Trading Infinite Memory for Uniform Randomness in Timed Games

Krishnendu Chatterjee
Thomas A. Henzinger
Vinayak Prabhu

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2008-4
http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-4.html

January 10, 2008

Copyright © 2008, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

Trading Infinite Memory for Uniform Randomness in Timed Games*

Krishnendu Chatterjee[1], Thomas A. Henzinger[1,2], and Vinayak S. Prabhu[1]

[1]EECS, UC Berkeley; [2]CCS, EPFL;
{c_krish,vinayak}@eecs.berkeley.edu, tah@epfl.ch

Abstract. We consider concurrent two-player timed automaton games with ω-regular objectives specified as parity conditions. These games offer an appropriate model for the synthesis of real-time controllers. Earlier works on timed games focused on pure strategies for each player. We study, for the first time, the use of *randomized* strategies in such games. While pure (i.e., nonrandomized) strategies in timed games require infinite memory for winning even with respect to reachability objectives, we show that randomized strategies can win with finite memory with respect to all parity objectives. Also, the synthesized randomized real-time controllers are much simpler in structure than the corresponding pure controllers, and therefore easier to implement. For safety objectives we prove the existence of pure finite-memory winning strategies. Finally, while randomization helps in simplifying the strategies required for winning timed parity games, we prove that randomization does not help in winning at more states.

1 Introduction

Timed automata [2] are models of real-time systems in which states consist of discrete locations and values for real-time clocks. The transitions between locations are dependent on the clock values. *Timed automaton games* [8, 1, 7, 13, 12] are used to distinguish between the actions of several players (typically a "controller" and a "plant"). We shall consider two-player timed automaton games with ω-regular objectives specified as *parity conditions*. The class of ω-regular objectives can express all safety and liveness specifications that arise in the synthesis and verification of reactive systems, and parity conditions are a canonical form to express ω-regular objectives [20]. The construction of a winning strategy for player 1 in such games corresponds to the *controller synthesis problem for real-time systems* [11, 16, 17, 21] with respect to achieving a desired ω-regular objective.

The issue of *time divergence* is crucial in timed games, as a naive control strategy might simply block time, leading to "zeno" runs. Such invalid solutions have often been avoided by putting strong syntactic constraints on the cycles of timed automaton games [17, 4, 13, 5], or by semantic conditions that discretize time [14]. Other works [16, 11, 6, 7] have required that time divergence be ensured by the controller —a one-sided, unfair view in settings where the player modeling the plant is allowed to block time. We use the more general, semantic and fully symmetric formalism of [8, 15] for dealing with the issue of time divergence. This setting places no syntactic restriction on the game structure, and gives both players equally powerful options for advancing time, but for a player to win, she must not be *responsible* for causing time to converge. It has been shown in [15] that this is equivalent to requiring that the players are restricted to the use of *receptive* strategies [3, 19], which, while being required to not prevent time from diverging, are not required to ensure time divergence. More formally, our timed games proceed in an infinite sequence of rounds. In each

* This research was supported in part by the NSF grants CCR-0208875, CCR-0225610, and CCR-0234690, by the Swiss National Science Foundation, and by the Artist2 European Network of Excellence.

round, both players simultaneously propose moves, with each move consisting of an action and a time delay after which the player wants the proposed action to take place. Of the two proposed moves, the move with the shorter time delay "wins" the round and determines the next state of the game. Let a set Φ of runs be the desired objective for player 1. Then player 1 has a *winning* strategy for Φ if she has a strategy to ensure that, no matter what player 2 does, one of the following two conditions hold: (1) time diverges and the resulting run belongs to Φ, or (2) time does not diverge but player-1's moves are chosen only finitely often (and thus she is not to be blamed for the convergence of time).

The winning strategies constructed in [8] for such timed automaton games assume the presence of an infinitely precise global clock to measure the progress of time, and the strategies crucially depend on the value of this global clock. Since the value of this clock needs to be kept in memory, the constructed strategies require *infinite memory.* In fact, the following example (Example 2) shows that infinite memory is necessary for winning with respect to reachability objectives. Besides the infinite-memory requirement, the strategies constructed in [8] are structurally complicated, and it would be difficult to implement the synthesized controllers in practice. Before offering a novel solution to this problem, we illustrate the problem with an example of a simple timed game whose solution requires infinite memory.

Example 1 (Signaling hub). Consider a signaling hub that both sends and receives signals at the same port. At any time the port can either receive or send a signal, but it cannot do both. Moreover, the hub must accept all signals sent to it. If both the input and the output signals arrive at the same time, then the output signal of the hub is discarded. The input signals are generated by other processes, and infinitely many signals cannot be generated in a finite amount of time. The time between input signals is not known a priori. The system may be modeled by the timed automaton game shown in Figure 1. The actions b_1 and b_2 correspond to input signals, and a_1 and a_2 to output signals. The actions b_i are controlled by the environment and denote input signals; the actions a_i are controlled by the hub and denote signals sent by the hub. The clock x models the time delay between signals: all signals reset this clock, and signals can arrive or be sent provided the value of x is greater than 0, ensuring that there is a positive delay between signals. The objective of the hub controller is to keep sending its own signals, which can be modeled as the generalized Büchi condition of switching infinitely often between the locations p and q (ie., the LTL objective $\Box(\Diamond p \wedge \Diamond q)$). □

Example 2 (Winning requires infinite memory). Consider the timed game of Figure 1. We let κ denote the valuation of the clock x. We let the special "action" $\perp$ denote a time move (representing time passage without an action). The objective of player 1 is to reach q starting from $s_0 = \langle p, x = 0\rangle$ (and similarly, to reach p from q). We let π_1 denote the strategy of player 1 which prescribes moves based on the history $r[0..k]$ of the game at stage k. Suppose player 1 uses only finite memory. Then player 1 can propose only moves from a finite set when at s_0. Since a zero time move keeps the game at p, we may assume that player 1 does not choose such moves. Let $\Delta > 0$ be the least time delay of these finitely many moves of player 1. Then player 2 can always propose a move $\langle \Delta/2, b\rangle$ when at s_0. This strategy will prevent player 1 from reaching q, and yet time diverges. Hence player 1 cannot win with finite memory; that is, there is no hub controller that uses only finite memory. However, player 1 has a winning strategy with infinite memory. For example, consider the player 1 strategy π_2 such that $\pi_2(r[0..k]) = \langle 1/2^{k+2}, a_1\rangle$ if $r[k] = \langle p, \kappa\rangle$. and $\pi_2(r[0..k]) = \langle 1, \perp\rangle$ otherwise. □

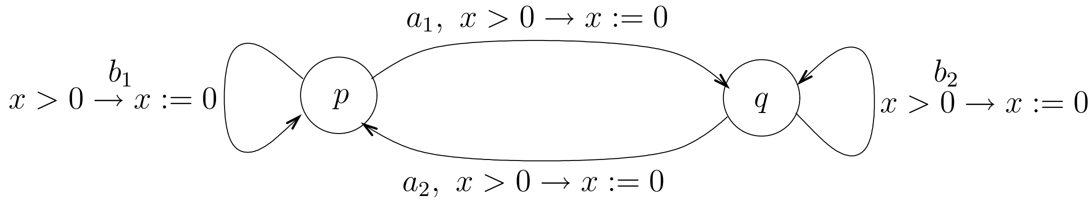

Fig. 1. A timed automaton game.

In this paper we observe that the infinite-memory requirement of Example 1 is due to the determinism of the permissible strategies: a strategy is *deterministic* (or *pure*) if in each round of the game, it proposes a unique move (i.e., action and time delay). A more general class of strategies are the randomized strategies: a *randomized* strategy may propose, in each round, a probability distribution of moves. We now show that in the game of Example 2 finite-memory randomized winning strategies do exist. Indeed, the needed randomization has a particularly simple form: player 1 proposes a unique action together with a time interval from which the time delay is chosen uniformly at random. Such a strategy can be implemented as a controller that has the ability to wait for a randomly chosen amount of time.

Example 3 (Randomization instead of infinite memory). Recall the game in Figure 1. Player 1 can play a randomized memoryless strategy π_3 such that $\pi_3(\langle p, \kappa\rangle) = \langle \mathsf{Uniform}((0, 1-\kappa(x))), a_i\rangle$; that is, the action a_i is proposed to take place at a time chosen uniformly at random in the interval $(0, 1-\kappa(x))$. Suppose player 2 always proposes the action b_i with varying time delays Δ_j at round j. Then the probability of player-1's move being never chosen is $\prod_{j=1}^{\infty}(1 - \Delta_j)$, which is 0 if $\sum_{j=1}^{\infty} \Delta_j = \infty$ (by Lemma 15). Interrupting moves with pure time moves does not help player 2, as $1 - \frac{\Delta_j}{1-\kappa(x)} < 1 - \Delta_j$. Thus the simple randomized strategy π_3 is winning for player 1 with probability 1. □

Previously, only deterministic strategies were studied for timed games; here, for the first time, we study randomized strategies. We show that randomized strategies are not more powerful than deterministic strategies in the sense that if player 1 can win with a randomized strategy, then she can also win with a deterministic strategy. However, as the example illustrated, randomization can lead to a reduction in the memory required for winning, and to a significant simplification in the structure of winning strategies. Randomization is therefore not only of theoretical interest, but can improve the implementability of synthesized controllers. It is for this reason that we set out, in this paper, to systematically analyze the trade-off between randomization requirements (no randomization; uniform randomization; general randomization), memory requirements (finite memory and infinite memory) and the presence of extra "controller clocks" for various classes of ω-regular objectives (safety; reachability; parity objectives).

Our results are as follows. First, we show that for safety objectives pure (no randomization) finite-memory winning strategies exist. Next, for reachability objectives, we show that pure (no randomization) strategies require infinite memory for winning, whereas uniform randomized finite-memory winning strategies exist. We then use the results for reachability and safety objectives in an inductive argument to show that uniform randomized finite-memory strategies suffice for all parity objectives, for which pure strategies require infinite memory (because reachability is a special case of parity). In all our uses of randomization, we only use uniform randomization over time, and more general forms of randomization (nonuniform distributions; randomized actions) are not required. This shows that in timed games, infinite memory can be traded against uniform randomness. Finally, we show that while randomization helps in simplifying winning strategies, and thus allows the construction of simpler controllers, randomization does not help a player in winning at more states, and thus does not allow the construction of more powerful controllers. In other words, the case for randomness rests in the simplicity of the synthesized real-time controllers, not in their expressiveness.

We note that in our setting, player 1 (i.e., the controller) can trade infinite memory also against finite memory together with an extra clock. We assume that the values of all clocks of the plant are observable. For an ω-regular objective Φ, we define the following winning sets depending on the power given to player 1: let $[\![\Phi]\!]_1$ be the set of states from which player 1 can win using any strategy (finite or infinite memory; pure or randomized) and any number of infinitely precise clocks; in $[\![\Phi]\!]_2$ player 1 can win using a pure finite-memory strategy and only one extra clock; in $[\![\Phi]\!]_3$ player 1 can win using a pure finite-memory strategy and no extra clock; and in $[\![\Phi]\!]_4$ player 1 can win using a randomized finite-memory strategy and no extra clock. Then, for every timed automaton game, we have $[\![\Phi]\!]_1 = [\![\Phi]\!]_2 = [\![\Phi]\!]_4$. We also have $[\![\Phi]\!]_3 \subseteq [\![\Phi]\!]_1$, with the subset inclusion being in general strict. It can be shown that at least one bit of memory is required for winning of reachability objectives despite player 1 being allowed randomized strategies. We do not know whether memory is required for winning safety objectives (even in the case of pure strategies).

We note that removing the global clock from winning strategies is nontrivial. The algorithm of [8] uses such a global clock in a μ-calculus formulation to construct winning strategies. Without a global clock, time cannot be measured directly, and we need to argue about other properties of runs which ensure time divergence. For safety objectives, we construct a formula that depends only on clock resets and on particular region valuations, and we argue that the satisfaction of that formula is both necessary and sufficient for winning. This allows us to construct pure finite-memory winning strategies for safety objectives. For reachability objectives, we construct "ranks" for sets of states of a μ-calculus formula, and use these ranked sets to obtain a randomized finite-memory strategy for winning. The proof requires special care, because our winning strategies are required to be invariant over the values of the global clock. Finally, we show that if player 1 does not have a pure (possibly infinite-memory) winning strategy from a state, then for every $\varepsilon > 0$ and for every randomized strategy of player 1, player 2 has a pure counter strategy that can ensure with probability at least $1 - \varepsilon$ that player 1 does not win. This shows that randomization does not help in winning at more states.

2 Timed Games

In this section we present the definitions of timed game structures, runs, objectives, strategies and the notions of sure and almost-sure winning in timed game structures.

Timed game structures. A *timed game structure* is a tuple $\mathcal{G} = \langle S, A_1, A_2, \Gamma_1, \Gamma_2, \delta \rangle$ with the following components.

- S is a set of states.
- A_1 and A_2 are two disjoint sets of actions for players 1 and 2, respectively. We assume that $\perp \notin A_i$, and write $A_i^\perp$ for $A_i \cup \{\perp\}$. The set of *moves* for player i is $M_i = \mathbb{R}_{\geq 0} \times A_i^\perp$. Intuitively, a move $\langle \Delta, a_i \rangle$ by player i indicates a waiting period of Δ time units followed by a discrete transition labeled with action a_i.

- $\Gamma_i : S \mapsto 2^{M_i} \setminus \emptyset$ are two move assignments. At every state s, the set $\Gamma_i(s)$ contains the moves that are available to player i. We require that $\langle 0, \bot \rangle \in \Gamma_i(s)$ for all states $s \in S$ and $i \in \{1, 2\}$. Intuitively, $\langle 0, \bot \rangle$ is a time-blocking stutter move.
- $\delta : S \times (M_1 \cup M_2) \mapsto S$ is the transition function. We require that for all time delays $\Delta, \Delta' \in \mathbb{R}_{\geq 0}$ with $\Delta' \leq \Delta$, and all actions $a_i \in A_i^\bot$, we have (1) $\langle \Delta, a_i \rangle \in \Gamma_i(s)$ iff both $\langle \Delta', \bot \rangle \in \Gamma_i(s)$ and $\langle \Delta - \Delta', a_i \rangle \in \Gamma_i(\delta(s, \langle \Delta', \bot \rangle))$; and (2) if $\delta(s, \langle \Delta', \bot \rangle) = s'$ and $\delta(s', \langle \Delta - \Delta', a_i \rangle) = s''$, then $\delta(s, \langle \Delta, a_i \rangle) = s''$.

The game proceeds as follows. If the current state of the game is s, then both players simultaneously propose moves $\langle \Delta_1, a_1 \rangle \in \Gamma_1(s)$ and $\langle \Delta_2, a_2 \rangle \in \Gamma_2(s)$. The move with the shorter duration "wins" in determining the next state of the game. If both moves have the same duration, then player 2 determines whether the next state will be determined by its move, or by the move of player 1. We use this setting as our goal is to compute the winning set for player 1 against all possible strategies of player 2. Formally, we define the *joint destination function* $\delta_{\mathsf{jd}} : S \times M_1 \times M_2 \mapsto 2^S$ by

$$\delta_{\mathsf{jd}}(s, \langle \Delta_1, a_1 \rangle, \langle \Delta_2, a_2 \rangle) = \begin{cases} \{\delta(s, \langle \Delta_1, a_1 \rangle)\} & \text{if } \Delta_1 < \Delta_2; \\ \{\delta(s, \langle \Delta_2, a_2 \rangle)\} & \text{if } \Delta_2 < \Delta_1; \\ \{\delta(s, \langle \Delta_2, a_2 \rangle), \delta(s, \langle \Delta_1, a_1 \rangle)\} & \text{if } \Delta_2 = \Delta_1. \end{cases}$$

The time elapsed when the moves $m_1 = \langle \Delta_1, a_1 \rangle$ and $m_2 = \langle \Delta_2, a_2 \rangle$ are proposed is given by $\mathsf{delay}(m_1, m_2) = \min(\Delta_1, \Delta_2)$. The boolean predicate $\mathsf{blame}_i(s, m_1, m_2, s')$ indicates whether player i is "responsible" for the state change from s to s' when the moves m_1 and m_2 are proposed. Denoting the opponent of player i by $\sim i = 3 - i$, for $i \in \{1, 2\}$, we define

$$\mathsf{blame}_i(s, \langle \Delta_1, a_1 \rangle, \langle \Delta_2, a_2 \rangle, s') \;=\; \big(\Delta_i \leq \Delta_{\sim i} \;\wedge\; \delta(s, \langle \Delta_i, a_i \rangle) = s'\big).$$

Runs. A *run* of the timed game structure $\mathcal{G}$ is an infinite sequence $r = s_0, \langle m_1^0, m_2^0 \rangle, s_1, \langle m_1^1, m_2^1 \rangle, \ldots$ such that $s_k \in S$ and $m_i^k \in \Gamma_i(s_k)$ and $s_{k+1} \in \delta_{\mathsf{jd}}(s_k, m_1^k, m_2^k)$ for all $k \geq 0$ and $i \in \{1, 2\}$. For $k \geq 0$, let $\mathsf{time}(r, k)$ denote the "time" at position k of the run, namely, $\mathsf{time}(r, k) = \sum_{j=0}^{k-1} \mathsf{delay}(m_1^j, m_2^j)$ (we let $\mathsf{time}(r, 0) = 0$). By $r[k]$ we denote the $(k+1)$-th state s_k of r. The run prefix $r[0..k]$ is the finite prefix of the run r that ends in the state s_k. Let Runs be the set of all runs of $\mathcal{G}$, and let $\mathsf{FinRuns}$ be the set of run prefixes.

Objectives. An *objective* for the timed game structure $\mathcal{G}$ is a set $\Phi \subseteq \mathsf{Runs}$ of runs. We will be interested in the classical reachability, safety and parity objectives. Parity objectives are canonical forms for ω-regular properties that can express all commonly used specifications that arise in verification.

- Given a set of states Y, the *reachability* objective $\mathsf{Reach}(Y)$ is defined as the set of runs that visit Y, formally, $\mathsf{Reach}(Y) = \{r \mid \text{ there exists } i \text{ such that } r[i] \in Y\}$.
- Given a set of states Y, the *safety* objective consists of the set of runs that stay within Y, formally, $\mathsf{Safe}(Y) = \{r \mid \text{ for all } i \text{ we have } r[i] \in Y\}$.
- Let $\Omega : S \mapsto \{0, \ldots, k-1\}$ be a parity index function. The parity objective for Ω requires that the maximal index visited infinitely often is even. Formally, let $\mathsf{InfOften}(\Omega(r))$ denote the set of indices visited infinitely often along a run r. Then the parity objective defines the following set of runs: $\mathsf{Parity}(\Omega) = \{r \mid \max(\mathsf{InfOften}(\Omega(r))) \text{ is even } \}$.

A timed game structure $\mathcal{G}$ together with the index function Ω constitute a *parity timed game* (of index k) in which the objective of player 1 is $\mathsf{Parity}(\Omega)$. We use similar notations for reachability and safety timed games.

Strategies. A *strategy* for a player is a recipe that specifies how to extend a run. Formally, a *probabilistic strategy* π_i for player $i \in \{1,2\}$ is a function π_i that assigns to every run prefix $r[0..k]$ a probability distribution $\mathcal{D}_i(r[k])$ over $\Gamma_i(r[k])$, the set of moves available to player i at the state $r[k]$. *Pure strategies* are strategies for which the state space of the probability distribution of $\mathcal{D}_i(r[k])$ is a singleton set for every run r and all k. We let Π_i^{pure} denote the set of pure strategies for player i, with $i \in \{1,2\}$. For $i \in \{1,2\}$, let Π_i be the set of strategies for player i. If both both players propose the same time delay, then the tie is broken by a *scheduler*. Let $\mathsf{TieBreak}$ be the set of functions from $\mathbb{R}_{\geq 0}$ to $\{1,2\}$. A *scheduler strategy* π_{sched} is a mapping from $\mathsf{FinRuns}$ to $\mathsf{TieBreak}$. If $\pi_{\mathsf{sched}}(r[0..k]) = h$, then the resulting state given player 1 and player 2 moves $\langle \Delta, a_1 \rangle$ and $\langle \Delta, a_2 \rangle$ respectively, is determined by the move of player $h(\Delta)$. We denote the set of all scheduler strategies by Π_{sched}. Given two strategies $\pi_1 \in \Pi_1$ and $\pi_2 \in \Pi_2$, the set of possible *outcomes* of the game starting from a state $s \in S$ is denoted $\mathsf{Outcomes}(s, \pi_1, \pi_2)$. Given strategies π_1 and π_2, for player 1 and player 2, respectively, a scheduler strategy π_{sched} and a starting state s we denote by $\Pr_s^{\pi_1,\pi_2,\pi_{\mathsf{sched}}}(\cdot)$ the probability space given the strategies and the initial state s.

Receptive strategies. We will be interested in strategies that are meaningful (in the sense that they do not block time). To define them formally we first present the following two sets of runs.

- A run r is *time-divergent* if $\lim_{k\to\infty} \mathsf{time}(r,k) = \infty$. We denote by $\mathsf{Timediv}$ the set of all time-divergent runs.
- The set $\mathsf{Blameless}_i \subseteq \mathsf{Runs}$ consists of the set of runs in which player i is responsible only for finitely many transitions. A run $s_0, \langle m_1^0, m_2^0 \rangle, s_1, \langle m_1^1, m_2^1 \rangle, \ldots$ belongs to the set $\mathsf{Blameless}_i$, for $i = \{1,2\}$, if there exists a $k \geq 0$ such that for all $j \geq k$, we have $\neg\, \mathsf{blame}_i(s_j, m_1^j, m_2^j, s_{j+1})$.

A strategy π_i is *receptive* if for all strategies $\pi_{\sim i}$, all states $s \in S$, and all runs $r \in \mathsf{Outcomes}(s, \pi_1, \pi_2)$, either $r \in \mathsf{Timediv}$ or $r \in \mathsf{Blameless}_i$. Thus, no what matter what the opponent does, a receptive strategy of player i cannot be responsible for blocking time. Strategies that are not receptive are not physically meaningful. A timed game structure $\mathcal{G}$ is *well-formed* if both players have receptive strategies. We restrict our attention to well-formed timed game structures. We denote Π_i^R to be the set of receptive strategies for player i. Note that for $\pi_1 \in \Pi_1^R, \pi_2 \in \Pi_2^R$, we have $\mathsf{Outcomes}(s, \pi_1, \pi_2) \subseteq \mathsf{Timediv}$.

Sure and almost-sure winning modes. Let $\mathsf{Sure}_1^{\mathcal{G}}(\Phi)$ (resp. $\mathsf{AlmostSure}_1^{\mathcal{G}}(\Phi)$) be the set of states s in $\mathcal{G}$ such that player 1 has a receptive strategy $\pi_1 \in \Pi_1^R$ such that for all scheduler strategies $\pi_{\mathsf{sched}} \in \Pi_{\mathsf{sched}}$ and for all player-2 receptive strategies $\pi_2 \in \Pi_2^R$, we have $\mathsf{Outcomes}(s, \pi_1, \pi_2) \subseteq \Phi$ (resp. $\Pr_s^{\pi_1,\pi_2,\pi_{\mathsf{sched}}}(\Phi) = 1$). Such a winning strategy is said to be a sure (resp. almost sure) winning receptive strategy. In computing the winning sets, we shall quantify over *all* strategies, but modify the objective to take care of time divergence. Given an objective Φ, let $\mathsf{TimeDivBl}_1(\Phi) = (\mathsf{Timediv} \cap \Phi) \cup (\mathsf{Blameless}_1 \setminus \mathsf{Timediv})$, i.e., $\mathsf{TimeDivBl}_1(\Phi)$ denotes the set of paths such that either time diverges and Φ holds, or else time converges and player 1 is not responsible for time to converge. Let $\underline{\mathsf{Sure}}_1^{\mathcal{G}}(\Phi)$ (resp. $\underline{\mathsf{AlmostSure}}_1^{\mathcal{G}}(\Phi)$) be the set of states in $\mathcal{G}$ such that for all $s \in \underline{\mathsf{Sure}}_1^{\mathcal{G}}(\Phi)$ (resp. $\underline{\mathsf{AlmostSure}}_1^{\mathcal{G}}(\Phi)$), player 1 has a strategy $\pi_1 \in \Pi_1$ such that for all strategies for all scheduler strategies $\pi_{\mathsf{sched}} \in \Pi_{\mathsf{sched}}$ and for all player-2 strategies $\pi_2 \in \Pi_2$, we have $\mathsf{Outcomes}(s, \pi_1, \pi_2) \subseteq \Phi$ (resp. $\Pr_s^{\pi_1,\pi_2,\pi_{\mathsf{sched}}}(\Phi) = 1$). Such a winning strategy is said to be a sure (resp. almost sure) winning for the non-receptive game. The following result establishes the connection between Sure and $\underline{\mathsf{Sure}}$ sets.

Theorem 1 ([15]). *For all well-formed timed game structures $\mathcal{G}$, and for all ω-regular objectives Φ, we have $\underline{\mathsf{Sure}}_1^{\mathcal{G}}(\mathsf{TimeDivBl}_1(\Phi)) = \mathsf{Sure}_1^{\mathcal{G}}(\Phi)$.*

We now define a special class of timed game structures, namely, timed automaton games.

Timed automaton games. Timed automata [2] suggest a finite syntax for specifying infinite-state timed game structures. A *timed automaton game* is a tuple $\mathcal{T} = \langle L, C, A_1, A_2, E, \gamma \rangle$ with the following components:

- L is a finite set of locations.
- C is a finite set of clocks.
- A_1 and A_2 are two disjoint sets of actions for players 1 and 2, respectively.
- $E \subseteq L \times (A_1 \cup A_2) \times \mathsf{Constr}(C) \times L \times 2^C$ is the edge relation, where the set $\mathsf{Constr}(C)$ of *clock constraints* is generated by the grammar

  $$\theta ::= x \leq d \mid d \leq x \mid \neg\theta \mid \theta_1 \wedge \theta_2$$

  for clock variables $x \in C$ and nonnegative integer constants d. For an edge $e = \langle l, a_i, \theta, l', \lambda \rangle$, the clock constraint θ acts as a guard on the clock values which specifies when the edge e can be taken, and by taking the edge e, the clocks in the set $\lambda \subseteq C$ are reset to 0. We require that for all edges $\langle l, a_i, \theta', l', \lambda' \rangle, \langle l, a_i, \theta'', l'', \lambda'' \rangle \in E$ with $l' \neq l''$, the conjunction $\theta' \wedge \theta''$ is unsatisfiable. This requirement ensures that a state and a move together uniquely determine a successor state.
- $\gamma : L \mapsto \mathsf{Constr}(C)$ is a function that assigns to every location an invariant for both players. All clocks increase uniformly at the same rate. When at location l, each player i must propose a move out of l before the invariant $\gamma(l)$ expires. Thus, the game can stay at a location only as long as the invariant is satisfied by the clock values.

A *clock valuation* is a function $\kappa : C \mapsto \mathbb{R}_{\geq 0}$ that maps every clock to a nonnegative real. The set of all clock valuations for C is denoted by $K(C)$. Given a clock valuation $\kappa \in K(C)$ and a time delay $\Delta \in \mathbb{R}_{\geq 0}$, we write $\kappa + \Delta$ for the clock valuation in $K(C)$ defined by $(\kappa + \Delta)(x) = \kappa(x) + \Delta$ for all clocks $x \in C$. For a subset $\lambda \subseteq C$ of the clocks, we write $\kappa[\lambda := 0]$ for the clock valuation in $K(C)$ defined by $(\kappa[\lambda := 0])(x) = 0$ if $x \in \lambda$, and $(\kappa[\lambda := 0])(x) = \kappa(x)$ if $x \notin \lambda$. A clock valuation $\kappa \in K(C)$ *satisfies* the clock constraint $\theta \in \mathsf{Constr}(C)$, written $\kappa \models \theta$, if the condition θ holds when all clocks in C take on the values specified by κ. A *state* $s = \langle l, \kappa \rangle$ of the timed automaton game $\mathcal{T}$ is a location $l \in L$ together with a clock valuation $\kappa \in K(C)$ such that the invariant at the location is satisfied, that is, $\kappa \models \gamma(l)$. Let S be the set of all states of $\mathcal{T}$. In a state, each player i proposes a time delay allowed by the invariant map γ, together either with the action $\perp$, or with an action $a_i \in A_i$ such that an edge labeled a_i is enabled after the proposed time delay. We require that for $i \in \{1, 2\}$ and for all states $s = \langle l, \kappa \rangle$, if $\kappa \models \gamma(l)$, either $\kappa + \Delta \models \gamma(l)$ for all $\Delta \in \mathbb{R}_{\geq 0}$, or there exist a time delay $\Delta \in \mathbb{R}_{\geq 0}$ and an edge $\langle l, a_i, \theta, l', \lambda \rangle \in E$ such that (1) $a_i \in A_i$ and (2) $\kappa + \Delta \models \theta$ and for all $0 \leq \Delta' \leq \Delta$, we have $\kappa + \Delta' \models \gamma(l)$, and (3) $(\kappa + \Delta)[\lambda := 0] \models \gamma(l')$. This requirement is necessary (but not sufficient) for well-formedness of the game. Given a timed automaton game $\mathcal{T}$, the definition of an associated timed game structure $[\![\mathcal{T}]\!]$ is standard [8]. We shall restrict our attention to randomization over time — a random move of a player will consist of a distribution over time over some interval I, denoted $\mathcal{D}^I$, together with a discrete action a_i.

Clock region equivalence. Timed automaton games can be solved using a region construction from the theory of timed automata [2]. For a real $t \geq 0$, let $\mathsf{frac}(t) = t - \lfloor t \rfloor$ denote the fractional part of t. Given a timed automaton game $\mathcal{T}$, for each clock $x \in C$, let c_x denote the largest integer constant that appears in any clock constraint involving x in $\mathcal{T}$ (let $c_x = 1$ if there is no clock constraint involving x). Two clock valuations κ_1, κ_2 are said to be *region equivalent*, denoted by $\kappa_1 \cong \kappa_2$ when all the following conditions hold: (a) for all clocks x with $\kappa_1(x) \leq c_x$ and

$\kappa_2(x) \leq c_x$, we have $\lfloor \kappa_1(x) \rfloor = \lfloor \kappa_2(x) \rfloor$; (b) for all clocks x, y with $\kappa_i(x) \leq c_x$ and $\kappa_i(y) \leq c_y$, we have $\mathsf{frac}(\kappa_1(x)) \leq \mathsf{frac}(\kappa_1(y))$ iff $\mathsf{frac}(\kappa_2(x)) \leq \mathsf{frac}(\kappa_2(y))$; (c) for all clocks x with $\kappa_1(x) \leq c_x$ and $\kappa_2(x) \leq c_x$, we have $\mathsf{frac}(\kappa_1(x)) = 0$ iff $\mathsf{frac}(\kappa_2(x)) = 0$; and (d) for any clock x, $\kappa_1(x) > c_x$ iff $\kappa_2(x) > c_x$. Two states $\langle \kappa_1, l_1 \rangle$ and $\langle \kappa_1, l_1 \rangle$ are region equivalent iff $l_1 = l_2$ and $\kappa_1 \cong \kappa_2$. A *region* R of a timed automaton game $\mathcal{T}$ is an equivalence class of states with respect to the region equivalence relation. We find it useful to sometimes denote a region R by a tuple $\langle l, h, \mathcal{P}(C) \rangle$ where (a) l is a location of $\mathcal{T}$; (b) h is a function which specifies the integer values of clocks $h : C \to (\mathbb{N} \cap [0, M])$ (M is the largest constant in $\mathcal{T}$); and (c) $\mathcal{P}(C)$ is a disjoint partition of the clocks $\{C_{-1}, C_0, \ldots C_n \mid \uplus C_i = C, C_i \neq \emptyset \text{ for } i > 0\}$. A state s with clock valuation κ is then in the region R when all the following conditions hold: (a) the location of s corresponds to the location of R; (b) for all clocks x with $\kappa(x) \leq c_x$, $\lfloor \kappa(x) \rfloor = h(x)$; (c) for $\kappa(x) > c_x$, $h(x) = c_x$; (d) for all pair of clocks (x, y), with $\kappa(x) \leq c_x$ and $\kappa(y) \leq c_y$, we have $\mathsf{frac}(\kappa(x)) < \mathsf{frac}(\kappa(y))$ iff $x \in C_i$ and $y \in C_j$ with $0 \leq i < j$ (so, $x, y \in C_k$ with $k \geq 0$ implies $\mathsf{frac}(\kappa(x)) = \mathsf{frac}(\kappa(y))$); (e) for $\kappa(x) \leq c_x$, $\mathsf{frac}(\kappa(x)) = 0$ iff $x \in C_0$; and (f) $x \in C_{-1}$ iff $\kappa(x) > c_x$. There are finitely many clock regions; more precisely, the number of clock regions is bounded by $|L| \cdot \prod_{x \in C}(c_x + 1) \cdot |C|! \cdot 2^{|C|}$.

For a state $s \in S$, we write $\mathsf{Reg}(s) \subseteq S$ for the clock region containing s. For a run r, we let the *region sequence* $\mathsf{Reg}(r) = \mathsf{Reg}(r[0]), \mathsf{Reg}(r[1]), \cdots$. Two runs r, r' are region equivalent if their region sequences are the same. Given a distribution $\mathcal{D}_{\mathsf{states}}$ over states, we obtain a corresponding distribution $\mathcal{D}_{\mathsf{reg}} = \mathsf{Reg}_d(\mathcal{D}_{\mathsf{states}})$ over regions as follows: for a region R we have $\mathcal{D}_{\mathsf{reg}}(R) = \mathcal{D}_{\mathsf{states}}(\{s \mid s \in R\})$. An ω-regular objective Φ is a region objective if for all region-equivalent runs r, r', we have $r \in \Phi$ iff $r' \in \Phi$. A strategy π_1 is a *region strategy*, if for all prefixes r_1 and r_2 such that $\mathsf{Reg}(r_1) = \mathsf{Reg}(r_2)$, we have $\mathsf{Reg}_d(\pi_1(r_1)) = \mathsf{Reg}_d(\pi_1(r_2))$. The definition for player 2 strategies is analogous. Two region strategies π_1 and π_1' are region-equivalent if for all prefixes r we have $\mathsf{Reg}_d(\pi_1(r)) = \mathsf{Reg}_d(\pi_1'(r))$. A parity index function Ω is a region parity index function if $\Omega(s_1) = \Omega(s_2)$ whenever $s_1 \cong s_2$. Henceforth, we shall restrict our attention to region objectives.

Encoding Time-Divergence by Enlarging the Game Structure. Given a timed automaton game $\mathcal{T}$, consider the enlarged game structure $\widehat{\mathcal{T}}$ with the state space $\widehat{S} \subseteq S \times \mathbb{R}_{[0,1)} \times \{\text{TRUE}, \text{FALSE}\}^2$, and an augmented transition relation $\widehat{\delta} : \widehat{S} \times (M_1 \cup M_2) \mapsto \widehat{S}$. In an augmented state $\langle s, \mathfrak{z}, tick, bl_1 \rangle \in \widehat{S}$, the component $s \in S$ is a state of the original game structure $[\![\mathcal{T}]\!]$, $\mathfrak{z}$ is value of a fictitious clock z which gets reset to 0 every time it hits 1, or if a move of player 1 is chosen, *tick* is true iff z hit 1 at last transition and bl_1 is true if player 1 is to blame for the last transition. Note that any strategy π_i in $[\![\mathcal{T}]\!]$, can be considered a strategy in $\widehat{\mathcal{T}}$. The values of the clock z, *tick* and bl_1 correspond to the values each player keeps in memory in constructing his strategy. Any run r in $\mathcal{T}$ has a corresponding unique run $\widehat{r}$ in $\widehat{\mathcal{T}}$ with $\widehat{r}[0] = \langle r[0], 0, \text{FALSE}, \text{FALSE} \rangle$ such that r is a projection of $\widehat{r}$ onto $\mathcal{T}$. For an objective Φ, we can now encode time-divergence as: $\mathsf{TimeDivBl}(\Phi) = (\Box\Diamond\, tick \to \Phi) \ \wedge\ (\neg\Box\Diamond\, tick \to \Diamond\Box\neg\, bl_1)$. Let $\widehat{\kappa}$ be a valuation for the clocks in $C \cup \{z\}$. A state of $\widehat{\mathcal{T}}$ can then be considered as $\langle \langle l, \widehat{\kappa} \rangle, tick, bl_1 \rangle$. We extend the clock equivalence relation to these expanded states: $\langle \langle l, \widehat{\kappa} \rangle\, tick, bl_1 \rangle \cong \langle \langle l', \widehat{\kappa}' \rangle, tick', bl_1' \rangle$ iff $l = l'$, $tick = tick'$, $bl_1 = bl_1'$ and $\widehat{\kappa} \cong \widehat{\kappa}'$. For every ω-regular region objective Φ of $\mathcal{T}$, we have $\mathsf{TimeDivBl}(\Phi)$ to be an ω-regular region objective of $\widehat{\mathcal{T}}$.

We now present a lemma that states for region ω-regular objectives region winning strategies exist, and all strategies region-equivalent to a region winning strategy are also winning. (see Appendix for proof).

Lemma 1. *Let $\mathcal{T}$ be a timed automaton game and $\widehat{\mathcal{T}}$ be the corresponding enlarged game structure. Let $\widehat{\Phi}$ be an ω-regular region objective of $\widehat{\mathcal{T}}$. Then the following assertions hold.*

- *There is a pure finite-memory region strategy π_1 that is sure winning for $\widehat{\Phi}$ from the states in $\underline{\mathsf{Sure}}_1^{\widehat{\mathcal{T}}}(\widehat{\Phi})$.*
- *If π_1 is a pure region strategy that is sure winning for $\widehat{\Phi}$ from $\underline{\mathsf{Sure}}_1^{\widehat{\mathcal{T}}}(\widehat{\Phi})$ and π_1' is a pure strategy that is region-equivalent to π_1, then π_1' is a sure winning strategy for $\widehat{\Phi}$ from $\underline{\mathsf{Sure}}_1^{\widehat{\mathcal{T}}}(\widehat{\Phi})$.*

Note that there is an infinitely precise global clock z in the enlarged game structure $\widehat{\mathcal{T}}$. If $\mathcal{T}$ does not have such a global clock, then strategies in $\widehat{\mathcal{T}}$ correspond to strategies in $\mathcal{T}$ where player 1 (and player 2) maintain the value of the infinitely precise global clock in memory (requiring infinite memory).

3 Safety Objectives: Pure Finite-memory Receptive Strategies Suffice

In this section we show the existence of pure finite-memory sure winning strategies for safety objectives in timed automaton games. Given a timed automaton game $\mathcal{T}$, we define two functions $P_{>0} : C \mapsto \{\text{TRUE}, \text{FALSE}\}$ and $P_{\geq 1} : C \mapsto \{\text{TRUE}, \text{FALSE}\}$. For a clock x, the values of $P_{>0}(x)$ and $P_{\geq 1}(x)$ indicate if the clock x was greater than 0 or greater than or equal to 1 respectively, during the last transition (excluding the originating state). Consider the enlarged game structure $\widetilde{\mathcal{T}}$ with the state space $\widetilde{S} = S \times \{\text{TRUE}, \text{FALSE}\} \times \{\text{TRUE}, \text{FALSE}\}^C \times \{\text{TRUE}, \text{FALSE}\}^C$ and an augmented transition relation $\widetilde{\delta}$. A state of $\widetilde{\mathcal{T}}$ is a tuple $\langle s, bl_1, P_{>0}, P_{\geq 1}\rangle$, where s is a state of $\mathcal{T}$, the component bl_1 is TRUE iff player 1 is to be blamed for the last transition, and $P_{>0}, P_{\geq 1}$ are as defined earlier. The clock equivalence relation can be lifted to states of $\widetilde{\mathcal{T}}$: $\langle s, bl_1, P_{>0}, P_{\geq 1}\rangle \cong_{\widetilde{A}} \langle s', bl_1', P_{>0}', P_{\geq 1}'\rangle$ iff $s \cong_{\mathcal{T}} s'$, $bl_1 = bl_1'$, $P_{>0} = P_{>0}'$ and $P_{\geq 1} = P_{\geq 1}'$.

Lemma 2. *Let $\mathcal{T}$ be a timed automaton game in which all clocks are bounded (i.e., for all clocks x we have $x \leq c_x$, for a constant c_x). Let $\widetilde{\mathcal{T}}$ be the enlarged game structure obtained from $\mathcal{T}$. Then player 1 has a receptive strategy from a state s iff $\langle s, \cdot\rangle \in \underline{\mathsf{Sure}}_1^{\widetilde{\mathcal{T}}}(\Phi)$, where*

$$\Phi = \Box\Diamond(bl_1 = \text{TRUE}) \rightarrow \left((\bigwedge_{x\in C} \Box\Diamond(x = 0)) \wedge \left(\begin{array}{c} (\bigvee_{x\in C} \Box\Diamond((P_{>0}(x) = \text{TRUE}) \wedge (bl_1 = \text{TRUE}))) \\ \vee \\ (\bigvee_{x\in C} \Box\Diamond((P_{\geq 1}(x) = \text{TRUE}) \wedge (bl_1 = \text{FALSE}))) \end{array} \right) \right).$$

Proof. We prove inclusion in both directions.

1. ($\Leftarrow$). For a state $\widetilde{s} \in \underline{\mathsf{Sure}}_1^{\widetilde{\mathcal{T}}}(\Phi)$, we show that player 1 has a receptive strategy from $\widetilde{s}$. Let π_1 be a pure sure winning strategy: since Φ is an ω-regular region objective such a strategy exists by Lemma 1. Consider a strategy π_1' for player 1 that is region-equivalent to π_1 such that whenever from a state $\widetilde{s}'$ the strategy π_1 proposes a move $\langle \Delta, a_1\rangle$ such that $\widetilde{s}' + \Delta$ satisfies $(x > 0)$, then π_1' proposes the move $\langle \Delta', a_1\rangle$ such that $\mathsf{Reg}(\widetilde{s}' + \Delta) = \mathsf{Reg}(\widetilde{s}' + \Delta')$ and $\widetilde{s}' + \Delta'$ satisfies $(x > 0) \wedge (\vee_{y\in C}\ y > 1/2)$. Such a move always exists; this is because, if there exists Δ such that $\widetilde{s} + \Delta \in R \subseteq (x > 0)$, then there exists Δ' such that $\widetilde{s} + \Delta' \in R \cap ((x > 0) \wedge (\vee_{y\in C}\ y > 1/2))$. Intuitively, player 1 jumps near the endpoint of R. By Lemma 1, π_1' is also sure-winning for Φ. The strategy π_1' ensures that in all resulting runs, if player 1 is not blameless, then all clocks are 0 infinitely often (since for all clocks $\Box\Diamond(x = 0)$), and that some clock has value more than 1/2 infinitely often. This implies time divergence. Hence player 1 has a receptive winning strategy from $\widetilde{s}$.

2. ($\Rightarrow$). For a state $\widetilde{s} \notin \underline{\mathsf{Sure}}_1^{\widetilde{\mathfrak{T}}}(\Phi)$, we show that player 1 does not have any receptive strategy starting from state $\widetilde{s}$. Let $\neg\Phi =$

$$(\Box\Diamond(bl_1 = \text{TRUE}))\wedge\left((\bigvee_{x\in C}\Diamond\Box(x>0))\vee\left(\bigwedge_{x\in C}\Diamond\Box\begin{pmatrix}(bl_1=\text{TRUE}\rightarrow(P_{>0}(x)=\text{FALSE}))\\ \wedge\\ (bl_1=\text{FALSE}\rightarrow(P_{\geq 1}(x)=\text{FALSE}))\end{pmatrix}\right)\right).$$

The objective of player 2 is $\neg\Phi$. Consider a state $\widetilde{s}'$ of $\widetilde{\mathfrak{T}}$. Suppose player 2 has some move from $\widetilde{s}'$ to a region R'', against a move of player 1 to a region R', then (by Lemma 14) it follows that from all states in $\mathsf{Reg}(\widetilde{s}')$, for each move of player 1 to R', player 2 has some move to R''. Since the objective $\neg\Phi$ is a region objective, only the region trace is relevant. Thus, for obtaining spoiling strategies of player 2, we may construct a finite-state region graph game, where the the states are the regions of the game, and edges specifies transitions across regions. Note that for a concrete move m_1 of player 1, if player 2 has a concrete move $m_2 = (\Delta_2, a_2)$ with a desired successor region R, then for any move $m_2' = (\Delta_2', a_2)$ with $\Delta_2' < \Delta_2$, the destination is R against the move m_1. The objective $\neg\Phi$ can be expressed as a disjunction of conjunction of Büchi and coBüchi objectives, and hence is a Rabin-objective. Then there exists a pure memoryless region-strategy for player 2 in the region-based game graph. In our original game, for all player 1 strategies π_1 there exists a player 2 strategy π_2 such that from every region the strategy π_2 specifies a destination region, and $\mathsf{Outcomes}(\widetilde{s}, \pi_1, \pi_2) \cap \neg\Phi \neq \emptyset$. Consider a player 1 strategy π_1 and the counter strategy π_2 satisfying the above conditions. Consider a run $r \in \mathsf{Outcomes}(\widetilde{s}, \pi_1, \pi_2) \cap \neg\Phi$. If for some clock, we have $\Diamond\Box(x > 0)$, then time converges (as all clocks are bounded in $\mathfrak{T}$), and thus π_1 is not a receptive strategy. Suppose we have $\wedge_{x\in C}\Box\Diamond(x = 0)$, then $\bigwedge_{x\in C}\Diamond\Box\,((bl_1 = \text{TRUE} \rightarrow (P_{>0}(x) = \text{FALSE})) \wedge (bl_1 = \text{FALSE} \rightarrow (P_{\geq 1}(x) = \text{FALSE})))$ holds. This means that after some point in the run, player 1 is only allowed to take moves which result in all the clock values being 0 throughout the move, this implies she can only take moves of time 0. Also, if player 2's move is chosen, then all the clock values are less than 1. Recall that in each step of the game, player 2 has a specific region he wants to go to. Consider a region equivalent strategy π_2' to the original player 2 spoiling strategy in which player 2 takes smaller and smaller times to get into a region R. If the new state is to have $\wedge_{x\in C}\,(P_{\geq 1}(x) = \text{FALSE})$, then player 2 gets there by choosing a time move smaller than $1/2^j$ in the j-th step. Since the destination regions are the same, and since smaller moves are always better, π_2' is also a spoiling strategy for player 2 against π_1. Moreover, time converges in the run where player 2 plays with π_2'. Thus, if a state $\widetilde{s} \notin \underline{\mathsf{Sure}}_1^{\widetilde{\mathfrak{T}}}(\Phi)$, then player 1 does not have a receptive strategy from $\widetilde{s}$.

$\Box$

Lemma 2 is generalized to all timed automaton games in the following lemma. Theorem 2 follows from Lemma 3 (see Appendix for the proof of the following lemma and the theorem).

Lemma 3. *Let $\mathfrak{T}$ be a timed automaton game, and $\widetilde{\mathfrak{T}}$ be the corresponding enlarged game. Then player 1 has a receptive strategy from a state s iff $\langle s, \cdot\rangle \in \underline{\mathsf{Sure}}_1^{\widetilde{\mathfrak{T}}}(\Phi^*)$, where $\Phi^* = \Box\Diamond(bl_1 = \text{TRUE}) \rightarrow \bigvee_{X\subseteq C}\ \phi_X$, and $\phi_X =$*

$$\left(\bigwedge_{x\in X}\Diamond\Box(x>c_x)\right)\wedge\left((\bigwedge_{x\in C\setminus X}\Box\Diamond(x=0))\wedge\begin{pmatrix}\left(\bigvee_{x\in C\setminus X}\Box\Diamond((P_{>0}(x)=\text{TRUE})\wedge(bl_1=\text{TRUE}))\right)\\ \vee\\ \left(\bigvee_{x\in C\setminus X}\Box\Diamond((P_{\geq 1}(x)=\text{TRUE})\wedge(bl_1=\text{FALSE}))\right)\end{pmatrix}\right).$$

Theorem 2. *Let $\mathfrak{T}$ be a timed automaton game and $\widetilde{\mathfrak{T}}$ be the corresponding enlarged game. Let Y be a union of regions of $\mathfrak{T}$. Then the following assertions hold.*

1. $\mathsf{Sure}_1^{\widetilde{\mathfrak{T}}}(\Box Y) = \underline{\mathsf{Sure}}_1^{\widetilde{\mathfrak{T}}}((\Box Y) \wedge \Phi^*)$, *where Φ^* is as defined in Lemma 3.*
2. *Player 1 has a pure, finite-memory, receptive, region strategy that is sure winning for the safety objective* $\mathsf{Safe}(Y)$ *at every state in* $\mathsf{Sure}_1^{\widetilde{\mathfrak{T}}}(\Box Y)$.

4 Reachability Objectives: Randomized Finite-memory Receptive Strategies Suffice

We have seen in Example 2 that pure sure winning strategies require infinite memory in general for reachability objectives. In this section, we shall show that uniform randomized almost-sure winning strategies with finite memory exist. This shows that we can trade-off infinite memory with uniform randomness.

Let S_R be the destination set of states that player 1 wants to reach. We only consider S_R such that S_R is a union of regions of $\mathfrak{T}$. For the timed automaton $\mathfrak{T}$, consider the enlarged game structure of $\mathfrak{T}$. We let $\widehat{S_R} = S_R \times \mathbb{R}_{[0,1]} \times \{\text{TRUE}, \text{FALSE}\}^2$. From the reachability objective (denoted $\mathsf{Reach}(S_R)$) we obtain the reachability parity objective with index function Ω_R as follows: $\Omega_R(\langle s, \mathfrak{z}, tick, bl_1\rangle) = 1$ if $tick \vee bl_1 = \text{TRUE}$ and $s \notin S_R$ (0 otherwise). We assume the states in S_R are absorbing. We let $\widehat{S_R} = S_R \times \mathbb{R}_{[0,1]} \times \{\text{TRUE}, \text{FALSE}\}^2$.

Lemma 4. *For a timed automaton game $\mathfrak{T}$, with the reachability objective S_R, consider the enlarged game structure $\widehat{\mathfrak{T}}$, and the corresponding reachability parity function Ω_R. Then we have that* $\underline{\mathsf{Sure}}_1(\mathsf{TimeDivBl}(\mathsf{Reach}(S_R))) = \underline{\mathsf{Sure}}_1(\mathsf{Parity}(\Omega_R))$.

We first present a μ-calculus characterization for the sure winning set (using only pure strategies) for player 1 for reachability objectives. The *controllable predecessor* operator for player 1, $\mathsf{CPre}_1 : 2^{\widehat{S}} \mapsto 2^{\widehat{S}}$, defined formally by $\widetilde{s} \in \mathsf{CPre}_1(Z)$ iff $\exists m_1 \in \widehat{\Gamma}_1(\widehat{s})\ \forall m_2 \in \widehat{\Gamma}_2(\widehat{s}) \,.\, \widehat{\delta}_{\mathsf{jd}}(\widehat{s}, m_1, m_2) \subseteq Z$. Informally, $\mathsf{CPre}_1(Z)$ consists of the set of states from which player 1 can ensure that the next state will be in Z, no matter what player 2 does. From Lemma 4 it follows that the sure winning set can be described as the μ-calculus formula: $\mu Y \nu X \left[(\Omega^{-1}(1) \cap \mathsf{CPre}_1(Y)) \cup (\Omega^{-1}(0) \cap \mathsf{CPre}_1(X))\right]$. The winning set can then be computed as a fixpoint iteration on regions of $\widehat{\mathfrak{T}}$. We can also obtain a pure winning strategy π_{pure} of player 1 as in [10]. Note that this strategy π_{pure} corresponds to an infinite-memory strategy of player 1 in the timed automaton game $\mathfrak{T}$, as she needs to maintain the value of the clock z in memory.

To compute randomized finite-memory almost-sure winning strategies, we will use the structure of the μ-calculus formula. Let $Y^* = \mu Y \nu X \left[(\Omega^{-1}(1) \cap \mathsf{CPre}_1(Y)) \cup (\Omega^{-1}(0) \cap \mathsf{CPre}_1(X))\right]$. The iterative fixpoint procedure computes $Y_0 = \emptyset \subseteq Y_1 \subseteq \cdots \subseteq Y_n = Y^*$, where $Y_{i+1} = \nu X \left[(\Omega^{-1}(1) \cap \mathsf{CPre}_1(Y_i)) \cup (\Omega^{-1}(0) \cap \mathsf{CPre}_1(X))\right]$. We can consider the states in $Y_i \setminus Y_{i-1}$ as being added in two steps, T_{2i-1} and $T_{2i}(= Y_i)$ as follows:

1. $T_{2i-1} = \Omega^{-1}(1) \cap \mathsf{CPre}_1(Y_{i-1})$. T_{2i-1} is clearly a subset of Y_i.
2. $T_{2i} = \nu X \left[T_{2i-1} \cup (\Omega^{-1}(0) \cap \mathsf{CPre}_1(X))\right]$. Note that $(T_{2i} \setminus T_{2i-1}) \cap \Omega^{-1}(1) = \emptyset$.

Thus, in the odd stages, we add states with index 1, and in even stages, we add states with index 0. The *rank* of a state $\widehat{s} \in Y^*$ is j if $\widehat{s} \in T_j \setminus \cup_{k=0}^{j-1} T_k$. For a state of even rank j, we have that player 1 can ensure that she has a move such that against all moves of player 2, the next state either (a) has index 0 and belongs to the same rank or less, or (b) the next state has index 1 and belongs to rank smaller than j. For a state of odd rank j, we have that player 1 can ensure that she has a move such that against all moves of player 2, the next state belongs to a lower rank (and has index either 1 or 0).

We now consider the rank sets for the reachability fixpoint in more detail. We have that S_R is a union of regions of $\mathcal{T}$. $T_0 = T_1 = \emptyset$, and T_2 consists of all the states in $\widehat{S_R}$ together with the states where $tick = bl_1 = \text{FALSE}$, and from where player 1 can ensure that the next state is either in $\widehat{S_R}$, or the next state continues to have $tick = bl_1 = \text{FALSE}$; formally $T_2 = \nu X(\Omega^{-1}(0) \cap \mathsf{CPre}_1(X))$. Henceforth, when we refer to a region R of $\mathcal{T}$, we shall mean the states $R \times \mathbb{R}_{[0,1]} \times \{\text{TRUE}, \text{FALSE}\}^2$ of $\widehat{\mathcal{T}}$.

Lemma 5. *Let $T_2 = \nu X(\Omega^{-1}(0) \cap \mathsf{CPre}_1(X))$. Then player 1 has a (randomized) memoryless strategy π_{rand} such that she can ensure reaching $\widehat{S}_R \subseteq \Omega^{-1}(0)$ with probability 1 against all receptive strategies of player 2 and all strategies of the scheduler from all states $\widehat{s}$ of a region R such that $R \cap T_2 \neq \emptyset$. Moreover, π_{rand} is independent of the values of the global clock, tick and bl_1.*

We break the proof of Lemma 5 into several parts. For a set T of states, we shall denote by $\mathsf{Reg}(T)$ the set of states that are region equivalent in $\mathcal{T}$ to some state in T. We let π_{pure} be the pure infinite-memory winning strategy of player 1 to reach $\widehat{S}_R$. First we prove the following result.

Lemma 6. *Let $T_2 = \nu X(\Omega^{-1}(0) \cap \mathsf{CPre}_1(X))$. Then, for every state in $\mathsf{Reg}(T_2)$, player 1 has a move to $\widehat{S}_R$.*

Proof. Suppose $T_2 \neq \widehat{S_R}$ (the other case is trivial). Then player 1 must have a move from every state in $T_2 \setminus \widehat{S_R}$ to $\widehat{S_R}$ in $\widehat{\mathcal{T}}$, for otherwise, for any state in $T_2 \setminus \widehat{S_R}$, player 2 (with cooperation from the scheduler) can allow player 1 to pick any move, which will result in an index of 1 in the next state, contradicting the fact that player 1 had a strategy to stay in T_2 forever (note all the states in T_2 have index 0). Moreover, since $\widehat{S}_R$ is a union of regions of $\mathcal{T}$, we have that the states in T_2 from which player 1 has a move to $\widehat{S}_R$, consist of a union of sets of the form $T_2 \cap R$ for R a region of $\mathcal{T}$. This implies that player 1 has a move to $\widehat{S}_R$ from all states in $\mathsf{Reg}(T_2)$ (Lemma 13). ◻

If at any time player 1's move is chosen, then player 1 comes to $\widehat{S}_R$, and from there plays a receptive strategy. We show that player 1 has a randomized memoryless strategy such that the probability of player 1's move being never chosen against a receptive strategy of player 2 is 0. This strategy will be pure on target left-closed regions, and a uniformly distributed strategy on target left-open regions. We now describe the randomized strategy.

Consider a state $\widehat{s}$ in some region $R' \subseteq \mathsf{Reg}(T_2 \setminus \widehat{S}_R)$ of $\mathcal{T}$. Now consider the set of times at which moves can be taken so that the state changes from $\widehat{s}$ to $\widehat{S}_R$. This set consists of a finite union of sets I_k of the form $(\alpha_l, \alpha_r), [\alpha_l, \alpha_r), (\alpha_l, \alpha_r]$, or $[\alpha_l, \alpha_r]$ where α_l, α_r are of the form d or $d - x$ for d some integer constant, and x some clock in C (this clock x is the same for all the states in R'). Furthermore, these intervals have the property that $\{\widehat{s} + \Delta \mid \Delta \in I_k\} \subseteq R_k$ for some region R_k, with $R_l \cap R_j = \emptyset$ for $j \neq l$. From a state $\widehat{s}$, consider the "earliest" interval contained in this union: the interval I such that the left endpoint is the infimum of the times at which player 1 can move to $\widehat{S}_R$. We have that $\{\widehat{s} + \Delta \mid \Delta \in I\} \subseteq R_1$. Consider any state $\widehat{s}' \in R'$. Then from $\widehat{s}'$, the earliest interval in the times required to get to $\widehat{S}_R$ is also of the form I. Note that in allowing time to pass to get to R_1, we may possibly go outside T_2 (recall that T_2 is not a union of regions of $\mathcal{T}$).

If this earliest interval I is left closed, then player 1 has a "shortest" move to $\widehat{S}_R$. Then this is the best move for player 1, and she will always propose this move. We call these regions *target left-closed*. If the target interval is left open, we call the region *target left-open*. Let the left and the right endpoints of target intervals be α_l, α_r respectively. Then let player 1 play a probabilistic strategy with time distributed uniformly at random over $(\alpha_l, (\alpha_l + \alpha_r)/2]$ on these target left-open regions. Let us denote this player-1 strategy by π_{rand}.

Lemma 7. *Let $T_2 = \nu X(\Omega^{-1}(0) \cap \mathsf{CPre}_1(X))$. Then, for every state in $\mathsf{Reg}(T_2)$ the strategy π_{rand} as described above ensures that player 1 stays inside $\mathsf{Reg}(T_2)$ surely.*

Proof. Consider a state $\widehat{s}$ in $T_2 \setminus \widehat{S}_R$. Since $\{\widehat{s} + t \mid t \in I\}$ is a subset of a single region of $\mathcal{T}$, no new discrete actions become enabled due to the randomized strategy of player 1. If player 2 can foil player 1 by taking a move to a region R'' for the player 1 randomized strategy, she can do so against any pure (infinite-memory) strategy of player 1. No matter what player 2 proposes at each step, player 1's strategy is such that the next state (against any player 2's moves) lies in a region R'' (of $\mathcal{T}$) such that $R'' \cap T_2 \neq \emptyset$. Because of this, player 1 can always play the above mentioned strategy at each step of the game, and ensure that she stays inside $\mathsf{Reg}(T_2)$ (until the destination $\widehat{S}_R$ is reached). ☐

Lemma 8. *Consider the the player 1 strategy π_{rand} and any receptive strategy π_2 of player 2. Let $r \in \mathsf{Outcomes}(s, \pi_{\mathsf{rand}}, \pi_2)$ be a run with $s \in \mathsf{Reg}(T_2)$. If there exists $m \geq 0$ such that $\pi_{\mathsf{rand}}(r[0..j])$ is left-closed for all $j \geq m$, then we have that r visits $\widehat{S}_R$.*

Proof. Consider a run r for the player 1 strategy π_{rand} against any strategy π_2 of player 2. Note that we must have $\mathsf{Reg}(r[k]) \subseteq \mathsf{Reg}(T_2)$ for every k by Lemma 7. Let $r[m] = \widehat{s}' = \langle s', \mathfrak{z}, tick, bl_1 \rangle \in R'$. Consider the pure winning strategy π_{pure} from a state $\widehat{s}'' = \langle s', \mathfrak{z}', tick', bl_1' \rangle \in R' \cap T_2$ (such a state must exist). The state $\widehat{s}''$ differs from $\widehat{s}'$ only in the values of the clock z, and the boolean variables $tick$ and bl_1. The new values do not affect the moves available to either player. Consider $\widehat{s}''$ as the starting state. The strategy π_{pure} cannot propose shorter moves to $\widehat{S}_R$, since π_{rand} proposes the earliest move to $\widehat{S}_R$. Hence, if a receptive player 2 strategy π_2 can prevent π_{rand} from reaching $\widehat{S}_R$ from $\widehat{s}'$, then it can also prevent π_{pure} from reaching $\widehat{S}_R$ from $\widehat{s}''$, a contradiction. ☐

Lemma 9. *Consider the the player 1 strategy π_{rand} and any receptive strategy π_2 of player 2. Let $r \in \mathsf{Outcomes}(s, \pi_{\mathsf{rand}}, \pi_2)$ be a run with $s \in \mathsf{Reg}(T_2)$. There exists $m \geq 0$ such that for all $j \geq m$, if $\pi_{\mathsf{rand}}(r[0..j])$ is left-open, then the left endpoint is $\alpha_l = 0$.*

Proof. Let α_l correspond to the left endpoints for one of the infinitely occurring target left-open regions R.

1. We show that we cannot have α_l to be of the form d for some integer $d > 0$.
   We prove by contradiction. Suppose α_l is of the form $d > 0$. Then player 2 could always propose a time blocking move of duration d, this would mean that if the scheduler picks the move of player 2 (as both have the same delay), the next state would have $tick = \text{TRUE}$, no matter what the starting value of the clock z in R, contradicting the fact that $R \cap T_2 \neq \emptyset$ $(T_2 = \nu X(\Omega^{-1}(0) \cap \mathsf{CPre}_1(X)))$. We have a contradiction, as player 1 had a pure winning strategy π_{pure} from every state in T_2. Take any $\widehat{s} \in R \cap T_2$. Then π_{pure} must have proposed some move to $\widehat{S}_R$, such that all the intermediate states (before the move time) had $tick = \text{FALSE}$. The strategy π_{rand} picks the earliest left most endpoint to get to $\widehat{S}_R$. This means that π_{pure} must also

propose a time which is greater than or equal to the move proposed by π_{rand}. Hence α_l cannot be d for $d > 0$ (otherwise the player 2 counter strategy to π_{pure} can take the game out of T_2 by making $tick = \text{TRUE}$).

2. We show that we cannot have α_l to be of the form $d - x$ for some integer $d > 0$.
We prove by contradiction. Suppose $\alpha_l = d - x$ for some some clock x for the target constraint. Let player 2 counter with any strategy.
Suppose clock x is not reset infinitely often in the run r. Then the fact that the clock x has not progressed beyond d at any time in the run without being reset implies time is convergent, contradicting the fact that player 2 is playing with a receptive strategy (note that only player 2's moves are being chosen). Thus, this situation cannot arise.
Suppose x is reset infinitely often. Then between a reset of x, and the time at which player 1 can jump to $\widehat{S}_R$, we must have a time distance of more than d. Suppose R' is one of the infinitely occurring regions in the run with the value of x being 0 in it. So player 2 has a strategy against our player 1 strategy such that one of the resulting runs contains a region subsequence $R' \rightsquigarrow R$. If this is so, then she would have a strategy which could do the same from every state in $R' \cap T_2$ against the pure winning strategy of player 1 (since the randomized strategy π_{rand} does not enable player 2 to go to more regions than against π_{pure}, as π_{rand} proposes moves to the earliest region in $\widehat{S}_R$). But, if so, we have that *tick* will be true no matter what the starting value of z in $R' \cap T_2$, before player 1 can take a jump to $\widehat{S}_R$ from $R \cap T_2$, taking the game outside of T_2. Since player 1 can stay inside T_2 at each step with the infinite memory strategy π_{pure}, this cannot be so, that is, we cannot observe the region subsequence $R' \rightsquigarrow R$ for the randomized strategy of player 1. Thus the case of $\alpha_l = d - x$ cannot arise infinitely often.

The only remaining option is $\alpha_l = 0$, and we must have that the only randomized moves player 1 proposes after a while are of the form $(0, \alpha_r/2]$. ☐

Lemma 10. *Consider runs r with $r[0] \in \mathsf{Reg}(T_2)$ for the player 1 strategy π_{rand} against any receptive strategy π_2 of player 2 and a scheduler strategy π_{sched}. Let $\mathcal{E}$ be the set of runs such that for all $m \geq 0$ there exists $j \geq m$ such that $\pi_{\mathsf{rand}}(r[0..j])$ is left-open, with the the left endpoint being $\alpha_l = 0$. Then, we have $\Pr_{r[0]}^{\pi_{\mathsf{rand}},\pi_2,\pi_{\mathsf{sched}}}(\mathsf{Reach}(S_R)\,|\mathcal{E}) = 1$.*

Proof. Let one of the infinitely often occurring player 1 left-open moves be to the region R. Player 1 proposes a uniformly distributed move over $(0, \alpha_r/2]$ to R. Let β_i be the duration of player 2's move for the ith visit to R Suppose $\alpha_r = d$. Then the probability of player 1's move being never chosen is less than $\prod_{i=1}^{\infty}(1 - \frac{2\beta_i}{d})$, which is 0 if $\sum_{i=1}^{\infty}\beta_i = \infty$ by Lemma 15. A similar analysis holds if player 2 proposes randomized moves with a time distribution $\mathcal{D}^{(\beta_i,\cdot]}, \mathcal{D}^{[\beta_i,\cdot]}, \mathcal{D}^{(\beta_i,\cdot)}$ or $\mathcal{D}^{[\beta_i,\cdot)}$. Suppose $\alpha_r = d - x$. Again, the probability of player 1's move being never chosen is less than $\prod_{i=1}^{\infty}(1 - \frac{2\beta_i}{(d-\kappa_i(x))})$, and since $\frac{\beta_i}{(d-\kappa_i(x))} > \frac{\beta_i}{d}$, this also is 0 if $\sum_{i=1}^{\infty}\beta_i = \infty$ by Lemma 15 . Finally, we note that if player 2 does not block time from T_2, then for at least one region, she must propose a β_i sequence such that $\sum_{i=1}^{\infty}\beta_i = \infty$, and we will have that for this region, player 1's move will be chosen eventually with probability 1. ☐

Proof (Lemma 5). Lemmas 6, 7, 8, 9 and 10 together imply that using the randomized memoryless strategy π_{rand}, player 1 can ensure going from *any* region R of $\mathcal{T}$ such that $R \cap T_2 \neq \emptyset$ to $\widehat{S}_R$ with probability 1, without maintaining the infinitely precise value of the global clock. ☐

The following lemma states that if for some state $s \in \mathcal{T}$, we have $(s, \mathfrak{z}, tick, bl_1) \in T_{2i+1}$, for some i, then for some $\mathfrak{z}', tick', bl_1'$ we have $(s, \mathfrak{z}', tick', bl_1') \in T_{2i}$. Then in Lemma 12 we present the inductive case of Lemma 5. The proof of Lemma 12 is similar to the base case i.e., Lemma 5.

Lemma 11. *Let R be a region of $\mathcal{T}$ such that $R \cap T_{2i+1} \neq \emptyset$. Then $R \cap T_{2i} \neq \emptyset$.*

Lemma 12. *Let R be a region of $\mathcal{T}$ such that $R \cap T_{2i} \neq \emptyset$, and $R \cap T_j = \emptyset$ for all $2 \leq j < 2i$. Then player 1 has a (randomized) memoryless strategy π_{rand} to go from R to some R' such that $R' \cap T_j \neq \emptyset$ for some $j < 2i$ with probability 1 against all receptive strategies of player 2 and all strategies of the scheduler. Moreover, π_{rand} is independent of the values of the global clock, tick and bl_1.*

Once player 1 reaches the target set, she can switch over to the finite-memory receptive strategy of Lemma 3. Thus, using Lemmas 3, 5, 11, and 12 we have the following theorem.

Theorem 3. *Let $\mathcal{T}$ be a timed automaton game, and let S_R be a union of regions of $\mathcal{T}$. Player 1 has a randomized, finite-memory, receptive, region strategy π_1 such that for all states $s \in \mathsf{Sure}_1(\mathsf{Reach}(S_R))$, and for all scheduler strategies π_{sched}, the following assertions hold: (a) for all receptive strategies π_2 of player 2 we have $\Pr_s^{\pi_1,\pi_2,\pi_{\mathsf{sched}}}(\mathsf{Reach}(S_R)) = 1$; and (b) for all strategies π_2 of player 2 we have $\Pr_s^{\pi_1,\pi_2,\pi_{\mathsf{sched}}}(\mathsf{TimeDivBl}_1(\mathsf{Reach}(S_R))) = 1$.*

5 Parity Objectives: Randomized Finite-memory Receptive Strategies Suffice

In this section we show that randomized finite-memory almost-sure strategies exist for parity objectives. Let $\Omega : S \mapsto \{0, \ldots, k\}$ be the parity index function. We consider the case when $k = 2d$ for some d, and the case when $k = 2d - 1$, for some d can be proved using similar arguments. If $k = 2d - 1$, then we will will look at the dual odd parity objective: $\mathsf{Parity}_{\mathsf{odd}}(\Omega') = \{r \mid \max(\mathsf{InfOften}(r)) \text{ is odd }\}$, with $\Omega' = \Omega + 1 : S \mapsto \{1, \ldots, 2d\}$. If we get an odd parity objective with $\Omega' : S \mapsto \{1, \ldots, 2d-1\}$, then we can map it back to an even parity objective with $\Omega = \Omega' - 1$.

Given a timed game structure $\mathcal{T}$, a set $X \subsetneq S$, and a parity function $\Omega : S \mapsto \{0, \ldots, 2d\}$, with $d > 0$, let $\langle \mathcal{T}', \Omega' \rangle = \mathsf{ModifyEven}(\mathcal{T}, \Omega, X)$ be defined as follows: (a) the state space S' of $\mathcal{T}'$ is $\{s^\perp\} \cup S \setminus X$, where $s^\perp \notin S$; (b) $\Omega'(s^\perp) = 2d - 2$, and $\Omega' = \Omega$ otherwise; (c) $\Gamma_i'(s) = \Gamma_i(s)$ for $s \in S \setminus X$, and $\Gamma_i'(s^\perp) = \Gamma_i(s^\perp) = \mathbb{R}_{\geq 0} \times \perp$; and (d) $\delta'(s, m) = \delta(s, m)$ if $\delta(s, m) \in S \setminus X$, and $\delta'(s, m) = s^\perp$ otherwise. We will use the function $\mathsf{ModifyEven}$ to play timed games on a subset of the original structure. The extra state, and the modified transition function are to ensure well-formedness of the reduced structure. We will now obtain receptive strategies for player 1 for the objective $\mathsf{Parity}(\Omega)$ using winning strategies for reachability and safety objectives. We consider the following procedure.

1. $i := 0$, and $\mathcal{T}_i = \mathcal{T}$.
2. Compute $X_i = \mathsf{Sure}_1^{\mathcal{T}_i}(\Diamond(\Omega^{-1}(2d)))$.
3. Let $\langle \mathcal{T}_i', \Omega' \rangle = \mathsf{ModifyEven}(\mathcal{T}_i, \Omega, X_i)$; and let $Y_i = \mathsf{Sure}_1^{\mathcal{T}_i'}(\mathsf{Parity}(\Omega'))$. Let $L_i = S_i \setminus Y_i$, where S_i is the set of states of $\mathcal{T}_i$.
4. Compute $Z_i = \mathsf{Sure}_1^{\mathcal{T}_i}(\Box(S_i \setminus L_i))$.
5. Let $(\mathcal{T}_{i+1}, \Omega) = \mathsf{ModifyEven}(\mathcal{T}, \Omega, S \setminus Z_i)$ and $i := i + 1$.
6. Go to step 2, unless $Z_{i-1} = S_i$.

Consider the sets $S \setminus Z_i$ that are removed in each iteration. For every L_i, the probability of player 1 winning in $\mathcal{T}$ is 0. This is because, from L_i, player 1 cannot visit the index $2d$ with positive probability, thus we can restrict our attention to $\mathcal{T}'$, and in this structure, L_i is not winning for player 1 almost surely. This in turn implies that $S \setminus \mathsf{Sure}_1^{\mathcal{T}_i}(\Box(S \setminus L_i))$ is a losing set for player 1 almost surely in the structure $\mathcal{T}$. Thus, at the end of the iterations, we have $\mathsf{Sure}_1^{\mathcal{T}}(\mathsf{Parity}(\Omega)) \subseteq Z_i$. Hence, we have $(S \setminus Z_i) \cap \mathsf{Sure}_1^{\mathcal{T}}(\mathsf{Parity}(\Omega)) = \emptyset$. We now exhibit randomized, finite-memory, receptive, region almost-sure winning strategies to show that the set Z_i is almost-sure winning.

The set Z_i on termination has two subsets: (a) $X_i = \mathsf{Sure}_1^{\mathcal{T}_i}(\Diamond(\Omega^{-1}(2d)))$; and (b) $Y_i = S_i \setminus X_i$ such that player 1 wins in the structure $\mathcal{T}_i'$ for the parity objective $\mathsf{Parity}(\Omega)$. Let π^Y be a randomized, finite-memory, receptive, region almost-sure winning strategy for player 1 in $\mathcal{T}_i'$; since the range of Ω $\mathcal{T}_i'$ is $\{0, 1, \dots, 2d-1\}$, by inductive hypothesis such a strategy exists. Consider any receptive strategy of player 2. If the game is in Y_i, then player 1 use the strategy π^Y, using the the run suffix r^Y, where r^Y is the largest suffix of the run such that all the states of r^Y belong to Y_i . Moreover, player 1 is never to blame if time converges (since π^Y is a receptive strategy). Suppose the game hits X_i. Then, player 1 uses a randomized, finite-memory, receptive, region almost-sure winning strategy π^X to visit the index $2d$, and as soon as $2d$ is visited, she switches over to a pure, finite-memory, receptive, region safety strategy for the objective $\Box(Z_i)$ to allow a fixed amount of time $\Delta > 0$ to pass. This can be done similar to the receptive strategies of Theorem 2 with an imprecise clock (in the imprecise clock the time elapse between any two ticks is at least Δ). Once time more than Δ has passed, player 1 switches over to π^X or π^Y, depending on whether the current state is in X_i or Y_i, respectively, and repeats the process. This is a receptive strategy which ensures that the maximal priority that is visited infinitely often is even almost-surely. The strategy also requires only a finite amount of memory.

Theorem 4. *Let $\mathcal{T}$ be a timed automaton game, and let Ω be a region parity index function. Suppose that player 1 has access to imprecise clock events such that between any two events, some time more than Δ passes for a fixed real $\Delta > 0$. Then, player 1 has a randomized, finite-memory, receptive, region strategy π_1 such that for all states $s \in \mathsf{Sure}_1(\mathsf{Parity}(\Omega))$, and for all scheduler strategies π_{sched}, the following assertions hold: (a) for all receptive strategies π_2 of player 2 we have $\Pr_s^{\pi_1,\pi_2,\pi_{\mathsf{sched}}}(\mathsf{Parity}(\Omega)) = 1$; and (b) for all strategies π_2 of player 2 we have $\Pr_s^{\pi_1,\pi_2,\pi_{\mathsf{sched}}}(\mathsf{TimeDivBl}_1(\mathsf{Parity}(\Omega))) = 1$.*

Winning sets with Randomization. In pure timed games, player 1 wins for the objective Φ iff she has a strategy π_1 that works against all possible strategies of player-2 for the objective $\mathsf{TimeDivBl}(\Phi)$ in the non-receptive game. Suppose player-1 wins from state s in a pure timed game. This means that player-1 has a strategy π_1 that wins against all possible pure strategies of player-2. A randomized strategy may be viewed as a random choice over pure strategies. Thus, π_1 will also win surely against all possible randomized strategies of player-2. Hence, if player-1 can win from state s in the pure case, she can win from s surely in the randomized game.

We now present an informal argument to show that if player-1 cannot win from s in the pure game, then she cannot do so either with randomized strategies. Let $\mathcal{T}$ be a timed automaton game with an ω-regular region objective Φ. Suppose $\widehat{s}$ is a not a sure winning state for player 1, i.e., $\widehat{s} \in S \setminus \mathsf{Sure}_1^{\mathcal{T}}(\Phi)$. We show that for all randomized strategies π_1, for all $\varepsilon > 0$, there exists a pure region strategy π_2 for player 2 and a strategy π_{sched} for the scheduler such that $\Pr_{\widehat{s}}^{\pi_1,\pi_2,\pi_{\mathsf{sched}}}(\mathsf{TimeDivBl}_1(\Phi)) \leq \varepsilon$. Using Lemma 14, it is possible to construct a *region-based turn-based game graph* $\widehat{\mathcal{T}}$, where player 1 first selects a destination region, then player 2 picks a counter-move to specify another destination region. Since Φ is an ω-regular region strategy, in the game graph $\widehat{\mathcal{T}}$, if player 1 cannot win surely, then there is a pure region spoiling strategy π_2^* for player 2 that works against all player 1 strategies in $\widehat{\mathcal{T}}$ (for some strategy of the scheduler). Fix some $\varepsilon > 0$, and a sequence $(\varepsilon_i)_{i\geq 0}$ such that $\varepsilon_i > 0$, for all $i \geq 0$, and $\sum_{i\geq 0} \varepsilon_i \leq \varepsilon$. Consider a randomized strategy π_1 of player 1 in $\widehat{\mathcal{T}}$. We will construct a counter strategy π_2 for player 2 to π_1. If player 1 proposes a pure move, then the counter move of player 2 can be derived from the strategy π_2^* in $\widehat{\mathcal{T}}$. Suppose player 1 proposes a randomized move of the form $\langle \mathcal{D}^{(\alpha,\beta)}, a_1^j \rangle$ (the case where the move is of the form $\langle \mathcal{D}^{[\alpha,\beta)}, a_1^j \rangle, \langle \mathcal{D}^{[\alpha,\beta]}, a_1^j \rangle, \langle \mathcal{D}^{(\alpha,\beta]}, a_1^j \rangle$ is similar) at a state $\widehat{s}_j$ in the j-th step. The interval (α, β) can be decomposed into $2k+1$ intervals $(\beta_0, \beta_1), \{\beta_1\}, (\beta_1, \beta_2), \{\beta_2\}, \ldots, \{\beta_k\}, (\beta_k, \beta_{k+1})$, with $\beta_0 = \alpha$ and $\beta_{k+1} = \beta$, such that for all $0 \leq i \leq k$, the set $H_i = \{\widehat{s}_j + \Delta \mid \beta_i < \Delta < \beta_{i+1}\}$ is a subset of a region $\widehat{R}_i$, and $\widehat{R}_i \neq \widehat{R}_j$, for $i \neq j$, and similar result hold for the singletons. Consider the counter strategy π_2^* of player 2 in the region game graph for the player 1 moves to $\widehat{R}_1, \ldots, \widehat{R}_{2k+1}$. The counter strategy π_2 at the j-th step is as follows.

- Suppose the strategy π_2^* allows player 1 moves to all $\widehat{R}_1, \ldots, \widehat{R}_{2k+1}$. Then the strategy π_2 picks a move in a region $\widehat{R}'$ such that $\widehat{R}'$ is a counter move of player 2 against $\widehat{R}_{2k+1}$ in π_2^*.
- Suppose the strategy π_2^* allows player 1 moves to $\widehat{R}_1, \ldots \widehat{R}_m$, and not to $\widehat{R}_{m+1}$. Let the counter strategy π_2^* pick some region $\widehat{R}'$ (together with some action a_2) against the player 1 move to $\widehat{R}_{m+1}$. The strategy π_2 is specified considering the following cases.
  1. Suppose $\widehat{R}'$ is a closed region, then from $\widehat{s}_j$ there is an unique time move Δ_j such that $\widehat{s}_j + \Delta_j \in \widehat{R}'$, and the strategy π_2 of player 2 picks $\langle \Delta_j, a_2 \rangle$ such that $\widehat{s} + \Delta_j \in \widehat{R}'$.
  2. Suppose $\widehat{R}'$ is an open region. If $\widehat{R}'$ lies "before" $\widehat{R}_1$, then π_2 picks any move to $\widehat{R}'$. Otherwise, let $\widehat{R}' = \widehat{R}_{2l+1}$ for some l with $2l+1 \leq m+1$. Then, player 2 has some move $\langle \Delta_j, a_2 \rangle$, such that $\langle \Delta_j, a_2 \rangle$ will "beat" player 1 moves to $\widehat{R}_{m+1}, \cdots, \widehat{R}_{2k+1}$ with probability greater than $1 - \varepsilon_j$ and $\widehat{s}_j + \Delta_j \in \widehat{R}'$, and π_2 picks the move (Δ_j, a_2).

The player 2 strategy π_2 ensures that some desired region sequence (complementary to player 1's objective) is followed with probability at least $1 - \varepsilon$ for some strategy of the scheduler. This gives us the following result.

Theorem 5. *Consider a timed automaton game $\mathcal{T}$ with an ω-regular objective Φ. For all $s \in S \setminus \mathsf{Sure}_1^{\mathcal{T}}(\Phi)$, for every $\varepsilon > 0$, for every randomized strategy $\pi_1 \in \Pi_1$ of player 1, there is a player 2 pure strategy $\pi_2 \in \Pi_2^{\mathsf{pure}}$ and a scheduler strategy $\pi_{\mathsf{sched}} \in \Pi_{\mathsf{sched}}$ such that $\Pr_s^{\pi_1,\pi_2,\pi_{\mathsf{sched}}}(\mathsf{TimeDivBl}_1(\Phi)) \leq \varepsilon$.*

It follows from Theorem 5 that for timed automaton games the set of sure and almost-sure winning states coincide for ω-regular objectives. It also shows that though randomization can get rid of infinite memory with respect to almost-sure winning, randomization does not help to win in more states.

References

1. B. Adler, L. de Alfaro, and M. Faella. Average reward timed games. In *FORMATS 05*, LNCS 3829, pages 65–80. Springer, 2005.
2. R. Alur and D.L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994.
3. R. Alur and T.A. Henzinger. Modularity for timed and hybrid systems. In *CONCUR 97*, LNCS 1243, pages 74–88. Springer, 1997.
4. E. Asarin and O. Maler. As soon as possible: Time optimal control for timed automata. In *HSCC 99*, LNCS 1569, pages 19–30. Springer, 1999.
5. P. Bouyer, F. Cassez, E. Fleury, and K.G. Larsen. Optimal strategies in priced timed game automata. In *FSTTCS 04*, LNCS 3328, pages 148–160. Springer, 2004.

6. P. Bouyer, D. D'Souza, P. Madhusudan, and A. Petit. Timed control with partial observability. In *CAV 03*, LNCS 2725, pages 180–192. Springer, 2003.
7. F. Cassez, A. David, E. Fleury, K.G. Larsen, and D. Lime. Efficient on-the-fly algorithms for the analysis of timed games. In *CONCUR 05*, LNCS 3653, pages 66–80. Springer, 2005.
8. L. de Alfaro, M. Faella, T.A. Henzinger, R. Majumdar, and M. Stoelinga. The element of surprise in timed games. In *CONCUR 03*, LNCS 2761, pages 144–158. Springer, 2003.
9. L. de Alfaro, T.A. Henzinger, and R. Majumdar. From verification to control: Dynamic programs for omega-regular objectives. In *LICS 01*, pages 279–290. IEEE Computer Society Press, 2001.
10. L. de Alfaro, T.A. Henzinger, and R. Majumdar. Symbolic algorithms for infinite-state games. In *CONCUR 01*, LNCS 2154, pages 536–550. Springer, 2001.
11. D. D'Souza and P. Madhusudan. Timed control synthesis for external specifications. In *STACS 02*, LNCS 2285, pages 571–582. Springer, 2002.
12. M. Faella, S. La Torre, and A. Murano. Automata-theoretic decision of timed games. In *VMCAI 02*, LNCS 2294, pages 94–108. Springer, 2002.
13. M. Faella, S. La Torre, and A. Murano. Dense real-time games. In *LICS 02*, pages 167–176. IEEE Computer Society, 2002.
14. T.A. Henzinger and P.W. Kopke. Discrete-time control for rectangular hybrid automata. *Theoretical Computer Science*, 221:369–392, 1999.
15. T.A. Henzinger and V.S. Prabhu. Timed alternating-time temporal logic. In *FORMATS 06*, LNCS 4202, pages 1–17. Springer, 2006.
16. O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems (an extended abstract). In *STACS 95*, pages 229–242, 1995.
17. A. Pnueli, E. Asarin, O. Maler, and J. Sifakis. Controller synthesis for timed automata. In *Proc. System Structure and Control*. Elsevier, 1998.
18. C.C. Pugh. *Real Analysis*. Springer, 2002.
19. R. Segala, R. Gawlick, J.F. Søgaard-Andersen, and N.A. Lynch. Liveness in timed and untimed systems. *Inf. Comput.*, 141(2):119–171, 1998.
20. W. Thomas. Languages, automata, and logic. In *Handbook of Formal Languages*, volume 3, Beyond Words, chapter 7, pages 389–455. Springer, 1997.
21. H. Wong-Toi and G. Hoffmann. The control of dense real-time discrete event systems. In *Proc. of 30th Conf. Decision and Control*, pages 1527–1528, 1991.

6 Appendix

We first present the proof ideas for Lemma 1. We start with the statement of a classical result of [2] that the region equivalence relation induces a time abstract bisimulation on the regions.

Lemma 13 ([2]). *Let Y, Y' be regions in the timed game structure $\mathcal{T}$. Suppose player i has a move from $s_1 \in Y$ to $s'_1 \in Y'$, for $i \in \{1, 2\}$. Then, for any $s_2 \in Y$, player i has a move from s_2 to some $s'_2 \in Y'$.*

Let Y, Y'_1, Y'_2 be regions. We prove in Lemma 14 that one of the following two conditions hold: (a) for all states in Y there is a move for player 1 with destination in Y'_1, such that for all player 2 moves with destination in Y'_2, the next state is in Y'_1; or (b) for all states in Y for all moves for player 1 with destination in Y'_1 there is a move of player 2 to ensure that the next state is in Y'_2.

Lemma 14. *Let Y, Y'_1, Y'_2 be regions in the timed game structure $\mathcal{T}$. Suppose player i has a pure-time move from $s_1 \in Y$ to $s'_1 \in Y'_i$, for $i \in \{1, 2\}$. Then, one of the following cases must hold:*

1. *From all states $s \in Y$, for every player-1 pure-time move m^s_1 with $\delta(s, m^s_1) \in Y'_1$, for all pure-time moves m^s_2 of player 2 with $\delta(s, m^s_2) \in Y'_2$, we have $\mathsf{blame}_1(s, m^s_1, m^s_2, \delta(s, m^s_1)) = \text{TRUE}$ and $\mathsf{blame}_2(s, m^s_1, m^s_2, \delta(s, m^s_2)) = \text{FALSE}$.*

2. *From all states $s \in Y$, for every player-1 pure-time move m_1^s with $\delta(s, m_1^s) \in Y_1'$, for all pure-time moves m_2^s of player 2 with $\delta(s, m_2^s) \in Y_2'$, we have* $\mathsf{blame}_2(s, m_1^s, m_2^s, \delta(s, m_2^s)) = \text{TRUE}$.

Proof. We first present the proof for the case when $Y_1' \neq Y_2'$. The proof follows from the fact that each region has a unique first *time-successor* region. A region R' is a first time-successor of $R \neq R'$ if for all states $s \in R$, there exists $\Delta > 0$ such that $s + \Delta \in R'$ and for all $\Delta' < \Delta$, we have $s + \Delta' \in R \cup R'$. The time-successor of $\langle l, h, \mathcal{P}(C)\rangle$ is $\langle l, h', \mathcal{P}'(C)\rangle$ when

- $h = h'$, $\mathcal{P}(C) = \langle C_{-1}, C_0 \neq \emptyset, C_1, \ldots, C_n\rangle$, and $\mathcal{P}'(C) = \langle C_{-1}, C_0' = \emptyset, C_1', \ldots, C_{n+1}'\rangle$ where $C_i' = C_{i-1}$, and $h(x) < c_x$ for every $x \in C_0$.
- $h = h'$, $\mathcal{P}(C) = \langle C_{-1}, C_0 \neq \emptyset, C_1, \ldots, C_n\rangle$, and $\mathcal{P}'(C) = \langle C_{-1}' = C_{-1} \cup C_0, C_0' = \emptyset, C_1, \ldots, C_n\rangle$, and $h(x) \geq c_x$ for every $x \in C_0$.
- $h = h'$, $\mathcal{P}(C) = \langle C_{-1}, C_0 \neq \emptyset, C_1, \ldots, C_n\rangle$, and $\mathcal{P}'(C) = \langle C_{-1}', C_0' = \emptyset, C_1', \ldots, C_{n+1}'\rangle$ where $C_i' = C_{i-1}$ for $i \geq 2$, $h(x) < c_x$ for every $x \in C_1' \subseteq C_0$, and $h(x) \geq c_x$ for every $x \in C_0 \setminus C_1'$, and $C_{-1}' = C_{-1} \cup C_0 \setminus C_1'$.
- $\mathcal{P}(C) = \langle C_{-1}, C_0 = \emptyset, C_1, \ldots, C_n\rangle$, $\mathcal{P}'(C) = \langle C_{-1}, C_0' = C_n, C_1, \ldots, C_{n-1}\rangle$, and $h'(x) = h(x) + 1 \leq c_x$ for every $x \in C_n$, and $h'(x) = h(x)$ otherwise.
- $\mathcal{P}(C) = \langle C_{-1}, C_0 = \emptyset, C_1, \ldots, C_n\rangle$, $\mathcal{P}'(C) = \langle C_{-1}' = C_{-1} \cup C_n, C_0, C_1, \ldots, C_{n-1}\rangle$, and $h'(x) = h(x) = c_x$ for every $x \in C_n$, and $h'(x) = h(x)$ otherwise.
- $\mathcal{P}(C) = \langle C_{-1}, C_0 = \emptyset, C_1, \ldots, C_n\rangle$, $\mathcal{P}'(C) = \langle C_{-1}' = C_{-1} \cup C_n \setminus C_0', C_0', C_1, \ldots, C_{n-1}\rangle$, and $h'(x) = h(x) + 1 \leq c_x$ for every $x \in C_1' \subseteq C_n$, $h'(x) = h(x) = c_x$ for every $x \in C_n \setminus C_1'$, and $h'(x) = h(x)$ otherwise.

In case $Y_1' = Y_2'$, then player 2 can pick the same time to elapse as player 1, and ensure that the conditions of the lemma hold. □

Lemma 14 indicates that if a state s belongs CPre_1 of a union of regions, then player 1 has a destination region R such that for any $s' \in \mathsf{Reg}(s)$, any move m_1' to R will take player 1 to the prescribed union of regions.

The proof of the first part of Lemma 1 then follows from the μ-calculus algorithm of [9] which uses the CPre_1 operator.

We now prove the second part of Lemma 1. Let π_1 be a pure region sure winning strategy for player 1 form $\underline{\mathsf{Sure}}_1^{\widehat{\mathfrak{T}}}(\mathsf{TimeDivBl}_1(\Phi))$, and let π_1' be surely region equivalent to π_1. Consider any strategy π_2 of player 2. We have $\mathsf{Outcomes}(s, \pi_1', \pi_2) = \{r \mid \forall k \geq 0 \, \exists m_1^k \in \mathsf{Support}(\pi_1'(r[0..k]))$ and $r[k+1] = \widehat{\delta}_{\mathsf{jd}}(r[k], m_1^k, \pi_2(r[0..k]))\}$. We then have $\mathsf{Outcomes}(s, \pi_1', \pi_2) = \{r \mid \exists \pi_1'' \in \Pi_1^{\mathsf{pure}} \forall k \geq 0 \, \pi_1''(r[0..k]) \in \mathsf{Support}(\pi_1'(r[0..k]))$ and $r[k+1] = \widehat{\delta}_{\mathsf{jd}}(r[k], \pi_1''(r[0..k]), \pi_2(r[0..k]))$ and π_1'' behaves like π_1 on other runs.$\}$. Now in the above set, each π_1'' is region equivalent to π_1, and hence is a winning strategy for player 1. Thus, in particular, $\mathsf{Outcomes}(s, \pi_1'', \pi_2) \subseteq \underline{\mathsf{Sure}}_1^{\widehat{\mathfrak{T}}}(\mathsf{TimeDivBl}_1(\Phi))$. taking the union over all π_1'', we have that $\mathsf{Outcomes}(s, \pi_1', \pi_2)$ is surely a subset of $\underline{\mathsf{Sure}}_1^{\widehat{\mathfrak{T}}}(\mathsf{TimeDivBl}_1(\Phi))$. A similar claim holds if π_1' is almost surely region equivalent to π_1. □

6.1 Proofs for Results on Safety Objectives

Proof of Lemma 3:
This result is a generalization of Lemma 2. Note that once a clock x becomes more than c_x, then its actual value can be considered irrelevant in determining regions. If only the clocks in $X \subseteq C$

have escaped beyond their maximum tracked values, the rest of the clocks still need to be tracked, and this gives rise to a sub-constraint ϕ_X for every $X \subseteq C$. □

Proof of Theorem 2:
We prove for the general case (where clocks might not be bounded).

1. If a state $\widetilde{s} \in \underline{\mathsf{Sure}}_1^{\widetilde{\mathcal{T}}}(\Box Y \wedge \Phi^*)$, then as in Lemma 3, there exists a receptive region strategy for player 1, and moreover this strategy ensures that the game stays in Y.
   If $s \notin \widetilde{s} \in \underline{\mathsf{Sure}}_1^{\widetilde{\mathcal{T}}}(\Box Y \wedge \Phi^*)$, then for every player-1 strategy π_1, there exists a player-2 strategy π_2 such that one of the resulting runs either violates $\Box Y$, or Φ^*. If Φ^* is violated, then π_1 is not a receptive strategy. If $\Box Y$ is violated, then player 2 can switch over to a receptive strategy as soon as the game gets outside Y. Thus, in both cases $s \notin \mathsf{Sure}_1^{\widetilde{\mathcal{T}}}(\Box Y)$.
2. Result similar to lemma 1 holds for the structure $\widetilde{\mathcal{T}}$. Since the objective Φ^* can be expressed as a Streett (strong fairness) objectives, it follows that player 1 has a pure finite-memory sure winning strategy for every state in $\underline{\mathsf{Sure}}_1^{\widetilde{\mathcal{T}}}((\Box Y) \wedge \Phi^*)$. The desired result then follows using the first part of the theorem.

□

6.2 Proofs for Results on Reachability Objectives

Proof of Lemma 4:
To show $\underline{\mathsf{Sure}}_1(\mathsf{TimeDivBl}(\mathsf{Reach}(S_R))) = \underline{\mathsf{Sure}}_1(\mathsf{Parity}(\Omega_R))$, we prove inclusion in both directions.

1. Suppose player 1 can win for the reachability objective S_R. Let π_1 be the winning strategy. Consider any player-2 strategy π_2, and any run $\widehat{r} \in \mathsf{Outcomes}(\langle s, 0, \text{FALSE}, \text{FALSE}\rangle, \pi_1, \pi_2)$. Suppose $\widehat{r}$ visits $\widehat{S_R}$. Then since S_R is absorbing, and all states in $\widehat{S_R}$ have index 0, only the index 0 is seen from some point on.
2. Suppose $\widehat{r}$ does not visit $\widehat{S_R}$, and let $\widehat{r}$ be time-diverging. If the moves of player 1 are chosen infinitely often in $\widehat{r}$, then the index 1 is visited infinitely often. If the moves of player 1 are chosen only finitely often, then from some point on, the clock z is reset only when it hits 1, and thus since time diverges, *tick* is true infinitely often. The index 1 is again visited infinitely often in this case.
   Suppose $\widehat{r}$ does not visit $\widehat{S_R}$, and let $\widehat{r}$ be time-converging. If the moves of player 1 are chosen infinitely often in $\widehat{r}$, then player 1 is to blame for blocking time. In this case 1 is visited infinitely often. If the moves of player 1 are only chosen finitely often, then again from some point on, the clock z is reset only when it hits 1. Since time does not diverge, *tick* is true only finitely often. Thus after some point, only the index 0 is seen, in agreement with the fact that player 1 is blameless.

□

Lemma 15 ([18]). *Let* $1 \geq \Delta_j \geq 0$ *for each* j. *Then,* $\lim_{n\to\infty} \prod_{j=1}^{n}(1 - \Delta_j) = 0$ *if* $\lim_{n\to\infty} \sum_{j=1}^{n} \Delta_j = \infty$.

Proof. Suppose $\Delta_j = 1$ for some j. Then, clearly $\lim_{n\to\infty} \prod_{j=1}^{n}(1 - \Delta_j) = 0$. Suppose $\Delta_j < 1$ for all j. We then have $\prod_{j=1}^{n}(1 - \Delta_j) > 0$ for all n. Consider $\ln\left(\prod_{j=1}^{n}(1 - \Delta_j)\right) = \sum_{j=1}^{n} \ln(1 - \Delta_j)$. Let $g(x) = x + \ln(1 - x)$. We have $g(0) = 0$ and $\frac{dg}{dx} = 1 - \frac{1}{1-x} = \frac{-x}{1-x} \leq 0$ for all $1 > x \geq 0$. Thus,

$g(x) \leq 0$ for all $1 > x \geq 0$. Hence, $0 \leq \Delta_j < -\ln(1-\Delta_j)$ for every j. Since $\lim_{n\to\infty}\sum_{j=1}^{n}\Delta_j = \infty$, we must have $\lim_{n\to\infty}\sum_{j=1}^{n}(-\ln(1-\Delta_j)) = \infty$, which means $\lim_{n\to\infty}\sum_{j=1}^{n}\ln(1-\Delta_j) = -\infty$. This in turn implies that $\lim_{n\to\infty}\prod_{j=1}^{n}(1-\Delta_j) = 0$. □

Proof of Lemma 11:
Consider a state $\langle s, \mathfrak{z}, tick, bl_1\rangle \in T_{2i+1}$ and let $s \in R$. All the states in T_{2i+1} have the property that player 1 can always guarantee that the next state has a lower rank, no matter what the move of player 2. Consider the player-2 move of $\langle 0, \bot\rangle$ at state $\langle s, \mathfrak{z}, tick, bl_1\rangle \in T_{2i+1}$. The next state is then going to be $\langle s, \mathfrak{z}, tick' = \text{FALSE}, bl_1' = \text{FALSE}\rangle$. Since $tick \vee bl_1 = \text{FALSE}$, the index of $\langle s, \mathfrak{z}, tick' = \text{FALSE}, bl_1' = \text{FALSE}\rangle$ is 0, and hence it must belong to an even rank which is lower than $2i+1$. Finally, we note that $\cup_{k=0}^{2i-1} T_k \subset T_{2i}$. □

Proof of Lemma 12:
The proof follows along similar line to that of Lemma 5. Let $\mathcal{A} = \{\widehat{s'} \mid \widehat{s'} \in R' \text{ and } R' \cap T_j \neq \emptyset \text{ for some } j < 2i\}$. Note that $\mathcal{A} \subseteq \mathsf{Reg}(T_{2i})$. We show player 1 can reach $\mathcal{A}$, without encountering a region R' such that $R' \cap (T_{2i} \cup \mathcal{A}) = \emptyset$. Let $\widehat{s} \in R$, with $R \cap T_{2i} \neq \emptyset$, and $R \cap T_j = \emptyset$ for all $2 \leq j < 2i$. The result follows from Lemmas 16, 17, 18, 19, and 20. □

Lemma 16. *Let R be a region of $\mathcal{T}$ such that $R \cap T_{2i} \neq \emptyset$, and $R \cap T_j = \emptyset$ for all $2 \leq j < 2i$. Then, player 1 has a move from every state in R to $\mathcal{A}$.*

Proof. Note that according to π_{pure}, player 1 always propose a move from $T_{2i} \cap R$ to $\mathcal{A}$ as the destination of the move of player 1 must be in rank $2i-1$ or lower (note that a move of player 1 being chosen makes the index 1). Thus, since player 1 has a move from $T_{2i} \cap R$ to $\mathcal{A}$ according to π_{pure}, he must have a move from every $\widehat{s} \in R$ to $\mathcal{A}$ by Lemma 13. □

Consider a state $\widehat{s}$ in some region $R' \subseteq \mathsf{Reg}(T_{2i})$ of $\mathcal{T}$. Now consider the set of times at which moves can be taken so that the state changes from $\widehat{s}$ to $\mathcal{A}$. This set consists of a finite union of sets I_k of the form $(\alpha_l, \alpha_r), [\alpha_l, \alpha_r), (\alpha_l, \alpha_r]$, or $[\alpha_l, \alpha_r]$ where α_l, α_r are of the form d or $d - x$ for d some integer constant, and x some clock in C (this clock x is the same for all the states in R'). Furthermore, these intervals have the property that $\{\widehat{s} + \Delta \mid \Delta \in I_k\} \subseteq R_k$ for some region R_k, with $R_l \cap R_j = \emptyset$ for $j \neq l$. From a state $\widehat{s}$, consider the "earliest" interval contained in this union: the interval I such that the left endpoint is the infimum of the times at which player 1 can move to $\mathcal{A}$. We have that $\{\widehat{s} + \Delta \mid \Delta \in I\} \subseteq R_1$. Consider any state $\widehat{s}' \in R'$. Then from s', the earliest interval in the times required to get to $\mathcal{A}$ is also of the form I. Note that in allowing time to pass to get to R_1, we may possibly go outside T_{2i} (recall that T_{2i} is not a union of regions of $\mathcal{T}$).

If this earliest interval is left closed, then player 1 has a "shortest" move to $\mathcal{A}$. Then, this is the best move in our strategy for player 1, and she will always propose this move. Let the left and the right endpoints of target intervals be α_l, α_r respectively. Then, if the target interval is left open, let player 1 play a probabilistic strategy with time distributed uniformly at random over $(\alpha_l, (\alpha_l + \alpha_r)/2]$. Let us denote this player-1 strategy by π_{rand}. Also note that the z, $tick$ and the bl_1 components play no role in determining the availability of moves.

Lemma 17. *Let R be a region of $\mathcal{T}$ such that $R \cap T_{2i} \neq \emptyset$, and $R \cap T_j = \emptyset$ for all $2 \leq j < 2i$. Then, the strategy π_{rand} ensures that from any state in R, the game stays in $\mathsf{Reg}(T_{2i})$ surely till $\mathcal{A}$ is visited.*

Proof. Let R be a region of $\mathcal{T}$ such that $R \cap T_{2i} \neq \emptyset$, and $R \cap T_j = \emptyset$ for all $2 \leq j < 2i$. Consider a state $\widehat{s} \in R \cap T_{2i}$. In π_{pure}, player 1 proposes a move to $\mathcal{A}$ from each state in $R \cap T_{2i}$. By Lemma 14, we have a unique set $M_2^{2i} = \{R' \mid$ player-2 moves to R' from R beat player-1 moves to $\mathcal{A}\}$. Since $\{\widehat{s} + t \mid t \in I\}$ constitutes a single region of $\mathcal{T}$, and I is the earliest interval that can land player 1 in $\mathcal{A}$, no new discrete actions become enabled due to the randomized strategy of player 1 — if player 2 can foil the randomized strategy of player 1 by taking a move to a region R' such that $R' \cap \mathsf{Reg}(T_{2i}) = \emptyset$, she can do so against π_{pure}. Thus, by induction using Lemma 14, we have that player 1 can guarantee with the randomized strategy that the game will stay in $\mathsf{Reg}(T_{2i})$ starting from a state in $R \cap T_{2i}$. Since the values z, *tick* and the bl_1 components play no role in determining the availability of moves, player 1 can ensure that the game states within $\mathsf{Reg}(T_{2i})$ starting from any state in a region R such that $R \cap T_{2i} \neq \emptyset$, and $R \cap T_j = \emptyset$ for all $2 \leq j < 2i$ till $\mathcal{A}$ is visited. ◻

If at any time the move of player 1 is chosen, then player 1 comes to $\mathcal{A}$. We show that when player 1 uses the randomized memoryless strategy π_{rand}, the probability of the move of player 1 being never chosen against a receptive strategy of player 2 is 0.

Lemma 18. *Let R be a region of $\mathcal{T}$ such that $R \cap T_{2i} \neq \emptyset$, and $R \cap T_j = \emptyset$ for all $2 \leq j < 2i$. Consider any receptive strategy π_2 of player 2, and a run $r \in \mathsf{Outcomes}(s, \pi_{\mathsf{rand}}, \pi_2)$ with $s \in R$. Suppose there exists $m \geq 0$ such that for all $k \geq m$, if $r[0..k]$ has not visited $\mathcal{A}$, then we have $\pi_{\mathsf{rand}}(r[0..k])$ to be left-closed. Then, we have that r visits $\mathcal{A}$.*

Proof. Note that if a move of player 1 is chosen at any point, then $\mathcal{A}$ is visited. Suppose the moves of player 1 are never chosen. Consider a run r against any strategy of player 2. Let us consider the run from $r[m]$ onwards. Only target left-closed regions occur form this point on. Let $r[m] = \widehat{s}' = \langle s', \mathfrak{z}, tick, bl_1\rangle \in R'$. Consider the pure winning strategy π_{pure} from a state $\widehat{s}'' = \langle s', \mathfrak{z}', tick', bl_1'\rangle \in R' \cap T_{2i}$ (such a state must exist). The state $\widehat{s}''$ differs from $\widehat{s}'$ only in the values of the clock z, and the boolean variables *tick* and bl_1. The new values do not affect the moves available to either player. Consider $\widehat{s}''$ as the starting state. The strategy π_{pure} cannot propose shorter moves to $\mathcal{A} \cap (\cup_{i=2}^{2i-1} T_j)$, since π_{rand} proposes the earliest move to $\mathcal{A}$. Hence, if a receptive player-2 strategy π_2 can prevent π_{rand} from reaching $\mathcal{A}$ from $\widehat{s}'$, then it can also prevent π_{pure} from reaching $\mathcal{A} \cap (\cup_{i=2}^{2i-1} T_j)$ from $\widehat{s}''$, a contradiction. ◻

Lemma 19. *Let R be a region of $\mathcal{T}$ such that $R \cap T_{2i} \neq \emptyset$, and $R \cap T_j = \emptyset$ for all $2 \leq j < 2i$. Consider any receptive strategy π_2 of player 2, and a run $r \in \mathsf{Outcomes}(s, \pi_{\mathsf{rand}}, \pi_2)$ with $s \in R$. There exists $m \geq 0$ such that for all $k \geq m$ if (a) $r[0..k]$ has not visited $\mathcal{A}$, and (b) $\pi_{\mathsf{rand}}(r[0..k])$ is left-open with left-endpoint being α_l, then we have $\alpha_l = 0$.*

Proof. Let α_l correspond to the left endpoint for one of the infinitely often occurring target left-open interval region R'.

1. We show that we cannot have α_l to be of the form d for some integer $d > 0$.
   We prove by contradiction. Suppose α_l is of the form d for some integer $d > 0$ for a region R'. Then, player 2 can always propose a time blocking move of d, this would mean that if the scheduler picks the move of player 2 (as both have the same delay), the next state will have *tick* true, no matter what the starting value of the clock z is. Now consider any state in $R' \cap T_{2i}$. The strategy π_{pure} always proposes some move to $\mathcal{A}$, and the time duration must be greater than d. Because of the d time-blocking move of player 2 new state will then be not in $\mathcal{A}$, and have *tick* = TRUE, hence, it will actually have an index of more than $2i$, contradicting the fact that π_{pure} ensured that the rank never decreased. Thus, $d > 0$ can never arise.

2. We show that we cannot have α_l to be of the form $d - x$ for some integer $d > 0$ and clock x. We prove by contradiction. Suppose clock x is not reset infinitely often in the run r. Then, the fact that the clock x has not progressed beyond d after some point in the run without being reset implies time is convergent, contradicting the fact that player 2 is playing with a receptive strategy (note that only moves of player 2 are being chosen). Thus, this situation cannot arise. Suppose x is reset infinitely often. Then, between a reset of x, and the time at which player 1 can jump to $\mathcal{A}$, we must have a time distance of more than d. Suppose R'' is one of the infinitely occurring regions in the run with the value of x being 0 in it. So player 2 has a strategy against our player-1 strategy such that one of the resulting runs contains a region subsequence $R'' \rightsquigarrow R'$. If this is so, then she would have a strategy which could do the same from every state in $R'' \cap T_{2i}$ against the pure winning strategy of player 1 (since the randomized strategy π_{rand} does not enable player 2 to go to more regions than against π_{pure}, as π_{rand} proposes moves to the earliest region in $\mathcal{A}$). But, if so, we have that *tick* will be true no matter what the starting value of z in $R'' \cap T_{2i}$, before player 1 can take a jump to $\mathcal{A}$ from $R' \cap T_{2i}$, taking the game outside of $\mathcal{A} \cup T_{2i}$. Since player 1 can stay inside T_{2i}, or visit $\mathcal{A}$ at each step with the infinite memory strategy π_{pure}, this cannot be so, that is, we cannot observe the region subsequence $R'' \rightsquigarrow R'$ for the player-1 randomized strategy. Hence the case of $\alpha_l = d - x$ cannot arise infinitely often.

The only remaining option is $\alpha_l = 0$, and we must have that the only randomized moves player 1 proposes after a while are of the form $(0, \alpha_r/2]$. ◻

Lemma 20. *Let R be a region of $\mathcal{T}$ such that $R \cap T_{2i} \neq \emptyset$, and $R \cap T_j = \emptyset$ for all $2 \leq j < 2i$. Consider any receptive strategy π_2 of player 2, and a strategy π_{sched} of the scheduler. Let $\mathcal{E}$ denote the set of runs containing runs $r \in \mathsf{Outcomes}(s, \pi_{\mathsf{rand}}, \pi_2)$ with $s \in R$. such that there exists $m \geq 0$ and for all $k \geq m$ (a) $r[0..k]$ has not visited $\mathcal{A}$, and (b) $\pi_{\mathsf{rand}}(r[0..k])$ is left-open with left-endpoint being $\alpha_l = 0$. Then, we have $\Pr_{r[0]}^{\pi_{\mathsf{rand}}, \pi_2, \pi_{\mathsf{sched}}}(\mathsf{Reach}(\mathcal{A}) \,|\, \mathcal{E}) = 1$.*

Proof. Let R' be one of the infinitely often occurring regions in r with the target left-endpoint being $\alpha_l = 0$. Let β_i be the duration of the move of player 2 for the ith visit to R' Suppose $\alpha_r = d$. Then the probability of a move of player 1 being never chosen is less than $\prod_{i=1}^{\infty}(1 - \frac{2\beta_i}{d})$, which is 0 if $\sum_{i=1}^{\infty} \beta_i = \infty$ by Lemma 15. A similar analysis holds if player 2 proposes randomized moves with a time distribution $\mathcal{D}^{(\beta_i, \cdot]}, \mathcal{D}^{[\beta_i, \cdot]}, \mathcal{D}^{(\beta_i, \cdot)}$ or $\mathcal{D}^{[\beta_i, \cdot)}$. Suppose $\alpha_r = d - x$. Suppose $\alpha_r = d - x$. Again, the probability of a move of player 1 being never chosen is less than $\prod_{i=1}^{\infty}(1 - \frac{2\beta_i}{(d - \kappa_i(x))})$, and since $\frac{\beta_i}{(d - \kappa_i(x))} > \frac{\beta_i}{d}$, this also is 0 if $\sum_{i=1}^{\infty} \beta_i = \infty$ by Lemma 15. Finally, we note that if player 2 does not block time from T_{2i}, then for at least one region, she must propose a β_i sequence such that $\sum_{i=1}^{\infty} \beta_i = \infty$, and we will have that for this region, a move of player 1 will be chosen eventually with probability 1. ◻