. It is easy to see that Player O has a winning strategy in $\Gamma_k(\overline{L(\mathcal{G})})$ by simply flipping the second letter picked by Player I. This is possible since Player I has to provide two letters during the first round. □

Remark 9. *The safety game $\mathcal{G}$ depicted in Fig. 2 is in fact undetermined under every delay $\delta > 0$. In the proof of Lemma 4, we have already established that environment does not win $\mathcal{G}$. Now, under every delay $\delta > 0$, controller has to fix at least two actions before getting access to the first action picked by environment. This implies that there is, for every strategy for controller under delay δ, at least one consistent play that is losing for her, i.e., a play in which environment picks h (t) if the second move fixed by controller is t (h). Thus, no strategy is winning for controller under delay δ.*

Let us remark that, according to our definition of environment strategies, he is not able to enforce a losing play for controller (the game is undetermined after all), as he does not get access to the second action fixed by controller. Also, this is again the difference to delay games: Player O has access to these first two actions when making his first move, and is thereby able to win.

The full relation between games under delayed control and delay games is depicted in Fig. 3, restricted to Borel winning conditions (note that both transformations described in Section 3 preserve Borelness). The equivalence between controller winning the game under delayed control and Player I winning the corresponding delay game has been shown in Lemma 1 and Lemma 2. Also, Lemma 2 and Remark 7 imply that undetermined safety games under delayed control and those won by environment get transformed into delay games that are won by Player O. Finally, Lemma 1 and Remark 7 imply that delay games won by Player O get transformed into undetermined safety games under delayed control or to ones that are won by environment.

---

[2]Note that under any delay $\delta > 0$, controller cannot do this strategically, as she has to fix her first two actions in advance. But, as environment has no access to these fixed actions, he cannot react to them strategically.
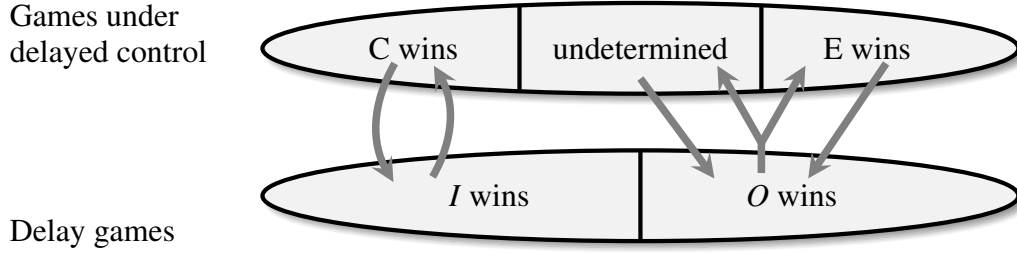
Figure 3: The relation between games under delayed control and delay games with Borel winning conditions. The upper ellipsis contains pairs $(\mathscr{G}, \delta)$ consisting of a game $\mathscr{G}$ under delayed control and a fixed delay δ; the lower one contains delay games $\Gamma_k(L)$ for some fixed k. The arrows represent the two transformations described in Section 3.

5   Refining the Correspondence: Sure Winning and Almost Sure Winning

It should be noted that the above transformations of games under delayed control into delay games and vice versa hinge on the fact that environment in the game under delayed control could, though lacking recent state information to do so strategically, by mere chance play the very same actions that the informed Player O in the delay game plays in his optimal adversarial strategy. That this constitutes a fundamental difference becomes apparent if we consider almost sure winning instead of sure winning. Almost sure winning calls for the existence of a mixed strategy that wins with probability 1, i.e., may fail on a set of plays with measure 0. This is different from sure winning in the sense of the definition of winning strategies for games under delayed control in Section 2.1, which calls for a strategy that never fails.

Remark 10. *We introduce mixed strategies for games under delayed control only, as delay games (with Borel winning conditions) are determined, which means that mixed strategies do not offer any advantage over pure strategies as introduced in Section 2.2.*

Given an even $\delta \geq 0$, a mixed strategy for controller in $\mathscr{G}$ under delay δ is a pair (α, τ_c) where $\alpha \in \mathscr{P}\left((\Sigma_c)^{\frac{\delta}{2}}\right)$ is a probability distribution over $(\Sigma_c)^{\frac{\delta}{2}}$ and $\tau_c \colon \mathrm{Pref}_c(\mathscr{G}) \to \mathscr{P}(\Sigma_c)$ maps play prefixes ending in S_c to probability distributions over actions of controller. A mixed strategy for environment is a mapping $\tau_e \colon \mathrm{Pref}_e(\mathscr{G}) \to \mathscr{P}(\Sigma_e)$.

The notion of consistency of a play with a strategy simply carries over, now inducing a Markov chain due to the probabilistic nature of the strategies. We say that a mixed strategy for controller (environment) *wins almost surely* if and only if it wins against any strategy of its opponent environment (controller) with probability 1, i.e., if and only if the winning condition is satisfied with probability 1 over the Markov chain induced by the game and the particular strategy combination. In this section, we write sure winning for winning as defined in Section 2, as is usual for games with randomized strategies.

The notion of almost sure winning alters chances for the players substantially by excluding the possibility of reliably playing an optimal strategy though lacking the information for doing so due to delayed observation. This can be seen from the following lemma, stating a fundamental difference between controller's power in games under delayed control and Player I's power in the corresponding delay games.

Lemma 5. *There is a game $\mathscr{G}$ under delayed control such that controller wins $\mathscr{G}$ almost surely under some delay δ while Player O (not Player I, which is the player corresponding to controller) wins the corresponding delay game $\Gamma_k(\overline{L(\mathscr{G})})$ for $k = \frac{\delta}{2}$, and surely so.*
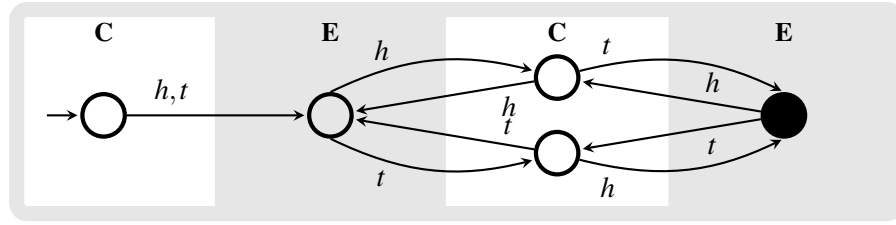
Figure 4: A reachability game that, under any positive delay, is won by controller almost surely via the simple randomized strategy of coin tossing (thus randomly generating head and tail events h and t), but won by player O surely if interpreted as a delay game due to the lookahead on Player I's actions granted to Player O. The initial state is marked by an arrow and controller wins if and only if the black vertex is visited at least once.

Proof. Consider the reachability game in Fig. 4 under delay 2 (or any larger delay). Intuitively, the players place a coin in each round (by picking either heads to tails with each move) and controller wins a play if the black state is visited, which happens if she selects a different coin placement than chosen by environment in the previous move.

Under any even (by definition) positive delay, controller wins this game with probability 1, i.e., almost surely, by a simple randomized strategy of coin tossing: by in each step randomly selecting action h or t with positive probability each, an eventual visit of the black state is guaranteed with probability 1, irrespective of being uninformed about environment's preceding move due to the delay.

The corresponding delay game $\Gamma_k(\overline{L(\mathcal{G})})$ for $k = \frac{\delta}{2}$, however, is easily won by Player O, because in delay games, the delayed Player I grants a lookahead to Player O. Hence, Player O can, due to the delay, already see the next move of Player I such that he can simply copy the next coin placement by Player I, safely staying in the non-black states and thereby win.   □

Note that Lemma 5 implies that the previously observed correspondence between Player I and controller breaks down when considering almost sure winning strategies instead of just sure winning strategies: Games under delayed control for which Player O wins the corresponding delay game, are no longer either undetermined or won by environment, but may well be won by controller almost surely.

This consequently refines the correspondence between games under delayed control and delay games shown in Fig. 3 as follows.

Theorem 7. *Given a game $\mathcal{G}$ and an even $\delta \geq 2$, the following correspondences between $\mathcal{G}$ and the corresponding delay game $\Gamma_k(\overline{L(\mathcal{G})})$ for $k = \frac{\delta}{2}$ hold:*

1. *Controller surely wins $\mathcal{G}$ under delay δ if and only if Player I surely wins $\Gamma_k(\overline{L(\mathcal{G})})$.*

2. *If controller almost surely wins $\mathcal{G}$ under delay δ but cannot surely win $\mathcal{G}$ under delay δ then Player O surely wins $\Gamma_k(\overline{L(\mathcal{G})})$.*

3. *If environment surely or almost surely wins $\mathcal{G}$ under delay δ then Player O wins $\Gamma_k(\overline{L(\mathcal{G})})$.*

4. *If $\mathcal{G}$ is undetermined under delay δ with respect to almost sure winning strategies then Player O wins $\Gamma_k(\overline{L(\mathcal{G})})$.*

5. *All the aforementioned classes are non-empty, i.e., there exist games under delayed control where controller wins, where controller wins almost surely (but not surely), where environment wins surely, where environment wins almost surely (but not surely), and games which are undetermined with respect to almost-sure winning strategies.*
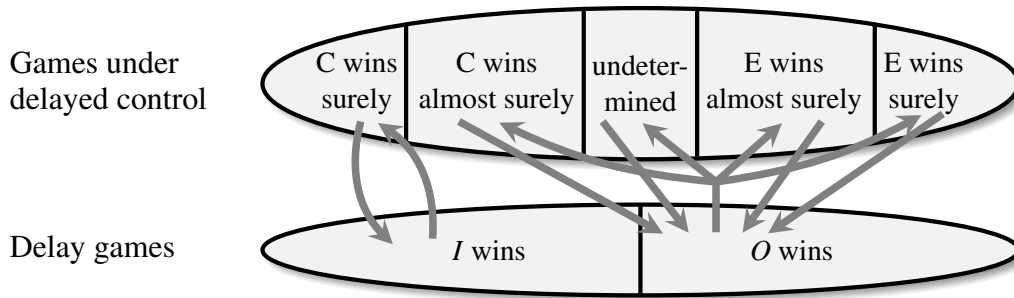
Figure 5: The relation between safety games under delayed control and delay games with Borel winning conditions. The upper ellipsis contains pairs $(\mathscr{G}, \delta)$ consisting of a game $\mathscr{G}$ under delayed control and a fixed delay δ; the lower one contains delay games $\Gamma_k(L)$ for some fixed k. The arrows represent the two transformations described in Section 3.

The above correspondences are depicted in Fig. 5.

Item 2. of the above lemma is of particular interest, as it expresses a delay-related strengthening of controller relative to Player I, letting controller win almost surely where Player I looses for sure. The correspondence between controller and Player I observed in the deterministic setting thus breaks down when almost sure winning is considered and mixed strategies are permitted.

Remark 11. *In contrast to games under delayed control, where mixed strategies provide additional power to both the controller and the environment, the notions of sure winning and almost sure winning coincide for delay games (with Borel winning conditions) due to their determinacy [8]. Admitting mixed strategies (and almost sure winning) does not provide additional power to either of the two players in a delay game, as the determinacy result always implies existence of an infallible pure strategy for one of the players.*

6   Conclusion

We have compared delay games [9] and games under delayed control [2], two types of infinite games aiming to model asynchronicity in reactive synthesis, and have exhibited the differences in definitions and charted the relation between them with respect to both deterministic and randomized strategies: One can efficiently transform a game under delayed control into a delay game such that controller wins the game under delayed control with delay δ by a deterministic strategy if and only if Player I wins the resulting delay game with lookahead of size $\frac{\delta}{2}$. Dually, one can efficiently transform a delay game into a game under delayed control such that Player I wins the delay game with lookahead of size δ if and only if controller wins the resulting game under delayed control with delay 2δ by a deterministic strategy. These results allow us to transfer known complexity results and bounds on the amount of delay from delay games to games under delayed control, for which no such results were known, when considering deterministic strategies. We also proved that the analogous results fail in the setting of randomized strategies and almost sure winning conditions, as well as for the relation between environment and Player O, both under deterministic and randomized strategies.

References

[1] J. Richard Büchi & Lawrence H. Landweber (1969): *Solving Sequential Conditions by Finite-State Strategies.* *Trans. Amer. Math. Soc.* 138, pp. pp. 295–311, doi:10.2307/1994916.

[2] Mingshuai Chen, Martin Fränzle, Yangjia Li, Peter Nazier Mosaad & Naijun Zhan (2021): *Indecision and delays are the parents of failure - taming them algorithmically by synthesizing delay-resilient control.* *Acta Informatica* 58(5), pp. 497–528, doi:10.1007/s00236-020-00374-7.

[3] Martin Fränzle, Sarah Winter & Martin Zimmermann (2023): *Strategies Resilient to Delay: Games under Delayed Control vs. Delay Games.* arXiv 2305.19985, doi:10.48550/arXiv.2305.19985.

[4] David Gale & Frank M. Stewart (1953): *Infinite games with perfect information.* Annals of Mathematics 28, pp. 245–266, doi:10.1515/9781400881970-014.

[5] Erich Grädel, Wolfgang Thomas & Thomas Wilke, editors (2002): *Automata, Logics, and Infinite Games: A Guide to Current Research.* *LNCS* 2500, Springer, doi:10.1007/3-540-36387-4.

[6] Michael Holtmann, Lukasz Kaiser & Wolfgang Thomas (2012): *Degrees of Lookahead in Regular Infinite Games.* *Log. Methods Comput. Sci.* 8(3), doi:10.2168/LMCS-8(3:24)2012.

[7] Frederick A. Hosch & Lawrence H. Landweber (1972): *Finite Delay Solutions for Sequential Conditions.* In Maurice Nivat, editor: *ICALP 1972*, North-Holland, Amsterdam, pp. 45–60.

[8] Felix Klein & Martin Zimmermann (2015): *What are Strategies in Delay Games? Borel Determinacy for Games with Lookahead.* In Stephan Kreutzer, editor: *CSL 2015*, *LIPIcs* 41, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 519–533, doi:10.4230/LIPIcs.CSL.2015.519.

[9] Felix Klein & Martin Zimmermann (2016): *How Much Lookahead is Needed to Win Infinite Games?* *Log. Methods Comput. Sci.* 12(3), doi:10.2168/LMCS-12(3:4)2016.

[10] Felix Klein & Martin Zimmermann (2016): *Prompt Delay.* In Akash Lal, S. Akshay, Saket Saurabh & Sandeep Sen, editors: *FSTTCS 2016*, *LIPIcs* 65, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 43:1–43:14, doi:10.4230/LIPIcs.FSTTCS.2016.43.

[11] Robert McNaughton (2000): *Playing Infinite Games in Finite Time.* In Arto Salomaa, Derick Wood & Sheng Yu, editors: *A Half-Century of Automata Theory: Celebration and Inspiration*, World Scientific, pp. 73–91.

[12] Amir Pnueli (1977): *The Temporal Logic of Programs.* In: *FOCS 1977*, IEEE Computer Society, pp. 46–57, doi:10.1109/SFCS.1977.32.

[13] Martin Zimmermann (2016): *Delay Games with WMSO+U Winning Conditions.* *RAIRO Theor. Informatics Appl.* 50(2), pp. 145–165, doi:10.1051/ita/2016018.

[14] Martin Zimmermann (2017): *Games with costs and delays.* In: *LICS 2017*, IEEE Computer Society, pp. 1–12, doi:10.1109/LICS.2017.8005125.

[15] Martin Zimmermann (2022): *Approximating the minimal lookahead needed to win infinite games.* Information Processing Letters 177, p. 106264, doi:10.1016/j.ipl.2022.106264.