Qualitative Analysis of Concurrent Mean-payoff Games*

Krishnendu Chatterjee† Rasmus Ibsen-Jensen‡

Abstract

We consider concurrent games played by two-players on a finite-state graph, where in every round the players simultaneously choose a move, and the current state along with the joint moves determine the successor state. We study the most fundamental objective for concurrent games, namely, mean-payoff or limit-average objective, where a reward is associated to every transition, and the goal of player 1 is to maximize the long-run average of the rewards, and the objective of player 2 is strictly the opposite (i.e., the games are zero-sum). The path constraint for player 1 could be qualitative, i.e., the mean-payoff is the maximal reward, or arbitrarily close to it; or quantitative, i.e., a given threshold between the minimal and maximal reward. We consider the computation of the almost-sure (resp. positive) winning sets, where player 1 can ensure that the path constraint is satisfied with probability 1 (resp. positive probability). Almost-sure winning with qualitative constraint exactly corresponds to the question of whether there exists a strategy to ensure that the payoff is the maximal reward of the game. Our main results for qualitative path constraints are as follows: (1) we establish qualitative determinacy results that show for every state either player 1 has a strategy to ensure almost-sure (resp. positive) winning against all player-2 strategies, or player 2 has a spoiling strategy to falsify almost-sure (resp. positive) winning against all player-1 strategies; (2) we present optimal strategy complexity results that precisely characterize the classes of strategies required for almost-sure and positive winning for both players; and (3) we present quadratic time algorithms to compute the almost-sure and the positive winning sets, matching the best known bound of the algorithms for much simpler problems (such as reachability objectives). For quantitative constraints we show that a polynomial time solution for the almost-sure or the positive winning set would imply a solution to a long-standing open problem (of solving the value problem of turn-based deterministic mean-payoff games) that is not known to be solvable in polynomial time.

1 Introduction

Concurrent games. Concurrent games are played by two players (player 1 and player 2) on finite-state graphs for an infinite number of rounds. In every round, both players independently choose moves (or actions), and the current state along with the two chosen moves determine the successor state. In *deterministic* concurrent games, the successor state is unique; in *stochastic* concurrent games, the successor state is given by a probability distribution. The outcome of the game (or a *play*) is an infinite sequence of states and action pairs. These games were introduced in a seminal work by Shapley [31], and have been one of the most fundamental and well-studied game models in stochastic graph games. An important sub-class of concurrent

*The first author was supported by FWF Grant No P 23499-N23, FWF NFN Grant No S11407-N23 (RiSE), ERC Start grant (279307: Graph Games), and Microsoft faculty fellows award. Work of the second author supported by the Sino-Danish Center for the Theory of Interactive Computation, funded by the Danish National Research Foundation and the National Science Foundation of China (under the grant 61061130540). The second author acknowledge support from the Center for research in the Foundations of Electronic Markets (CFEM), supported by the Danish Strategic Research Council.

†IST Austria. Email: krish.chat@ist.ac.at

‡IST Austria. E-mail: ribsen@ist.ac.at

games are *turn-based* games, where in each state at most one player can choose between multiple moves (if the transition is stochastic we have turn-based stochastic games, and if the transition is deterministic we have turn-based deterministic games).

Mean-payoff (limit-average) objectives. The most well-studied objective for concurrent games is the *limit-average* (or mean-payoff) objective, where a reward is associated to every transition and the payoff of a play is the limit-inferior (or limit-superior) average of the rewards of the play. The original work of Shapley [31] considered *discounted* sum objectives (or games that stop with probability 1); and concurrent stochastic games with limit-average objectives (or games that have zero stop probabilities) was introduced by Gillette in [19]. The player-1 *value* $\mathrm{val}(s)$ of the game at a state s is the supremum value of the expectation that player 1 can guarantee for the limit-average objective against all strategies of player 2. The games are zero-sum where the objective of player 2 is the opposite. Concurrent limit-average games and many important sub-classes have received huge attention over the last five decades. The prominent sub-classes are turn-based games as restrictions of the game graphs, and *reachability* objectives as restrictions of the objectives. A reachability objective consists of a set U of *terminal* states (absorbing or sink states that are states with only self-loops), and the set U is exactly the set of states where out-going transitions are assigned reward 1 and all other transitions are assigned reward 0. Many celebrated results have been established for concurrent limit-average games and its sub-classes: (1) the existence of values (or determinacy or equivalence of switching of strategy quantifiers for the players as in von-Neumann's min-max theorem) for concurrent discounted games was established in [31]; (2) the existence of values (or determinacy) for concurrent reachability games was established in [18]; (3) the existence of values (or determinacy) for turn-based stochastic limit-average games was established in [27]; (4) the result of Blackwell-Fergusson established existence of values for the celebrated game of Big-Match [4]; and (5) developing on the results of [4] and Bewley-Kohlberg on Puisuex series [3] the existence of values for concurrent limit-average games was established in [28]. The decision problem of whether the value $\mathrm{val}(s)$ is at least a rational constant λ can be decided in PSPACE [11, 22]; and is *square-root sum* hard even for concurrent reachability games [17].[1] The algorithmic question of the value computation has also been studied in depth for special classes such as ergodic concurrent games [24] (where all states can be reached with probability 1 from all other states); turn-based stochastic reachability games [12]; and turn-based deterministic limit-average games [15, 34, 20, 6]. The decision problem of whether the value $\mathrm{val}(s)$ is at least a rational constant λ lie in NP $\cap$ coNP both for turn-based stochastic reachability games and turn-based deterministic limit-average games. They are among the rare and intriguing combinatorial problems that lie in NP $\cap$ coNP, but are not known to be in PTIME. The existence of polynomial time algorithms for the above decision questions are long-standing open problems.

Qualitative winning modes. In another seminal work, the notion of *qualitative winning* modes was introduced in [14] for concurrent reachability games. In qualitative winning modes, instead of the exact value computation the question is whether the objective can be satisfied with probability 1 (*almost-sure* winning) or with positive probability (*positive* winning). The qualitative analysis is of independent interest and importance in many applications (such as in system analysis) where we need to know whether the correct behaviour arises with probability 1. For instance, when analysing a randomized embedded scheduler, we are interested in whether every thread progresses with probability 1 [13]. Even in settings where it suffices to satisfy certain specifications with probability $p < 1$, the correct choice of p is a challenging problem, due to the simplifications introduced during modelling. For example, in the analysis of randomized distributed algorithms it is quite common to require correctness with probability 1 (see, e.g., [30, 26, 32]).

[1] The square-root sum problem is an important problem from computational geometry, where given a set of natural numbers $n_1, n_2, \ldots, n_k$, the question is whether the sum of the square roots exceed an integer b. The square root sum problem is not known to be in NP.

More importantly it was shown in [14] that the qualitative analysis for concurrent reachability games can be solved in polynomial time (quadratic time for almost-sure winning, and linear time for positive winning). Moreover the algorithms were discrete graph theoretic algorithms, and the combinatorial algorithms were independent of the precise transition probabilities. Since qualitative analysis is robust to numerical perturbations and modelling errors in the transition probabilities, and admits efficient combinatorial algorithms for the special case of concurrent reachability games, they have been studied in many different contexts such as Markov decision processes and turn-based stochastic games with ω-regular objectives [10]; pushdown stochastic games with reachability objectives [16, 17, 5]; and partial-observation games with ω-regular objectives [9, 2, 1, 8, 7, 29], to name a few. However, the qualitative analysis for the very important problem of concurrent limit-average games has not been studied before. In this work, we consider qualitative analysis of concurrent limit-average games, and the qualitative analysis is significantly different and involved as compared to concurrent reachability games. We first classify the various notion of strategies that are relevant for concurrent games.

Classes of strategies. In general a strategy in a concurrent game, considers the past history of the game (the finite sequence of states and actions played so far), and specifies a probability distribution over the next moves. Thus a strategy requires memory to remember the past history of the game. A strategy is *stationary* if it is independent of the past history and only depends on the current state; and a strategy is *positional* if it is stationary and does not use randomization. The complexity of a stationary strategy is described by its *patience* which is the inverse of the minimum non-zero probability assigned to a move. The notion of patience was introduced in [18] and also studied in the context of concurrent reachability games [23, 21]. A strategy is *Markov* if it only depends on the length of the play and the current state. For a strategy that requires infinite-memory, the *time-dependent* memory needed is the amount of memory required for first T rounds of the game, for $T > 0$. For example, the time-dependent memory required by a Markov strategy is T, for all $T > 0$. We first show with an example the difference between concurrent reachability games and concurrent limit-average games for qualitative analysis.

Example. Consider the classical game of *matching penny* where in every round player 1 and player 2 choose independently between two moves, namely, head and tail, and player 1 wins if the moves of both the players match in any round. The game of matching penny is modelled as a concurrent reachability game with two states s_0 and s_1, where s_1 is the terminal state. In s_0, both players choose head and tail, and if they match the successor state is s_1, otherwise s_0. A stationary strategy for player 1 that chooses both moves with equal probability is an almost-sure winning strategy. Consider a variant of the matching penny game where player 1 wins immediately if the matching moves are tails, but if the matching moves are heads, then player 1 gets a reward of 1 and the game continues. The classical matching penny game and the variant matching penny game are shown pictorially in Figure 1. For every $\epsilon > 0$, the stationary strategy for player 1 that plays head with probability $1 - \epsilon$ and tail with probability ϵ is an almost-sure winning strategy for the objective to ensure that the limit-average payoff is at least $1 - \epsilon$. For an almost-sure winning strategy for the objective to ensure that the limit-average payoff is exactly 1, infinite-memory strategies are required, and a Markov strategy that in round $j \geq 0$, for 2^{j^2}-steps plays tail with probability $\frac{1}{2^j}$ and head with probability $1 - \frac{1}{2^j}$, and then goes to round $j + 1$, is an almost-sure winning strategy. The variant matching penny game cannot be modeled as a concurrent reachability game.

Our results. First, note that for limit-average objectives the rewards can be scaled and shifted, and hence without loss of generality we restrict ourselves to the problem where the rewards are between 0 and 1. We consider three kinds of path constraints (or objectives): (i) *exact qualitative constraint* that consists of the set of paths where the limit-average payoff is 1; (ii) *limit qualitative constraint* that consists of the set of paths with limit-average payoff at least $1 - \epsilon$, for all $\epsilon > 0$; and (iii) *quantitative constraint* that consists of

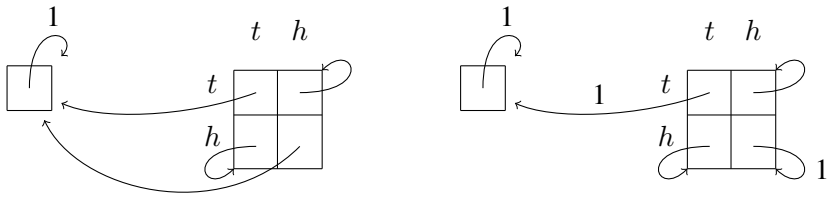

Figure 1: The classical matching pennies game (left) and our variant matching pennies game (right), where rewards 1 are annotated with the transition and all other rewards are 0.

		Exact Qual.	Limit Qual.	Reachability
Ensuring strategy	Sufficient	**Markov** **Time-dep:** T	**Stationary** **Patience** $\left(\frac{n \cdot m}{\epsilon}\right)^{n^{O(n)}}$	Stationary Patience m
	Necessary	**Infinite memory** **Time-dep:** T	**Stationary** **Patience** $\left(\frac{1}{\epsilon}\right)^{2^{\Omega(n)}}$	Stationary Patience m
Spoiling strategy	Sufficient	**Markov** **Time-dep:** T	**Markov** **Time-dep:** T	Markov Time-dep: T
	Necessary	**Infinite memory** **Time-dep:** T	**Infinite memory** **Time-dep:** T	Infinite memory Time-dep: T

Table 1: Strategy complexity for almost-sure winning for exact qualitative, limit qualitative constraints, and reachability objectives in concurrent games, where m is the number of moves and n is the number of states. The results in boldface are new results included in the present paper.

the set of paths where the limit-average payoff is at least λ, for $\lambda \in (0, 1)$. The significance of qualitative constraint are as follows: first, they present the most strict guarantee as path constraint; and second, almost-sure winning with qualitative constraint exactly corresponds to the question whether there exists a strategy to ensure that the payoff is the maximal reward of the transitions of the game. Our results are as follows:

1. *Almost-sure winning.* Our results for almost-sure winning are as follows:

   (a) *(Qualitative determinacy).* First we establish (in Section 3.1) *qualitative* determinacy for concurrent limit-average games for almost-sure winning where we show that for every state either player 1 has a strategy to ensure almost-sure winning for both exact qualitative constraint as well and limit qualitative constraint against all player-2 strategies; or player 2 has a spoiling strategy to ensure that both the exact qualitative constraint and limit qualitative constraint are violated with positive probability against all player-1 strategies. Observe that the qualitative determinacy result implies the equivalence of switching the order of quantifier of strategies for the players. The qualitative determinacy result is achieved by characterizing the almost-sure winning set with a discrete combinatorial iterative algorithm which is cubic time.

   (b) *(Strategy complexity).* In case of concurrent reachability games, stationary almost-sure winning strategies with patience m (where m is the number of moves) exist for player 1; and spoiling strategies for player 2 require infinite memory and Markov strategies are sufficient [14]. In contrast, we show that for exact qualitative path constraint, almost-sure winning strategies require

		Exact Qual.	Limit Qual.	Reachability
Ensuring strategy	Sufficient	**Markov** **Time-dep:** T	**Markov** **Time-dep:** T	Stationary Patience m
	Necessary	**Infinite memory** **Time-dep:** T	**Infinite memory** **Time-dep:** T	Stationary Patience m
Spoiling strategy	Sufficient	**Stationary** **Patience** m	**Stationary** **Patience** m	Positional
	Necessary	**Stationary** **Patience** m	**Stationary** **Patience** m	Positional

Table 2: Strategy complexity for positive winning for exact qualitative, limit qualitative constraints, and reachability objectives in concurrent games, where m is the number of moves. The results in boldface are new results included in the present paper.

infinite memory for player 1 and Markov strategies are sufficient; whereas the spoiling strategies require infinite memory for player 2 and Markov strategies are sufficient. For limit qualitative constraint, we show that for all $\epsilon > 0$, stationary almost-sure winning strategies exist for player 1, whereas spoiling strategies for player 2 require infinite memory and Markov strategies are sufficient. We establish asymptotically matching double exponential upper and lower bound for the patience required by almost-sure winning strategies for limit qualitative constraints. In all cases where infinite-memory strategies are required we establish that the optimal (matching upper and lower bound) time-dependent memory bound is T, for all $T > 0$. Our results are summarized in Table 1 (and the results are in Section 3.2).

(c) *(Improved algorithm).* Finally we present an improved algorithm for the computation of the almost-sure winning set of exact and limit qualitative constraint that is quadratic time (in Section 3.3). Our algorithm matches the bound of the currently best known algorithm for the computation of the almost-sure winning set of the special case of concurrent reachability games.

2. *Positive winning.* Our results for positive winning are as follows:

   (a) *(Qualitative determinacy and algorithm).* We establish the qualitative determinacy for positive winning; and our qualitative determinacy characterization already presents a quadratic time algorithm to compute the positive winning sets for exact and limit qualitative path constraints. Moreover, also for positive winning the exact and limit qualitative path constraints winning sets coincide. The results are presented in Section 4.1.

   (b) *(Strategy complexity).* In case of concurrent reachability games, stationary positive winning strategies with patience m (where m is the number of moves) exist for player 1; and positional (stationary and deterministic) spoiling strategies exist for player 2 [14]. In contrast, we show that positive winning strategies for player 1 both for exact and limit qualitative path constraints require infinite memory and Markov strategies are sufficient, and the optimal time-dependent memory bound is T, for all $T > 0$. We also show that (a) stationary spoiling strategies exist for player 2, (b) they require randomization, and (c) the optimal bound for patience is m. Our results are summarized in Table 2 (and the results are in Section 4.2).

3. *(Hardness of polynomial computability for quantitative constraints).* Finally we show (in Section 5)

that for quantitative path constraints, both the almost-sure and the positive winning problems even for turn-based stochastic games with rewards only $\{0, 1\}$ are at least as hard the value computation of turn-based deterministic mean-payoff games with arbitrary integer rewards. Thus solving the almost-sure or the positive winning problem with quantitative path constraint with boolean rewards in polynomial time would imply the solution of a long-standing open problem. Observe that we show hardness for the almost-sure and positive winning in turn-based stochastic boolean reward games with quantitative constraints. Note that (i) turn-based deterministic boolean reward games with quantitative constraints can be solved in polynomial time (the pseudo-polynomial time algorithm of [34] is polynomial for boolean rewards); (ii) almost-sure and positive winning for turn-based stochastic reachability games can be solved in polynomial time [10]; and (iii) almost-sure and positive winning with qualitative constraints can be solved in polynomial time as shown by our results (even for concurrent games). Thus our hardness result is tight in the sense that the natural restrictions in terms of game graphs, objectives, or qualitative constraints yield polynomial time algorithms.

Important remarks. Observe that for positive winning our algorithm is quadratic, as compared to the linear time algorithm for positive reachability in concurrent games. However, for the special case of turn-based deterministic games, the positive winning set for exact qualitative path constraints coincide with the winning set for coBüchi games (where the goal is to ensure that eventually always a set T of states are visited). The long-standing best known algorithms for turn-based deterministic coBüchi games are quadratic, which are a special case of positive winning for concurrent limit-average games with qualitative path constraints. Therefore our algorithm matches the current best known quadratic bound of the simpler case. Finally, our results that for qualitative analysis Markov strategies are sufficient are in sharp contrast to general concurrent limit-average games where Markov strategies are not sufficient (for example in the celebrated Big-Match game [4]).

2 Definitions

In this section we present the definitions of game structures, strategies, objectives, winning modes and other basic notions.

Probability distributions. For a finite set A, a *probability distribution* on A is a function $\delta : A \to [0, 1]$ such that $\sum_{a \in A} \delta(a) = 1$. We denote the set of probability distributions on A by $\mathcal{D}(A)$. Given a distribution $\delta \in \mathcal{D}(A)$, we denote by $\mathrm{Supp}(\delta) = \{x \in A \mid \delta(x) > 0\}$ the *support* of the distribution δ.

Concurrent game structures. A (two-player) *concurrent stochastic game structure* $G = (S, A, \Gamma_1, \Gamma_2, \delta)$ consists of the following components.

- A finite state space S and a finite set A of actions (or moves).
- Two move assignments $\Gamma_1, \Gamma_2 : S \to 2^A \setminus \emptyset$. For $i \in \{1, 2\}$, assignment Γ_i associates with each state $s \in S$ the non-empty set $\Gamma_i(s) \subseteq A$ of moves available to player i at state s. For technical convenience, we assume that $\Gamma_i(s) \cap \Gamma_j(t) = \emptyset$ unless $i = j$ and $s = t$, for all $i, j \in \{1, 2\}$ and $s, t \in S$. If this assumption is not met, then the moves can be trivially renamed to satisfy the assumption.
- A probabilistic transition function $\delta : S \times A \times A \to \mathcal{D}(S)$, which associates with every state $s \in S$ and moves $a_1 \in \Gamma_1(s)$ and $a_2 \in \Gamma_2(s)$ a probability distribution $\delta(s, a_1, a_2) \in \mathcal{D}(S)$ for the successor state.

We will denote by $\delta_{\min}$ the minimum non-zero transition probability, i.e., $\delta_{\min} = \min_{s,t \in S} \min_{a_1 \in \Gamma_1(s), a_2 \in \Gamma_2(s)} \{\delta(s,a_1,a_2)(t) \mid \delta(s,a_1,a_2)(t) > 0\}$. We will denote by n the number of states (i.e., $n = |S|$), and by m the maximal number of actions available for a player at a state (i.e., $m = \max_{s \in S} \max\{|\Gamma_1(s)|, |\Gamma_2(s)|\}$). For all states $s \in S$, moves $a_1 \in \Gamma_1(s)$ and $a_2 \in \Gamma_2(s)$, we indicate by $\mathrm{Dest}(s,a_1,a_2) = \mathrm{Supp}(\delta(s,a_1,a_2))$ the set of possible successors of s when moves a_1 and a_2 are selected. The size of the transition relation of a game structure is defined as $|\delta| = \sum_{s \in S} \sum_{a_1 \in \Gamma_1(s)} \sum_{a_2 \in \Gamma_2(s)} |\mathrm{Dest}(s,a_1,a_2)|$.

Turn-based stochastic games, turn-based deterministic games and MDPs. A game structure G is *turn-based stochastic* if at every state at most one player can choose among multiple moves; that is, for every state $s \in S$ there exists at most one $i \in \{1,2\}$ with $|\Gamma_i(s)| > 1$. A turn-based stochastic game with deterministic transition function is a turn-based deterministic game. A game structure is a player-2 *Markov decision process (MDP)* if for all $s \in S$ we have $|\Gamma_1(s)| = 1$, i.e., only player 2 has choice of actions in the game, and player-1 MDPs are defined analogously.

Plays. At every state $s \in S$, player 1 chooses a move $a_1 \in \Gamma_1(s)$, and simultaneously and independently player 2 chooses a move $a_2 \in \Gamma_2(s)$. The game then proceeds to the successor state t with probability $\delta(s,a_1,a_2)(t)$, for all $t \in S$. A *path* or a *play* of G is an infinite sequence $\omega = \big((s_0,a_1^0,a_2^0),(s_1,a_1^1,a_2^1),(s_2,a_1^2,a_2^2)\dots\big)$ of states and action pairs such that for all $k \geq 0$ we have (i) $a_1^k \in \Gamma_1(s_k)$ and $a_2^k \in \Gamma_2(s_k)$; and (ii) $s_{k+1} \in \mathrm{Supp}(\delta(s_k,a_1^k,a_2^k))$. We denote by Ω the set of all paths.

Strategies. A *strategy* for a player is a recipe that describes how to extend prefixes of a play. Formally, a strategy for player $i \in \{1,2\}$ is a mapping $\sigma_i : (S \times A \times A)^* \times S \to \mathcal{D}(A)$ that associates with every finite sequence $x \in (S \times A \times A)^*$ of state and action pairs, and the current state s in S, representing the past history of the game, a probability distribution $\sigma_i(x \cdot s)$ used to select the next move. The strategy σ_i can prescribe only moves that are available to player i; that is, for all sequences $x \in (S \times A \times A)^*$ and states $s \in S$, we require that $\mathrm{Supp}(\sigma_i(x \cdot s)) \subseteq \Gamma_i(s)$. We denote by Σ_i the set of all strategies for player $i \in \{1,2\}$. Once the starting state s and the strategies σ_1 and σ_2 for the two players have been chosen, the probabilities of events are uniquely defined [33], where an *event* $\mathcal{A} \subseteq \Omega$ is a measurable set of paths. For an event $\mathcal{A} \subseteq \Omega$, we denote by $\Pr_s^{\sigma_1,\sigma_2}(\mathcal{A})$ the probability that a path belongs to $\mathcal{A}$ when the game starts from s and the players use the strategies σ_1 and σ_2. We will consider the following special classes of strategies:

1. *Stationary (memoryless) and positional strategies.* A strategy σ_i is *stationary* (or memoryless) if it is independent of the history but only depends on the current state, i.e., for all $x, x' \in (S \times A \times A)^*$ and all $s \in S$, we have $\sigma_i(x \cdot s) = \sigma_i(x' \cdot s)$, and thus can be expressed as a function $\sigma_i : S \to \mathcal{D}(A)$. For stationary strategies, the complexity of the strategy is described by the *patience* of the strategy, which is the inverse of the minimum non-zero probability assigned to an action [18]. Formally, for a stationary strategy $\sigma_i : S \to \mathcal{D}(A)$ for player i, the patience is $\max_{s \in S} \max_{a \in \Gamma_i(s)} \{\frac{1}{\sigma_i(s)(a)} \mid \sigma_i(s)(a) > 0\}$. A strategy is *pure (deterministic)* if it does not use randomization, i.e., for any history there is always some unique action a that is played with probability 1. A pure stationary strategy σ_i is also called a *positional* strategy, and represented as a function $\sigma_i : S \to A$.

2. *Strategies with memory and finite-memory strategies.* A strategy σ_i can be equivalently defined as a pair of functions (σ_i^u, σ_i^n), along with a set Mem of memory states, such that (i) the next move function $\sigma_i^n : S \times \mathsf{Mem} \to \mathcal{D}(A)$ given the current state of the game and the current memory state specifies the probability distribution over the actions; and (ii) the memory update function $\sigma_i^u : S \times A \times A \times \mathsf{Mem} \to \mathsf{Mem}$ given the current state of the game, the action pairs, and the current

memory state updates the memory state. Any strategy can be expressed with an infinite set Mem of memory states, and a strategy is a *finite-memory* strategy if the set Mem of memory states is finite, otherwise it is an *infinite-memory* strategy.

3. *Markov strategies.* A strategy σ_i is a *Markov strategy* if it only depends on the length of the play and current state. Formally, for all finite prefixes $x, x' \in (S \times A \times A)^*$ such that $|x| = |x'|$ (i.e., the length of x and x' are the same, where the length of x and x' are the number of states that appear in x and x', respectively) and all $s \in S$ we have $\sigma_i(x \cdot s) = \sigma_i(x' \cdot s)$.

4. *Time-dependent memory.* Consider a strategy σ_i with memory Mem. For every finite sequence $x \in (S \times A \times A)^*$ there is an unique memory element $t(x) = m \in \mathsf{Mem}$ such that after the finite sequence x the current memory state is m (note that the memory update function is a deterministic function). For a time bound T, the time-dependent memory of the strategy σ_i, is the size of the set of memory elements used for histories upto length T, i.e., $|\{m \in \mathsf{Mem} \mid \exists x \in (S \times A \times A)^*, |x| \leq T, t(x) = m\}|$. Note that a Markov strategy can be played with time-dependent memory of size T, for all $T > 0$. A trivial upper bound on the time-dependent memory of a strategy is $(|S| \cdot |A| \cdot |A|)^T$, for all $T \geq 0$.

Repeated games with absorbing states. A state s is *absorbing* if for all actions $a_1 \in \Gamma_1(s)$ and all actions $a_2 \in \Gamma_2(s)$ we have $\mathrm{Dest}(s, a_1, a_2) = \{s\}$. A game is a *repeated game with absorbing states* if all states, other than one special state s^*, are absorbing. In the present paper all absorbing states will only have a single action for each player. Hence, no strategy will need memory, once an absorbing state is reached. In a repeated game with absorbing states, updates of memory will therefore only happen in state s^* and implies that the play has only been in state s^* since the start of the play. We will therefore write $\sigma_i^u(s^*, a_1, a_2, m)$ as $\sigma_i^u(a_1, a_2, m)$.

Objectives. An objective $\Phi \subseteq \Omega$ is a measurable subset of paths. In this work we will consider *limit-average* (or mean-payoff) objectives. We will consider concurrent games with a *boolean* reward function $\mathrm{r} : S \times A \times A \to \{0, 1\}$ that assigns a reward value $\mathrm{r}(s, a_1, a_2)$ for all $s \in S$, $a_1 \in \Gamma_1(s)$ and $a_2 \in \Gamma_2(s)$ (see Remark 19 for general real-valued reward functions[2]). For a path $\omega = \big((s_0, a_1^0, a_2^0), (s_1, a_1^1, a_2^1), \dots\big)$, the limit-inferior average (resp. limit-superior average) is defined as follows: $\mathsf{LimInfAvg}(\omega) = \liminf_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \mathrm{r}(s_i, a_1^i, a_2^i)$ (resp. $\mathsf{LimSupAvg}(\omega) = \limsup_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \mathrm{r}(s_i, a_1^i, a_2^i)$). For a threshold $\lambda \in [0, 1]$ we consider the following objectives:

$$
\begin{array}{llll}
\mathsf{LimInfAvg}(\lambda) & = \{\omega \mid \mathsf{LimInfAvg}(\omega) \geq \lambda\}; & \mathsf{LimSupAvg}(\lambda) & = \{\omega \mid \mathsf{LimSupAvg}(\omega) \geq \lambda\}; \\
\overline{\mathsf{LimInfAvg}}(\lambda) & = \{\omega \mid \mathsf{LimInfAvg}(\omega) < \lambda\}; & \overline{\mathsf{LimSupAvg}}(\lambda) & = \{\omega \mid \mathsf{LimSupAvg}(\omega) < \lambda\}; \\
\overline{\mathsf{LimInfAvg}}_{\leq}(\lambda) & = \{\omega \mid \mathsf{LimInfAvg}(\omega) \leq \lambda\}; & \overline{\mathsf{LimSupAvg}}_{\leq}(\lambda) & = \{\omega \mid \mathsf{LimSupAvg}(\omega) \leq \lambda\}.
\end{array}
$$

For the analysis of concurrent games with boolean limit-average objectives we will also need *reachability* and *safety* objectives. Given a target set $U \subseteq S$, the reachability objective $\mathsf{Reach}(U)$ requires some state in U be visited at least once, i.e., defines the set $\mathsf{Reach}(U) = \{\omega = \big((s_0, a_1^0, a_2^0), (s_1, a_1^1, a_2^1), \dots\big) \mid \exists i \geq 0. s_i \in U\}$ of paths. The dual safety objective for a set $F \subseteq S$ of safe states requires that the set F is never left, i.e., $\mathsf{Safe}(F) = \{\omega = \big((s_0, a_1^0, a_2^0), (s_1, a_1^1, a_2^1), \dots\big) \mid \forall i \geq 0. s_i \in F\}$. Observe that reachability

[2]We consider boolean rewards for simplicity in presentation, and in Remark 19 we argue how the results extend to general rewards.

objectives are a very special case of boolean reward limit-average objectives where states in U are absorbing and are exactly the states with reward 1, and similarly for safety objectives.

μ-calculus, complementation, and levels. Consider a μ-calculus expression $\Psi = \mu X.\psi(X)$ over a finite set S, where $\psi : 2^S \mapsto 2^S$ is monotonic. The least fixpoint $\Psi = \mu X.\psi(X)$ is equal to the limit $\lim_{k\to\infty} X_k$, where $X_0 = \emptyset$, and $X_{k+1} = \psi(X_k)$. For every state $s \in \Psi$, we define the *level* $k \geq 0$ of s to be the integer such that $s \notin X_k$ and $s \in X_{k+1}$. The greatest fixpoint $\Psi = \nu X.\psi(X)$ is equal to the limit $\lim_{k\to\infty} X_k$, where $X_0 = S$, and $X_{k+1} = \psi(X_k)$. For every state $s \notin \Psi$, we define the *level* $k \geq 0$ of s to be the integer such that $s \in X_k$ and $s \notin X_{k+1}$. The *height* of a μ-calculus expression $\gamma X.\psi(X)$, where $\gamma \in \{\mu, \nu\}$, is the least integer h such that $X_h = \lim_{k\to\infty} X_k$. An expression of height h can be computed in $h + 1$ iterations. Given a μ-calculus expression $\Psi = \gamma X.\psi(X)$, where $\gamma \in \{\mu, \nu\}$, the complement $\neg\Psi = S \setminus \Psi$ of γ is given by $\overline{\gamma}X.\neg\psi(\neg X)$, where $\overline{\gamma} = \mu$ if $\gamma = \nu$, and $\overline{\gamma} = \nu$ if $\gamma = \mu$.

Almost-sure and positive winning sets. Given an objective Φ, the *almost-sure winning set* for player 1 for the objective Φ, denoted as $\mathsf{Almost}_1(\Phi)$, is the set of states such that there exists a strategy (referred to as almost-sure winning strategy) for player 1 to ensure that the objective is satisfied with probability 1 (almost-surely) against all strategies of the opponent. The *positive winning set*, denoted $\mathsf{Positive}_1(\Phi)$, requires that player 1 can ensure that the probability to satisfy Φ is positive. Formally we have

- $\mathsf{Almost}_1(\Phi) = \{s \mid \exists\sigma_1.\forall\sigma_2.\, \Pr_s^{\sigma_1,\sigma_2}(\Phi) = 1\}$; and
- $\mathsf{Positive}_1(\Phi) = \{s \mid \exists\sigma_1.\forall\sigma_2.\, \Pr_s^{\sigma_1,\sigma_2}(\Phi) > 0\}$.

The almost-sure and positive winning sets Almost_2 and $\mathsf{Positive}_2$ for player 2 are obtained analogously, as above, by switching the roles of player 1 and player 2, respectively.

3 Almost-sure Winning

In this section we will present three results: (1) establish *qualitative* determinacy for almost-sure winning; (2) establish the strategy complexity for almost-sure winning; and (3) finally present an improved algorithm to compute the almost-sure winning set; for exact and limit qualitative constraints in concurrent games.

3.1 Qualitative determinacy

We will establish the qualitative determinacy results through a polynomial time algorithm to compute the set $\mathsf{Almost}_1(\mathsf{LimInfAvg}(\lambda))$ and $\mathsf{Almost}_1(\mathsf{LimSupAvg}(\lambda))$ for $\lambda = 1$ in concurrent games with boolean rewards. To present our algorithm we first define a three-argument predecessor operator $\mathrm{ASP}(X, Y, Z)$ and then give our algorithm as a μ-calculus formula with the predecessor operator.

Predecessor operator. Consider sets $X, Y, Z \subseteq S$ such that $Y \subseteq Z \subseteq X$; and we consider the following three sets of actions:

1. $\mathrm{Allow}_1(s, X) = \{a_1 \in \Gamma_1(s) \mid \forall a_2 \in \Gamma_2(s) : \mathrm{Dest}(s, a_1, a_2) \subseteq X\}$;
2. $\mathrm{Bad}_2(s, X, Y) = \{a_2 \in \Gamma_2(s) \mid \exists a_1 \in \mathrm{Allow}_1(s, X) : \mathrm{Dest}(s, a_1, a_2) \cap Y \neq \emptyset\}$;
3. $\mathrm{Good}_1(s, X, Y, Z) = \{a_1 \in \mathrm{Allow}_1(s, X) \mid \forall a_2 \in \Gamma_2(s) \setminus \mathrm{Bad}_2(s, X, Y) : \mathrm{Dest}(s, a_1, a_2) \subseteq Z \wedge \mathrm{r}(s, a_1, a_2) = 1\}$.

The intuitive description of the action sets are as follows: (i) the set $\text{Allow}_1(s, X)$ consists of all actions for player 1, such that against all actions of player 2 the set X is not left; (ii) the set $\text{Bad}_2(s, X, Y)$ is the set of player 2 actions a_2, such that there is a player 1 action a_1 in $\text{Allow}_1(s, X)$ such that given a_1 and a_2 the set Y is reached in one-step from s with positive probability; and (iii) $\text{Good}_1(s, X, Y, Z)$ is the set of actions for player 1 in $\text{Allow}_1(s, X)$ such that for all actions for player 2 that are not in $\text{Bad}_2(s, X, Y)$ the next state is in Z and the reward is 1. The set $\text{ASP}(X, Y, Z)$ is the set of states where $\text{Good}_1(s, X, Y, Z)$ is non-empty, i.e., $\text{ASP}(X, Y, Z) = \{s \mid \text{Good}_1(s, X, Y, Z) \neq \emptyset\}$ (the word ASP is an acronym for allow-stay-progress, i.e., (i) it allows the play to remain in X with probability 1, and either (ii) progress to Y with positive probability or (iii) stay in Z and get reward 1 with high probability). Let $X^* = \nu X.\mu Y.\nu Z.\text{ASP}(X, Y, Z)$ be the fixpoint. We will show that

$$X^* = \mathsf{Almost}_1(\mathsf{LimInfAvg}(1)) = \mathsf{Almost}_1(\mathsf{LimSupAvg}(1)).$$

Moreover we will show that $X^* = \bigcap_{\epsilon>0} \mathsf{Almost}_1(\mathsf{LimInfAvg}(1-\epsilon)) = \bigcap_{\epsilon>0} \mathsf{Almost}_1(\mathsf{LimSupAvg}(1-\epsilon))$. In the following two lemmas we establish that $X^* \subseteq \mathsf{Almost}_1(\mathsf{LimInfAvg}(1)) \subseteq \mathsf{Almost}_1(\mathsf{LimSupAvg}(1))$ as follows: (1) in the first lemma we show that for all $\epsilon > 0$ there is a stationary strategy to ensure from all states in X^* that the limit-inferior mean-payoff is at least $1 - \epsilon$ and the set X^* is never left; and (2) in the second lemma we use the stationary strategies of the first lemma repeatedly to construct a Markov almost-sure winning strategy.

Lemma 1. *For all $\epsilon > 0$, there exists a stationary strategy σ_1^ϵ with patience at most $(\frac{n \cdot m}{\delta_{\min} \cdot \epsilon})^{n^{n+2}}$, such that for all strategies σ_2 and all $s \in X^*$ we have $\Pr_s^{\sigma_1^\epsilon, \sigma_2}(\mathsf{LimInfAvg}(1-\epsilon) \cap \mathsf{Safe}(X^*)) = 1$.*

Proof. We first analyse the computation of X^*. We have

$$X^* = \mu Y.\nu Z.\text{ASP}(X^*, Y, Z);$$

this is achieved by simply replacing X with X^* in the μ-calculus expression $\nu X.\mu Y.\nu Z.\text{ASP}(X, Y, Z)$, then getting rid of the outer-most ν quantifier, and evaluating the rest of the μ-calculus expression. Since X^* is a fixpoint we have $X^* = \mu Y.\nu Z.\text{ASP}(X^*, Y, Z)$. Thus the computation of X^* is achieved as follows: we have $Y_0 = \emptyset$ and $Y_{i+1} = \nu Z.\text{ASP}(X^*, Y_i, Z)$. Let ℓ be the smallest number such that $Y_\ell = \nu Z.\text{ASP}(X^*, Y_\ell, Z)$, and we have $Y_\ell = X^*$. For a state $s \in X^*$, let $Aw(s) = |\text{Allow}_1(s, X^*)|$ denote the size of the set of allowable actions; and for $j \geq 0$, let $Gd(s, j) = |\text{Good}_1(s, X^*, Y_{\ell-j-1}, Y_{\ell-j})|$ denote the size of the set of good actions for the triple $X^*, Y_{\ell-j-1}, Y_{\ell-j}$.

Fix $\epsilon > 0$. The desired strategy σ_1^ϵ will be constructed as a finite sequence of strategies $\sigma_1^{\epsilon,1}, \sigma_1^{\epsilon,2}, \ldots \sigma_1^{\epsilon,\ell}$. We start with the definition of $\sigma_1^{\epsilon,1}$, and for a state s and action $a \in \Gamma_1(s)$ we have the following: if $s \in Y_\ell \setminus Y_{\ell-1}$, then

$$\sigma_1^{\epsilon,1}(s)(a) = \begin{cases} \frac{1-\epsilon}{Gd(s,0)} & a \in \text{Good}_1(s, X^*, Y_{\ell-1}, Y_\ell) \text{ and } Aw(s) \neq Gd(s,0) \\ \frac{1}{Gd(s,0)} & a \in \text{Good}_1(s, X^*, Y_{\ell-1}, Y_\ell) \text{ and } Aw(s) = Gd(s,0) \\ \frac{\epsilon}{Aw(s)-Gd(s,0)} & a \in \text{Allow}_1(s, X^*) \setminus \text{Good}_1(s, X^*, Y_{\ell-1}, Y_\ell) \\ 0 & a \notin \text{Allow}_1(s, X^*) \end{cases}$$

and if $s \notin (Y_\ell \setminus Y_{\ell-1})$, then $\sigma_1^{\epsilon,1}(s)$ is an arbitrary probability distribution over $\Gamma_1(s)$. For $j > 1$, let $\beta_j = n^{-\frac{n^{j-1}-1}{n-1}} \cdot (\frac{m}{\delta_{\min}})^{-\frac{n^j-1}{n-1}+1} \cdot \epsilon^{\frac{n^j-1}{n-1}}$. We now define $\sigma_1^{\epsilon,j}$, for $j > 1$. For a state s and action $a \in \Gamma_1(s)$,

we have the following: if $s \in Y_{\ell-j} \setminus Y_{\ell-j-1}$,

$$\sigma_1^{\epsilon,j}(s)(a) = \begin{cases} \frac{1-\beta_j}{Gd(s,j)} & a \in \text{Good}_1(s, X^*, Y_{\ell-j-1}, Y_{\ell-j}) \text{ and } Aw(s) \neq Gd(s,j) \\ \frac{1}{Gd(s,j)} & a \in \text{Good}_1(s, X^*, Y_{\ell-j-1}, Y_{\ell-j}) \text{ and } Aw(s) = Gd(s,j) \\ \frac{\beta_j}{Aw(s)-Gd(s,j)} & a \in \text{Allow}_1(s, X^*) \setminus \text{Good}_1(s, X^*, Y_{\ell-j-1}, Y_{\ell-j}) \\ 0 & a \notin \text{Allow}_1(s, X^*) \end{cases}$$

and if $s \notin (Y_{\ell-j} \setminus Y_{\ell-j-1})$, then $\sigma_1^{\epsilon,j}(s)(a) = \sigma_1^{\epsilon,j-1}(s)(a)$. The strategy σ_1^ϵ is then $\sigma_1^{\epsilon,\ell}$.

Bounds on patience. Observe that the patience is at most $(\frac{n \cdot m}{\delta_{\min} \cdot \epsilon})^{n^{n+2}}$, because that is a bound on the inverse of β_ℓ, by definition (note that ℓ is at most n).

We will now show that σ_1^ϵ has the desired properties to ensure the safety and limit-average objectives.

Ensuring safety. First observe that the strategy σ_1^ϵ plays actions not in $\text{Allow}_1(s, X^*)$ with probability 0, for states $s \in X^*$. For all actions $a_1 \in \Gamma_1(s)$, if there is an action $a_2 \in \Gamma_2(s)$ such that $\text{Dest}(s, a_1, a_2) \cap (S \setminus X^*) \neq \emptyset$, then a_1 does not belong to $\text{Allow}_1(s, X^*)$ and hence is played with probability 0 (at s for all $s \in X^*$). This implies that for all $s' \in X^*$ and for all strategies σ_2 we have that $\Pr_{s'}^{\sigma_1^\epsilon, \sigma_2}(\mathsf{Safe}(X^*)) = 1$. Hence the safety property is guaranteed.

Ensuring $\mathsf{LimInfAvg}(1-\epsilon)$. We now focus on the mean-payoff objective. Since the strategy σ_1^ϵ is a stationary strategy, fixing the strategy σ_1^ϵ for player 1, we obtain an MDP for player 2. In MDPs, there exist optimal positional strategies for the player to minimize mean-payoff objectives [27]. Hence we only focus on positional strategies as counter strategies for player 2 against σ_1^ϵ.

We will show the following by induction on j: for all positional strategies σ_2 for player 2, for all $s \in X^* \setminus Y_{\ell-j}$ one of the following two properties hold: either (1) the set $Y_{\ell-j}$ is reached within at most $\frac{\epsilon}{\beta_{j+1}}$ steps in expectation; or (2) we have

$$\Pr_s^{\sigma_1^{\epsilon,j}, \sigma_2}(\mathsf{LimInfAvg}(1-\epsilon)) = 1. \tag{1}$$

We present the inductive proof now.

Base case. First the base case, $j = 1$. Let $s \in Y_\ell \setminus Y_{\ell-1} = X^* \setminus Y_{\ell-1}$. Consider $\sigma_1^{\epsilon,1}$ and a positional strategy σ_2 for player 2. After fixing both the chosen strategies, since both the strategies are stationary we obtain a Markov chain. Let the random variable indicating the play from s in the Markov chain be denoted as P. There are now two cases.

- We consider the case when the play P enters a state s_1 from which no state s_2 can be reached, where σ_2 plays an action in $\text{Bad}_2(s_2, X^*, Y_{\ell-1})$. Then once s_1 is reached, for any state s_3 that appears after s_1 we have that $\sigma_1^{\epsilon,1}$ plays some action in $\text{Good}_1(s_3, X^*, Y_{\ell-1}, Y_\ell)$ with probability $1-\epsilon$ (and hence get a payoff of 1). Hence P satisfies Equation 1 in this case.

- In the other case the play P can always reach a state s_2 such that σ_2 plays an action in $\text{Bad}_2(s_2, Y_\ell, Y_{\ell-1}) = \text{Bad}_2(s_2, X^*, Y_{\ell-1})$ and we can therefore enter a state in $s_3 \in Y_{\ell-1}$ with probability at least $\frac{\epsilon \cdot \delta_{\min}}{m}$ from s_2. Since $\frac{\epsilon \cdot \delta_{\min}}{m}$ is a lower bound on the smallest positive probability in the Markov chain, any state that can be reached is actually reached within at most n steps with probability at least $(\frac{\epsilon \cdot \delta_{\min}}{m})^n$. Hence the probability to reach a state in $Y_{\ell-1}$ within at most n steps is at least $(\frac{\epsilon \cdot \delta_{\min}}{m})^n$. Therefore we need at most $n \cdot (\frac{m}{\epsilon \cdot \delta_{\min}})^n = \frac{\epsilon}{\beta_2}$ steps in expectation to reach $Y_{\ell-1}$.

Inductive case. We now consider the inductive case for $j > 1$, and the argument is similar to the base case. Let $s \in X^* \setminus Y_{\ell-j}$. As above we fix a positional strategy σ_2 for player 2 and consider the Markov chain induced by σ_1^ϵ and σ_2, and denote the random variable for a play from s in the Markov chain as P. We have two cases.

- We consider the case when the play P enters a state s_1 from which no state s_2 can be reached, where σ_2 plays an action in $\mathrm{Bad}_2(s_1, X^*, Y_{\ell-j})$. Hence once s_1 is reached along P, no state in $Y_{\ell-j}$ can be reached along the play. We can view the states in $Y_\ell \setminus Y_{\ell-j+1}$ as either (i) already satisfying the desired Equation 1 by the inductive hypothesis, or (ii) else entering a state in $Y_{\ell-j+1} \setminus Y_{\ell-j}$ (no state in $Y_{\ell-j}$ can be reached) after having giving payoff at least 0 for at most $\frac{\epsilon}{\beta_j}$ time steps in expectation (by the inductive hypothesis). But in all states s_3 in $Y_{\ell-j+1} \setminus Y_{\ell-j}$ that the play P visits after entering the state s_1, the strategy $\sigma_1^{\epsilon,0}$ chooses an action in $\mathrm{Good}_1(s_3, X^*, Y_{\ell-j}, Y_{\ell-j+1})$ with probability $1 - \beta_j$; and hence from s_3 the play P enters another state in $Y_{\ell-j+1} \setminus Y_{\ell-j}$ and get payoff 1. If we do not get payoff 1 in any state in $Y_{\ell-j+1} \setminus Y_{\ell-j}$, we expect to get at most $\frac{\epsilon}{\beta_j}$ times payoff 0 and then again enter some state in $Y_{\ell-j+1} \setminus Y_{\ell-j}$. Hence the probability that any given payoff is 0 is at most

$$\frac{\beta_j \cdot \frac{\epsilon}{\beta_j}}{1 - \beta_j + \beta_j \cdot \frac{\epsilon}{\beta_j}} \leq \epsilon$$

  and hence the play satisfies Equation 1.

- In the other case the play P can always reach a state s_2 such that σ_2 plays an action in $\mathrm{Bad}_2(s_2, X^*, Y_{\ell-j})$ and we can therefore enter a state in $s_3 \in Y_{\ell-j}$ with probability at least $\kappa_j = \frac{\beta_j \cdot \delta_{\min}}{m}$ from s_2. Since κ_j is a lower bound on the smallest positive probability in the Markov chain, any state that can be reached is actually reached within at most n steps with probability at least κ_j^n. In expectation we will need p^{-1} trails before an event that happens with probability $p > 0$ happens. We therefore needs κ_j^{-n} trails, each using n steps for a total of

$$n \cdot \kappa_j^{-n}$$

  steps. Therefore we need at most

$$\begin{aligned} n \cdot \kappa_j^{-n} &= n \cdot \frac{m^n}{\beta_j^n \cdot \delta_{\min}^n} \\ &= n \cdot m^n \cdot \delta_{\min}^{-n} \cdot n^{n \cdot \frac{n^{j-1}-1}{n-1}} \cdot \left(\frac{m}{\delta_{\min}}\right)^{n \cdot (\frac{n^j-1}{n-1} - 1)} \cdot \epsilon^{-n \cdot \frac{n^j-1}{n-1}} \\ &= \frac{\epsilon}{\beta_{j+1}} \end{aligned}$$

  steps in expectation to reach $Y_{\ell-j}$.

Note that if Equation 1 is not satisfied and the condition (1) (that the set $Y_{\ell-j}$ is reached after at most $\frac{\epsilon}{\beta_{j+1}}$ steps in expectation) is satisfied, then it implies that $Y_{\ell-j}$ is reached eventually with probability 1. Hence by induction it follows that either Equation 1 is satisfied by $\sigma_1^{\epsilon,\ell}$ or Y_0 is reached eventually with probability 1. Since Y_0 is the empty set, if player 1 plays $\sigma_1^{\epsilon,\ell} = \sigma_1^\epsilon$ and player 2 plays any positional strategy σ_2, then Equation 1 must be satisfied, i.e., for all $s \in X^*$, for all positional strategies of player 2

we have $\Pr_s^{\sigma_1^\epsilon,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon)) = 1$. But since σ_1^ϵ is stationary, as already mentioned, the mean-payoff objective is minimized by a positional strategy for player 2. Hence, it follows that for all $s \in X^*$ and all strategies for player 2 (not necessarily positional) we have $\Pr_s^{\sigma_1^\epsilon,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon)) = 1$. Since safety is already ensured by σ_1^ϵ, it follows that for all $s \in X^*$ and all strategies for player 2 we have $\Pr_s^{\sigma_1^\epsilon,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon) \cap \mathsf{Safe}(X^*)) = 1$. □

Lemma 2. *Let U be a set of states such that for all $\epsilon > 0$ there exists a stationary strategy σ_1^ϵ that against all strategies σ_2 and all $s \in U$, ensures*

$$\Pr_s^{\sigma_1^\epsilon,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon) \cap \mathsf{Safe}(U)) = 1.$$

Then there exists a Markov strategy σ_1^* for player 1, such that for all strategies σ_2 and all $s \in U$ we have

$$\Pr_s^{\sigma_1^*,\sigma_2}(\mathsf{LimInfAvg}(1)) = 1.$$

Proof. The construction of the desired strategy σ_1^* is as follows: consider the sequence $\epsilon_1, \epsilon_2, \ldots$ such that $\epsilon_1 = \frac{1}{4}$ and $\epsilon_{i+1} = \frac{\epsilon_i}{2}$. At any point, the strategy σ_1^* will play according to $\sigma_1^{\epsilon_i}$ for some $i \geq 1$. Initially the strategy plays as $\sigma_1^{\epsilon_1}$. The strategy $\sigma_1^{\epsilon_i}$ ensures that against any strategy σ_2 and starting in any state $s \in U$ we get that $\Pr_s^{\sigma_1^{\epsilon_i},\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon_i)) = 1$. Hence after a finite number of steps (that can be upper bounded with a bound J_i) with probability 1, the average-payoff is at least $1 - 2 \cdot \epsilon_i$ against any counter-strategy of player 2; and the safety objective ensures that the set U is never left. The bound J_i can be pre-computed: an easy description of the computation of the bound J_i is through value-iteration (on the player-2 MDP obtained by fixing the stationary strategy $\sigma_1^{\epsilon_i}$ as the strategy for player 1), and playing the game for a finite number of steps that ensure limit-average $1 - 2 \cdot \epsilon_i$ with probability 1, and then use the finite number as the bound J_i. Once the payoff is at least $1 - 2 \cdot \epsilon_i$ the strategy switches to the strategy $\sigma_1^{\epsilon_{i+1}}$ for J_{i+1} steps. As the length of the play goes to ∞, for all $\epsilon > 0$, for all $s \in U$ and all strategies σ_2 we have $\Pr_s^{\sigma_1^*,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon)) = 1$, and since this holds for all $\epsilon > 0$, we have $\Pr_s^{\sigma_1^*,\sigma_2}(\mathsf{LimInfAvg}(1)) = 1$. Using the bounds on the sequence $(J_i)_{i\geq 1}$ for the number of steps required before we switch strategies in the sequence of strategies $(\sigma_1^{\epsilon_i})_{i\geq 1}$, we obtain that the strategy σ_1^* is a Markov strategy. The desired result follows. □

Lemma 1 and Lemma 2 establishes one required inclusion (Lemma 3), and we establish the other inclusion in Lemma 4.

Lemma 3. *We have $X^* \subseteq \mathsf{Almost}_1(\mathsf{LimInfAvg}(1))$.*

Lemma 4. *We have*

$$\overline{X}^* = S \setminus X^* \subseteq \{s \in S \mid \exists \sigma_2^* \forall \sigma_1.\, \Pr_s^{\sigma_1,\sigma_2^*}(\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)) > 0\},$$

where $c = (\frac{\delta_{\min}}{m})^{n-1} \cdot \frac{1}{m}$. Moreover, there exist witness Markov strategies σ_2^* for player 2 to ensure $\overline{\mathsf{LimSupAvg}}_{\leq}(1-c) > 0$ from $\overline{X}^*$.

Proof. We will construct a Markov strategy σ_2 for player 2, such that the limit supremum average reward is at most $1 - c$ with positive probability for plays that start in a state in $\overline{X}^*$. This implies that we have $\overline{X}^* \subseteq \{s \in S \mid \exists \sigma_2^* \forall \sigma_1.\, \Pr_s^{\sigma_1,\sigma_2^*}(\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)) > 0\}$. We first consider the computation of X^*. Let $X_0 = S$ and $X_i = \mu Y.\nu Z.\mathrm{ASP}(X_{i-1}, Y, Z)$, for $i \geq 1$. Thus we will obtain a sequence $X_0 \supset X_1 \supset$

$X_2 \cdots \supset X_{k-1} \supset X_k = X_{k+1} = X^*$. For a set U of states, let us denote by $\overline{U} = S \setminus U$ the complement of the set U. We will construct a spoiling strategy σ_2^* for player 2 as the end of a sequence of strategies, $\sigma_2^1, \sigma_2^2, \ldots, \sigma_2^k = \sigma_2^*$, where $k \leq n$. The strategy σ_2^j will be constructed such that for all strategies σ_1 for player 1, for all $0 \leq j \leq k$, and for all $s \in \overline{X}_j$, we have

1. *(Property 1).* Either $\Pr_s^{\sigma_1,\sigma_2^j}(\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)) > 0$; or

2. *(Property 2).* $\Pr_s^{\sigma_1,\sigma_2^j}(\mathsf{Reach}(\overline{X}_{j-1})) > 0$ (recall that $\overline{X}_j = S \setminus X_j$).

The proof of the result will be by induction on j, and intuitively the correctness of the strategy construction of σ_2^j will use the nested iteration of the μ-calculus formula for X^*.

We assume that $X^* \neq S$, because if $S = X^* \subseteq \mathsf{Almost}_1(\mathsf{LimSupAvg}(1))$, then $S \setminus X^*$ is the empty set and we are trivially done.

Construction of σ_2^1. We first describe the details of σ_2^1 as the later strategies will be constructed similarly. Since $X^* \neq S$, we have $\overline{X}_1 = S \setminus X_1$ is non-empty. We first show that for all $s \in \overline{X}_1$ we have that $\Gamma_2(s) \setminus \mathrm{Bad}_2(s, S, X_1)$ is non-empty; otherwise if $\Gamma_2(s) \setminus \mathrm{Bad}_2(s, S, X_1)$ is empty, then $\mathrm{Good}_1(s, S, X_1, X_1)$ is the whole set $\Gamma_1(s)$ of actions, which implies $s \in \mathrm{ASP}(S, X_1, X_1) = X_1$ (leading to a contradiction that $s \in \overline{X}_1$). The description of the strategy σ_2^1 is as follows: for all $s \in \overline{X}_1$ the strategy plays all actions in $\Gamma_2(s) \setminus \mathrm{Bad}_2(s, S, X_1)$ uniformly at random; and for s not in $\overline{X}_1$, the strategy $\sigma_2^1(s)$ is an arbitrary probability distribution over $\Gamma_2(s)$. To prove the correctness of the construction of σ_2^1 we analyse the computation of the set X_1 as follows: the set X_1 is obtained as a sequence $Z_1^0 \supset Z_1^1 \supset Z_1^2 \supset \cdots \supset Z_1^\ell = Z_1^{\ell+1} = X_1$ where $Z_1^0 = S$ and $Z_1^{i+1} = \mathrm{ASP}(S, X_1, Z_1^i)$.

1. We first show that for all states s in $\overline{Z}_1^1 = S \setminus Z_1^1$, we have: (1) the next state is in the set $\overline{X}_1$ with probability 1, and (2) the probability that the reward is 0 in one step from s is at least $\frac{1}{m}$. We know that the set $\mathrm{Good}_1(s, S, X_1, Z_1^0) = \mathrm{Good}_1(s, S, X_1, S) = \emptyset$. Moreover, $\mathrm{Allow}_1(s, S)$ is the set of all player 1 actions $\Gamma_1(s)$. First, for every action a of player 1, for all actions $b \in \Gamma_2(s) \setminus \mathrm{Bad}_2(s, S, X_1)$ we have $\mathrm{Dest}(s, a, b) \subseteq \overline{X}_1$, and hence it follows that the set $\overline{X}_1$ is never left (i.e., the next state is always in X_1 with probability 1). Second, since $\mathrm{Good}_1(s, S, X_1, S) = \emptyset$, for every action a of player 1, there exists an action $b \in \Gamma_2(s) \setminus \mathrm{Bad}_2(s, S, X_1)$ such that $\mathrm{r}(s, a, b) = 0$ (note that for all actions a and b the condition $\mathrm{Dest}(s, a, b) \subseteq Z_1^0 = S$ is trivially satisfied, and hence the reward must be 0 to show that the action does not belong to the good set of actions). Since all actions in $\Gamma_2(s) \setminus \mathrm{Bad}_2(s, S, X_1)$ are played uniformly at random for every action a for player 1 the probability that the reward is 0 in one step is at least $\frac{1}{m}$.

2. For $i > 1$ we show that for all states s in $\overline{Z}_1^i = S \setminus Z_1^i$, we have: (1) the next state is in the set $\overline{X}_1$ with probability 1; and (2) either (i) the probability to reach the set $\overline{Z}_1^{i-1}$ in one step from s is at least $\frac{\delta_{\min}}{m}$ or (ii) the probability to get reward 0 in one step from s is at least $\frac{1}{m}$. As in the previous case since $\mathrm{Allow}_1(s, S)$ is the set of all actions $\Gamma_1(s)$, it follows from the same argument as above that the next state is in $\overline{X}_1$ with probability 1. We now focus on the second part of the claim. We know that $\mathrm{Good}_1(s, S, X_1, Z_1^{i-1})$ is empty. Hence for all actions a for player 1 there exists an action $b \in \Gamma_2(s) \setminus \mathrm{Bad}_2(s, S, X_1)$ such that either (i) $\mathrm{r}(s, a, b) = 0$, or (ii) $\mathrm{Dest}(s, a, b) \cap \overline{Z}_1^{i-1} \neq \emptyset$ (i.e., $\mathrm{Dest}(s, a, b) \not\subseteq Z_1^{i-1}$). It follows that either the reward is 0 with probability at least $\frac{1}{m}$ in one step from s or the set $\overline{Z}_1^{i-1}$ is reached with probability at least $\frac{\delta_{\min}}{m}$ in one step from s.

It follows from above that from any state in $S \setminus X_1$, there is a path of length at most n such that each step occurs with probability atleast $\frac{\delta_{\min}}{m}$ (except for the last which occurs with probability at least $\frac{1}{m}$) and the last reward is 0, and the path always stays in $S \setminus X_1$, given player 2 plays the strategy σ_2^1, irrespective of the strategy of player 1. Hence for all $s \in \overline{X}_1 = S \setminus X_1$ and for all strategies σ_1 we have $\Pr_s^{\sigma_1,\sigma_2^1}(\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)) = 1$. We present a remark about the above construction as it will be used later.

Remark 5. *Let $X_1 = \mu Y.\nu Z.\text{ASP}(S, Y, Z)$, and $\overline{X}_1 = S \setminus X_1$. Then there exists a stationary strategy σ_2^1 with patience at most m for player 2 such that for all strategies σ_1 for player 1 we have $\Pr_s^{\sigma_1,\sigma_2^1}(\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)) = 1$, for all $s \in \overline{X}_1$.*

We now describe the inductive construction of the strategy σ_2^j from σ_2^{j-1}, for $j \geq 2$. Let $0 < \epsilon < 1$ be given. For plays which are in state $s \notin X_{j-1} \setminus X_j$, the strategy σ_2^j follows σ_2^{j-1}. If the play is in state $s \in X_{j-1} \setminus X_j$ in round i the strategy σ_2^j uses a binary random variable B^i (where B^i is independent of B^ℓ for $\ell \neq i$) which is 1 with probability $\frac{\epsilon}{2^i}$ and 0 otherwise. If B^i is 1, then σ_2^j chooses an action uniformly at random from $\Gamma_2(s)$, otherwise it chooses an action uniformly at random from $\Gamma_2(s) \setminus \text{Bad}_2(s, X_{j-1}, X_j)$. Notice the fact that B^i is independent of B^ℓ, for $\ell \neq i$, ensures that σ_2^j is a Markov strategy. We show that $\Gamma_2(s) \setminus \text{Bad}_2(s, X_{j-1}, X_j)$ is non-empty. If $\text{Allow}_1(s, X_{j-1})$ is empty, then $\text{Bad}_2(s, X_{j-1}, X_j)$ is empty and therefore $\Gamma_2(s) \setminus \text{Bad}_2(s, X_{j-1}, X_j)$ is non-empty. Otherwise, if $\text{Allow}_1(s, X_{j-1})$ is non-empty, we can use that $\text{Good}_1(s, X_{j-1}, X_j, X_j)$ is empty, because $s \in X_{j-1} \setminus X_j$. But by definition of $\text{Good}_1(s, X_{j-1}, X_j, X_j)$ this implies that $\Gamma_2(s) \setminus \text{Bad}_2(s, X_{j-1}, X_j)$ is non-empty.

Consider a counter-strategy σ_1 for player 1. In round ℓ, if the play is in a state s in $X_{j-1} \setminus X_j$ and the conditional probability that σ_1 chooses an action a with positive probability which is not in $\text{Allow}_1(s, X_{j-1})$, then there is an action b such that $\text{Dest}(s, a, b) \cap \overline{X}_{j-1} \neq \emptyset$. Hence since σ_2^j plays all actions with positive probability we see that such plays reaches $S \setminus X_{j-1} = \overline{X}_{j-1}$ with positive probability (in this case the desired Property 2 holds). Therefore, we consider the case such that σ_1 only plays action with positive probability that are in $\text{Allow}_1(s, X_{j-1})$. With probability at least $1 - \sum_{i=1}^{\infty} \frac{\epsilon}{2^i} = 1 - \epsilon > 0$, we have that $B^i = 0$ for all i. If B^i is 0 for all i, the proof proceeds like in the base case (correctness proof for σ_2^1), except that we view X_{j-1} as the set of all states (note that no state outside X_{j-1} can be reached because σ_1 only plays actions in $\text{Allow}_1(s, X_{j-1})$ and that $B^i = 0$ for all i). In this scenario, as in the proof of the base case, we have that the strategy σ_2^j ensures that all plays starting in states $s \in \overline{X}_j \setminus \overline{X}_{j-1}$ do not leave the set $\overline{X}_j \setminus \overline{X}_{j-1}$, and the probability that the objective $\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)$ is satisfied is strictly greater than 0 for all strategies of player 1. This establishes by induction the desired properties 1 and 2. Since $\overline{X}_0$ is empty, it follows that for all states $s \in \overline{X}^*$ and any strategy σ_1 for player 1, we have $\Pr_s^{\sigma_1,\sigma_2^*}(\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)) > 0$ (as in the proof of Lemma 1). Notice that since σ_2^j is a Markov strategy for all j, it follows that σ_2^* is also a Markov strategy. The desired result is established. □

Theorem 6 (Qualitative determinacy and polynomial time computability)**.** *The following assertions hold for all concurrent game structures with boolean rewards:*

1. *We have*

$$\begin{aligned} X^* &= \mathsf{Almost}_1(\mathsf{LimInfAvg}(1)) = \mathsf{Almost}_1(\mathsf{LimSupAvg}(1)) \\ &= \bigcap_{\varepsilon>0} \mathsf{Almost}_1(\mathsf{LimInfAvg}(1-\varepsilon)) = \bigcap_{\varepsilon>0} \mathsf{Almost}_1(\mathsf{LimSupAvg}(1-\varepsilon)); \end{aligned}$$

and

$$\begin{aligned} S \setminus X^* &= \mathsf{Positive}_2(\overline{\mathsf{LimInfAvg}}(1)) = \mathsf{Positive}_2(\overline{\mathsf{LimSupAvg}}(1)) \\ &= \bigcup_{c>0} \mathsf{Positive}_2(\overline{\mathsf{LimInfAvg}}_{\leq}(1-c)) = \bigcup_{c>0} \mathsf{Positive}_2(\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)); \end{aligned}$$

where $X^* = \nu X.\mu Y.\nu Z.\text{ASP}(X, Y, Z)$.

2. *The set* X^* *can be computed in cubic time (in time* $O(n^2 \cdot |\delta|)$*, where* $|\delta| = \sum_{s \in S} \sum_{a \in \Gamma_1(s)} \sum_{b \in \Gamma_2(s)} |\text{Dest}(s, a, b)|$*) by straight-forward computation of the* μ*-calculus formula* $\nu X.\mu Y.\nu Z.\text{ASP}(X, Y, Z)$.

Proof. Trivially we have $\mathsf{Almost}_1(\mathsf{LimInfAvg}(1)) \subseteq \bigcap_{\varepsilon>0} \mathsf{Almost}_1(\mathsf{LimInfAvg}(1-\varepsilon))$ and $\bigcup_{c>0} \mathsf{Positive}_2(\overline{\mathsf{LimInfAvg}}(1-c)) \subseteq \mathsf{Positive}_2(\overline{\mathsf{LimInfAvg}}(1))$ (also similarly for LimSupAvg). By Lemma 3 we have $X^* \subseteq \mathsf{Almost}_1(\mathsf{LimInfAvg}(1))$ and by Lemma 4 we have $S \setminus X^* \subseteq \bigcup_{c>0} \mathsf{Positive}_2(\overline{\mathsf{LimSupAvg}}_{\leq}(1-c))$. Also observe that trivially we have $\bigcup_{c>0} \mathsf{Positive}_2(\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)) = \bigcup_{c>0} \mathsf{Positive}_2(\overline{\mathsf{LimSupAvg}}(1-c))$ (and similarly for LimInfAvg). Thus we obtain all the desired equalities. The second item trivially follows as the μ-calculus defines a nested iterative algorithm. □

3.2 Strategy Complexity

In this section we will establish the complexities of the witness almost-sure and positive winning strategies for player 1 and player 2, from their respective winning sets. We start with a lemma that shows a lower bound on the time-dependent memory of infinite-memory strategies. The authors would like to thank Kristoffer Arnsfelt Hansen for the proof of the following lemma.

Lemma 7. *In a repeated game with absorbing states, if a strategy requires infinite memory, then more than* T *memory states are required by the strategy for the first* T *rounds, for all* $T > 0$*, i.e., the size of the time-dependent memory is at least* T*, for all* $T > 0$*.*

Proof. Let σ be a strategy that requires infinite memory. Consider the directed graph where the states are the memory states of σ and where there is an edge from state m to state m', if there exists an action a consistent with σ and an action b for the other player, such that $\sigma^u(a, b, m) = m'$. Since the set of actions for each player is finite, the out-degree of all states are finite.

We have by definition of σ that the graph is infinite and we can reach infinitely many memory states from the start state. In a graph where each state has finite out-degree there are two possibilities. Either it is possible to reach a state from the start state in T steps that is not reachable in $T-1$ steps, for each $T > 0$; or only a finite number of states can be reached from the start state. Since we can reach an infinite number of states from the start state we must be in the first case. Therefore at least T memory states can be reached from the start state in T steps, for all $T > 0$. □

Recall that a Markov strategy is an infinite-memory strategy with time-dependent memory of size T, for all $T > 0$. In view of Lemma 7 it follows that if infinite-memory requirement is established for repeated games with absorbing states, then time-dependent memory bound of Markov strategies match the lower bound of the time-dependent memory.

Infinite-memory for almost-sure winning strategies. In case of concurrent reachability games, stationary almost-sure winning strategies exist. In contrast we show that for concurrent games with boolean reward functions, almost-sure winning strategies for exact qualitative constraint require infinite memory.

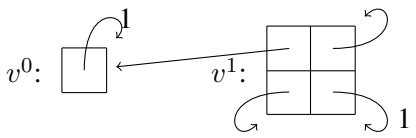

Figure 2: The example illustrates G^1 where all states are in $\mathsf{Almost}_1(\mathsf{LimInfAvg}(1))$, but no finite-memory almost-sure winning strategy exists for player 1 for the objective $\mathsf{LimInfAvg}(1)$. All transitions with reward different from 0 (i.e., reward 1) have the reward annotated on the transition.

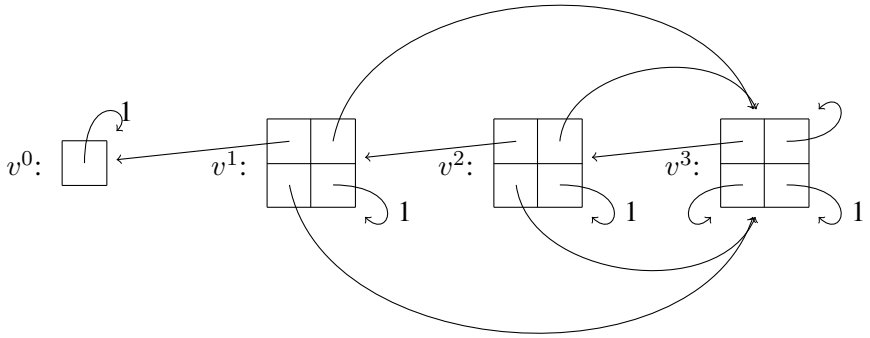

Figure 3: The example illustrates G^3 of the family $\{G^n\}$ where all states are in $\bigcap_{\epsilon>0} \mathsf{Almost}_1(\mathsf{LimInfAvg}(1-\epsilon))$ but all witness stationary strategies that ensure so for player 1 require patience at least double exponential in n. All transitions with reward different from 0 (i.e., reward 1) have the reward annotated on the transition.

Game family. Let G^n be the following game. The game G^n has $n+1$ states, namely, v^0, v^1, $\dots$ v^n. The state v^0 is absorbing with reward 1. For $\ell \geq 1$, the state v^ℓ has two actions for both players. The actions are a_1^ℓ and a_2^ℓ for player 1 and b_1^ℓ and b_2^ℓ for player 2, respectively. Also $\mathrm{r}(v^\ell, a_i^\ell, b_j^\ell) = 0$ except for $i = j = 2$, for which $\mathrm{r}(v^\ell, a_2^\ell, b_2^\ell) = 1$. Furthermore, (i) $\delta(v^\ell, a_i^\ell, b_j^\ell) = v_n$ for $i \neq j$; (ii) $\delta(v^\ell, a_1^\ell, b_1^\ell) = v_{\ell-1}$; and (iii) $\delta(v^\ell, a_2^\ell, b_2^\ell) = v_\ell$. There is an illustration of G^1 in Figure 2 and an illustration of G^3 in Figure 3. We first show that all states are in X^*: in G^n, if we consider X^* to be the set of all states and evaluate $\mu Y.\nu Z.\mathrm{ASP}(X^*, Y, Z)$, then we obtain that $Y_0 = \emptyset$, and for $i \geq 0$ we have $Y_{i+1} = \{v^0, v^1, \dots, v^i\}$ because $\mathrm{Allow}_1(v^j, X^*)$ is the set of all actions available for player 1 at state v^j, for all $1 \leq j \leq n$, and $\mathrm{Bad}_2(v^i, X^*, Y_i) = b_1^i$ and $\mathrm{Good}_1(v^i, X^*, Y_i, Y_{i+1}) = a_2^i$. Thus it follows that X^* is the set of all states, i.e., all states belong to $\mathsf{Almost}_1(\mathsf{LimInfAvg}(1))$.

Lemma 8. *All almost-sure winning strategies for player 1 in the game G^1 require infinite memory for the objective* $\mathsf{LimInfAvg}(1)$.

Proof. The proof will be by contradiction. Assume towards contradiction that there is a strategy σ_1 that uses only a finite number of memory states and is almost-sure winning for the objective $\mathsf{LimInfAvg}(1)$. Let the smallest non-zero probability the strategy σ_1 plays a_1^1 in any memory state be p. We will show that there is a strategy σ_2 for player 2 that ensure

$$\Pr_{v^1}^{\sigma_1,\sigma_2}(\overline{\mathsf{LimSupAvg}}_{\leq}(1-p)) = 1.$$

The strategy σ_2 for player 2 is to play b_1^1 (in v^1) if given the play so far, the strategy σ_1 is in a memory state where a_2^1 is played with probability 1. Otherwise player 2 plays b_2^1 (in v^1). Hence, the probability to reach v^0 from v^1 is 0. But the probability that a_2^1 is played at the same time as b_1^1 in v^1 is then at most $1-p$ in any round. Thus we have $\Pr_{v^1}^{\sigma_1,\sigma_2}(\overline{\mathsf{LimSupAvg}}_{\leq}(1-p)) = 1$ contradicting that σ_1 is an almost-sure winning strategy for the objective $\mathsf{LimInfAvg}(1)$. It follows that every almost-sure winning strategy for player 1 requires infinite memory for the objective $\mathsf{LimInfAvg}(1)$ (note that since all states are in X^* in G^1 it follows that almost-sure winning strategies exist for player 1). □

Double exponential lower bound for patience. We have already established in the previous section (Lemma 1 and Theorem 6) that for all $\epsilon > 0$ stationary almost-sure winning strategies exist with at most double exponential patience for objectives $\mathsf{LimInfAvg}(1-\epsilon)$, for all states in X^*. We now establish a double exponential lower bound on patience.

Lemma 9. *Let n be given. Given $0 < \epsilon \leq \frac{1}{3}$, let σ_1 be a stationary strategy for player 1 that achieves*

$$\forall v \in X^* \forall \sigma_2 : \Pr_v^{\sigma_1,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon)) = 1, \tag{2}$$

in G^n. Then σ_1 has patience at least $\epsilon^{-1.5^{n-1}}$.

Proof. Let n be given. Let σ_1 be any stationary strategy that satisfies the condition of the lemma (i.e., Equation 2). Let $x_i = \sigma_1(v^{n-i})(a_1^{n-i})$. First notice that $x_i > 0$, otherwise, consider a stationary strategy σ_2 for player 2 such that $\sigma_2(v^{n-i})(b_1^{n-i}) = 1$, which ensures that all payoffs of any play starting in v^{n-i} would be 0. We will now show that $x_i \leq \epsilon^{1.5^i}$ for $i < n$. The proof will be by induction on i. The proof will use two base cases $i = 0$ and $i = 1$, because the inductive proof then becomes simpler.

First base case. First the base case $i = 0$. We have that $x_0 \leq \epsilon$, because if σ_2 is a stationary strategy such that $\sigma_2(v_n)(b_2^n) = 1$, then $\sigma_1(v_n)(a_2^n) \geq 1 - \epsilon$ because it must satisfy the Equation 2. Since Equation 2 is satisfied for all σ_2 we have that $0 < x_0 \leq \epsilon$ as desired.

Second base case. The second base case is for $i = 1$. Let P be a play starting in v^{n-1}. If σ_2 is a stationary strategy such that $\sigma_2(v^{n-1})(b_2^n) = 1$ and $\sigma_2(v_n)(b_1^n) = 1$, then any time there is a reward of 1, the play must be in state v^{n-1}. But whenever v^n is reached we expect at least ϵ^{-1} time steps with reward 0, before the play reaches v^{n-1}. Therefore x_1 must be such that $\frac{\epsilon^{-1} \cdot x_1}{1 - x_1 + \epsilon^{-1} \cdot x_1} \leq \epsilon$ because it must satisfy the Equation 2. Hence, we have that

$$\begin{aligned}
\frac{\epsilon^{-1} \cdot x_1}{1 - x_1 + \epsilon^{-1} \cdot x_1} &\leq \epsilon \Rightarrow \\
\frac{\epsilon^{-1}}{x_1^{-1} - 1 + \epsilon^{-1}} &\leq \epsilon \Rightarrow \\
\epsilon^{-1} \leq \epsilon \cdot (x_1^{-1} - 1 + \epsilon^{-1}) &\Rightarrow \\
\epsilon^{-2} \leq x_1^{-1} - 1 + \epsilon^{-1} &\Rightarrow \\
\epsilon^{-1.5} \leq x_1^{-1} \ , &
\end{aligned}$$

where the last implication is because $\epsilon \leq \frac{1}{3}$. Since Equation 2 is satisfied for all σ_2 we have that $0 < x_1 \leq \epsilon^{1.5}$ as desired.

Inductive case. We now consider the inductive case for $i > 1$. The proof is similar to the base cases, especially the second. If σ_2 is a stationary strategy such that both $\sigma_2(v^n)(b_2^{n-i}) = 1$ and $\sigma_2(v_n)(b_1^{n-j}) = 1$ for $j < i$, then for any play starting in v^{n-i} can only get a reward of 1 in v^{n-i}. But by induction $\prod_{j=0}^{i-1} x_j \leq \prod_{j=0}^{i-1} \epsilon^{1.5^j} = \epsilon^{\sum_{j=0}^{i-1} 1.5^j} = \alpha_i$. That implies that more than α_i^{-1} steps are needed to reach v^{n-i} from v^n (because clearly the play must pass through state v_{n-j} for $j \leq i$). Hence whenever the play is in v^{n-i}, there is a reward of 1 with probability $1 - x_i$ and a reward of 0 for more than α_i^{-1} time steps with probability x_i. Hence x_i must be such that

$$\begin{aligned}
\frac{\alpha_i^{-1} \cdot x_i}{1 - x_i + \alpha_i^{-1} \cdot x_i} &\leq \epsilon \Rightarrow \\
\frac{\alpha_i^{-1}}{x_i^{-1} - 1 + \alpha_i^{-1}} &\leq \epsilon \Rightarrow \\
\alpha_i^{-1} \leq \epsilon \cdot (x_i^{-1} - 1 + \alpha_i^{-1}) &\Rightarrow \\
\frac{\alpha_i^{-1}}{\epsilon} \leq x_i^{-1} - 1 + \alpha_i^{-1} &\Rightarrow \\
\frac{\alpha_i^{-1}}{\epsilon} + 1 - \alpha_i^{-1} \leq x_i^{-1} &\Rightarrow \\
\left(\frac{1}{\epsilon} - 1\right) \cdot \alpha_i^{-1} \leq x_i^{-1} &\Rightarrow \\
\alpha_i^{-1} \leq x_i^{-1} \ , &
\end{aligned}$$

where the last implication comes from the fact that $\epsilon \leq \frac{1}{3} \leq \frac{1}{2}$. But since $\sum_{j=0}^{i-1} 1.5^j > 1.5^i$ for $i > 1$, the result follows. □

Theorem 10 (Strategy complexity)**.** *For concurrent games with boolean reward functions the following assertions hold:*

1. *Almost-sure winning strategies for objectives* $\mathsf{LimInfAvg}(1)$ *(and* $\mathsf{LimSupAvg}(1)$*) for player 1 require infinite-memory in general; whenever there exists an almost-sure winning strategy for objectives* $\mathsf{LimInfAvg}(1)$ *(and* $\mathsf{LimSupAvg}(1)$*), then a Markov almost-sure winning strategy exists; and the optimal bound for time-dependent memory is* T*, for all rounds* $T > 0$*.*

2. *For all* $\epsilon > 0$*, stationary almost-sure winning strategies exist for player 1 for objectives* $\mathsf{LimInfAvg}(1-\epsilon)$ *(and* $\mathsf{LimSupAvg}(1-\epsilon)$*); and the asymptotically optimal bound for patience for such stationary almost-sure winning strategies is double exponential in the size of the state space.*

3. *Positive winning strategies for player 2 for objectives* $\overline{\mathsf{LimInfAvg}}(1)$ *and* $\overline{\mathsf{LimInfAvg}}_{\leq}(1-c)$*, for some constant* $c > 0$*, (also* $\overline{\mathsf{LimSupAvg}}(1)$ *and* $\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)$*, for some constant* $c > 0$*) require infinite-memory in general; whenever such positive winning strategies exist, Markov strategies are sufficient and the optimal bound for time-dependent memory is* T*, for all rounds* $T > 0$*.*

Proof. The proofs are as follows:

1. Lemma 8 shows that infinite-memory is required, and Lemma 1, Lemma 2, and Theorem 6 show that Markov strategies are sufficient for almost-sure winning. The sufficiency of Markov strategies establishes the T upper bound for time-dependent memory; and Lemma 8 (along with the fact that the game in the lemma is a repeated game with absorbing states) and Lemma 7 establishes the T lower bound for time-dependent memory.

2. The existence of stationary almost-sure winning strategies with double exponential patience for objectives $\mathsf{LimInfAvg}(1-\epsilon)$, for all $\epsilon > 0$ follows from Lemma 1 and Theorem 6. The double exponential lower bound for patience follows from Lemma 9.

3. For the special case of concurrent reachability and safety games, $\overline{\mathsf{LimInfAvg}}(1)$ and $\overline{\mathsf{LimInfAvg}}_{\leq}(1-c)$, for some constant $c > 0$, coincide, and the infinite-memory requirement for player 2 for positive winning strategies follows from [14]. Moreover the example to show the infinite-memory requirement (from [14]) is a repeated game with absorbing states. The sufficiency of Markov strategies follows from Lemma 7; and the optimal time-dependent memory bound of T follows from the sufficiency of Markov strategies (upper bound) and Lemma 7 and the infinite-memory requirement (lower bound).

The desired result follows. □

3.3 Improved Algorithm

In this section we will present an improved algorithm for the computation of the almost-sure winning set $\mathsf{Almost}_1(\mathsf{LimInfAvg}(1))$. The naive computation using the μ-calculus formula gives a cubic time complexity, and we will present an alternative quadratic time algorithm. The key idea is to generalize the small-progress measure algorithm of [25] with the more involved predecessor operator.

The key intuition. The key intuition of the algorithm is to assign to each state s a *level*, denoted $\ell(s)$, which range in the set $\{0, 1, \ldots, n\}$. The level is like a ranking function and the algorithm iteratively updates the level of every state. The initial level of all states are n, and the levels can only decrease during the execution of the algorithm. At the end of the execution of the algorithm, the set X^* will be exactly the set of states which have a strictly positive level. The total change of levels is at most quadratic and by charging the work done to the change of the levels we show that the work done is also at most quadratic.

Basic procedures. The algorithm will consist of two procedures, namely, Process(s) and Remove(s, b), for $s \in S$ and $b \in \Gamma_2(s)$. To describe the procedures we first define three action sets as follows: $\text{Allow}_1(s) \subseteq \Gamma_1(s)$, $\text{Bad}_2(s) \subseteq \Gamma_2(s)$ and $\text{Good}_1(s) \subseteq \Gamma_1(s)$. The sets will have similar intuitive meaning as the corresponding set in the μ-calculus expression. In the algorithm, whenever the set $\text{Good}_1(s)$ becomes empty, the level $\ell(s)$ of s will be decreased by one. For a fixed level of all the states, the sets are as follows:

- $\text{Allow}_1(s)$ is the set of all actions $a \in \Gamma_1(s)$ such that for all actions $b \in \Gamma_2(s)$ we have $\text{Dest}(s, a, b) \cap L_0 = \emptyset$, where L_0 is the set of states with level 0.
- $\text{Bad}_2(s)$ is the set of all actions $b \in \Gamma_2(s)$ such that there exists $a \in \text{Allow}_1(s)$ and $t \in S$ such that $t \in \text{Dest}(s, a, b)$ and $\ell(t) > \ell(s)$.
- $\text{Good}_1(s)$ is the set of all actions $a \in \text{Allow}_1(s)$ such that for all $b \in \Gamma_2(s) \setminus \text{Bad}_2(s)$ we $\text{r}(s, a, b) = 1$ and for all $t \in \text{Dest}(s, a, b)$ we have $\ell(t) \geq \ell(s)$.

For all $b \in \Gamma_2(s)$, the algorithm keeps track of the number of actions a in $\text{Allow}_1(s)$ and t in S, such that $t \in \text{Dest}(s, a, b)$ and $\ell(t) > \ell(s)$. We denote this number by $\text{b}(s, b)$. Observe that an action $b \in \Gamma_2(s)$ is in $\text{Bad}_2(s)$ if and only if $\text{b}(s, b) > 0$. We are now ready to describe the two basic procedures.

1. The procedure Process(s) recalculates the actions in $\text{Allow}_1(s)$, $\text{Bad}_2(s)$ and $\text{Good}_1(s)$, based on the current level of all states. It also recalculates $\text{b}(s, b)$. The running time of the procedure is $O(\sum_{a \in \Gamma_1(s)} \sum_{b \in \Gamma_2(s)} |\text{Dest}(s, a, b)|)$, by simple enumeration over the actions of both players and the possible successor given the actions. The procedure Process(s) will run (i) once for each time state s changes level; (ii) each time some state changes to level 0; and (iii) once during the initialization of the algorithm.
2. The procedure Remove(s, b) is run only when $\text{b}(s, b)$ is zero. The procedure Remove(s, b) removes b from $\text{Bad}_2(s)$ and for each action $a \in \text{Good}_1(s)$ checks if $\text{r}(s, a, b) = 0$. If so, it removes such a's from $\text{Good}_1(s)$. The running time of the procedure is $O(\sum_{a \in \Gamma_1(s)} |\text{Dest}(s, a, b)|)$ (again by simple enumeration). It follows from the description of $\text{b}(s, b)$ that as long as the level of the state s is fixed we only decrease $\text{b}(s, b)$. Hence we will run Remove(s, b) at most n times, once for each level of s.

The informal description of the algorithm. The informal description of the algorithm is as follows. In the initialization phase first all states s are assigned level $\ell(s) = n$, and then every state is processed using the procedure Process(s). The algorithm is an iterative one and in every iteration executes the following steps (unless a fixpoint is reached). It first considers the set of states s such that $\text{Good}_1(s)$ is empty and decrement the level of s. If the level of a state reaches 0, then a flag z is assigned to true. If z is true, then we process every state using the procedure Process. Otherwise, for every state s such that $\text{Good}_1(s)$ is empty, the algorithm processes s using Process(s); updates $\text{b}(t, b)$ for all predecessors t of s and removes an action when the $\text{b}(t, b)$ count reaches zero. The algorithm reaches a fixpoint when the level of no state has changed (the algorithm keeps track of this with a flag c). The algorithm outputs $\widetilde{X}^*$ which is the set of states s with strictly positive level (i.e., $\ell(s) > 0$ at the end of the execution). The formal description of the algorithm is presented in Figure 4, and we refer to the algorithm as IMPROVEDALGO. We first present the runtime analysis and then present the correctness argument.

Runtime analysis. As described above other than the initialization phase, whenever the procedure Process(s) is run, the level of the state s has decreased or the level of some other state has reached 0. Hence for every state s, the procedure can run at most $2 \cdot n$ times. Therefore the total running time for all Process operations over all iterations is $O(n \cdot |\delta|)$, where $|\delta| = \sum_{s \in S} \sum_{a \in \Gamma_1(s)} \sum_{b \in \Gamma_2(s)} |\text{Dest}(s, a, b)|$.

Algorithm 1: IMPROVEDALGO: Input: Concurrent game structure G with boolean reward function.

for $s \in S$ **do** $\ell(s) \leftarrow n$;
for $s \in S$ **do** Process(s);
$c \leftarrow true$;
while $c = true$ **do**
  $c \leftarrow false$; $z \leftarrow false$;
  for $s \in S$ *st.* $\ell(s) > 0$ *and* $\text{Good}_1(s) = \emptyset$ **do**
    $c \leftarrow true$; $\ell(s) \leftarrow \ell(s) - 1$; **if** $\ell(s) = 0$ **then** $z \leftarrow true$;
  if $z = true$ **then**
    for $s \in S$ **do** Process(s);
  else
    for $s \in S$ *st.* $\ell(s) > 0$ *and* $\text{Good}_1(s) = \emptyset$ **do**
      Process(s);
      for t, a, b *st.* $s \in \text{Dest}(t, a, b)$ *and* $\ell(t) = \ell(s)$ **do**
        $\text{b}(t, b) \leftarrow \text{b}(t, b) - 1$; **if** $\text{b}(t, b) = 0$ **then** Remove(t, b);
return $\widetilde{X}^* = \{s \mid \ell(s) > 0\}$;

Figure 4: Improved Algorithm

The procedure Remove(s, b) is invoked when $\text{b}(s, b)$ reaches zero, and as long as the level of s is fixed the count $\text{b}(s, b)$ can only decrease. This implies that we run Remove(s, b) at most n times, once for each level of s. Hence $O(n \cdot |\delta|)$ is the total running time of operation Remove over all iterations.[3]

Correctness analysis. We will now present the correctness analysis in the following lemma.

Lemma 11. *Given a concurrent game structure with a boolean reward function as input, let $\widetilde{X}^*$ be the output of algorithm* IMPROVEDALGO *(Figure 4). Then we have $\widetilde{X}^* = X^* = \mathsf{Almost}_1(\mathsf{LimInfAvg}(1))$.*

Proof. We first observe that for the algorithm IMPROVEDALGO at the end of any iteration, for all s the sets $\text{Allow}_1(s)$, $\text{Bad}_2(s)$ and $\text{Good}_1(s)$ are correctly calculated based on the current level of all states as defined by the description. Let us denote by $\ell^*(s)$ the level of a state s at the end of the execution of the algorithm. Recall that $\widetilde{X}^*$ is the set of states s with $\ell^*(s) > 0$. Also recall that $X^* = \nu X.\mu Y.\nu Z.\text{ASP}(X, Y, Z)$. The correctness proof will show two inclusions. We present them below.

- *First inclusion:* $\widetilde{X}^* \subseteq X^*$. We will show that $\widetilde{X}^*$ is a fixpoint of the function $f(X) = \mu Y.\nu Z.\text{ASP}(X, Y, Z)$. Let $\widetilde{Y}_0 = \emptyset$, and $\widetilde{Y}_i = \{s \mid \ell^*(s) > n - i\}$ for $0 < i < n$. Then for all $0 < i < n$ and for all $s \in \widetilde{Y}_i \setminus \widetilde{Y}_{i-1}$ we have $s \in \nu Z.\text{ASP}(\widetilde{X}^*, \widetilde{Y}_{i-1}, Z)$. The fact that $s \in \text{ASP}(\widetilde{X}^*, \widetilde{Y}_{i-1}, \widetilde{Y}_i)$ follows since: (i) $\text{Allow}_1(s)$ as computed by the algorithm is $\text{Allow}_1(s, \widetilde{X}^*)$; (ii) $\text{Bad}_2(s)$ as computed the algorithm is $\text{Bad}_2(s, \widetilde{X}^*, \widetilde{Y}_{i-1})$; and (iii) $\text{Good}_1(s)$ as computed by the algorithm is $\text{Good}_1(s, \widetilde{X}^*, \widetilde{Y}_{i-1}, \widetilde{Y}_i)$. Hence it follows that $\widetilde{X}^*$ is a fixpoint of $f(X) = \mu Y.\nu Z.\text{ASP}(X, Y, Z)$. Since X^* is the greatest fixpoint of $f(X)$ we have that $\widetilde{X}^* \subseteq X^*$.

[3]The running time assumes a data structure that for a given s can find the set $\mathsf{Pred}(s) = \{(t, a, b) \mid s \in \text{Dest}(t, a, b)\}$ of predecessors in time $O(|\mathsf{Pred}(s)|)$, which can be easily achieved with a linked list data structure.

- *Second inclusion:* $X^* \subseteq \widetilde{X}^*$. Let i and s be such that $s \in Y_i \setminus Y_{i-1}$, where $Y_0 = \emptyset$, and for $i > 0$ we have $Y_i = \nu Z.\text{ASP}(X^*, Y_{i-1}, Z)$. We will show that $i = n + 1 - \ell^*(s)$. That implies that $\ell^*(s) > 0$, because of the following: We have that s can be in $Y_j \setminus Y_{j-1}$ for at most one value of j, because $Y_{k-1} \subseteq Y_k$ for all k. Since $Y_j \setminus Y_{j-1}$ is non-empty for all $j > 0$ till the fixpoint is reached we have $X^* = Y_n$. Together that gives us that $\ell^*(s) > 0$.

  We will first show that $i \geq n + 1 - \ell^*(s)$. Assume towards contradiction that $\ell^*(s) < n + 1 - i$. Let k be the first iteration of the algorithm in which some state $t \in Y_j \setminus Y_{j-1}$ goes from level $n + 1 - j$ to level $n - j$ (this is well-defined because s must do so in some iteration by assumption). We can WLOG assume that s changes from level $n + 1 - i$ to $n - i$ in iteration k. But at the end of iteration $k - 1$, we then have that $\text{Allow}_1(s, X^*) \subseteq \text{Allow}_1(s)$ and therefore $\text{Bad}_2(s) \subseteq \text{Bad}_2(s, X^*, Y_{i-1})$ and therefore $\text{Good}_1(s, X^*, Y_{i-1}, Y_i) \subseteq \text{Good}_1(s)$, implying that $\text{Good}_1(s)$ cannot be empty. Hence s does not change level in iteration k. That is a contradiction.

  We will next show that $i \leq n + 1 - \ell^*(s)$. Assume towards contradiction that $\ell^*(s) > n + 1 - i$. Let ℓ be the highest level for which there is a state $t \in Y_j \setminus Y_{j-1}$ such that $\ell = \ell^*(t)$ and $\ell^*(t) > n + 1 - j$ (since $\ell^*(s) > n + 1 - i$ this is well defined). We can WLOG assume that $\ell^*(s) = \ell$. By the first part of this proof we have that $\widetilde{X}^* \subseteq X^*$, implying that $\text{Allow}_1(s) \subseteq \text{Allow}_1(s, X^*)$. By definition of ℓ, we then get that $\text{Bad}_2(s, X^*, Y_{\ell-1}) \subseteq \text{Bad}_2(s)$. Let U be the set of states, such that for all $t \in U$ we have that $\ell^*(t) = \ell$. Since for all $t \in Y_j \setminus Y_{j-1}$ we have that $j \geq n + 1 - \ell^*(t)$, we get that $Y_\ell \setminus Y_{\ell-1} \subset U$ (they are not equal since $Y_\ell \setminus Y_{\ell-1}$ does not contain s). We have that $\text{Good}_1(t)$ is non-empty for all $t \in U$. This implies that $U \subseteq T$, where T is a fixpoint of $\text{ASP}(X^*, Y_{\ell-1}, T)$. But $Y_\ell \subset T$ is the largest such fixpoint by definition. That is a contradiction.

The desired result follows. □

Theorem 12. *The algorithm* IMPROVEDALGO *correctly computes the set* $\mathsf{Almost}_1(\mathsf{LimInfAvg}(1))$ *for a concurrent game structure with boolean reward function in quadratic time (in time* $O(n \cdot |\delta|)$*, where* $|\delta| = \sum_{s \in S} \sum_{a \in \Gamma_1(s)} \sum_{b \in \Gamma_2(s)} |\text{Dest}(s, a, b)|$*).*

4 Positive Winning

In this section we will present qualitative determinacy for positive winning and then establish the strategy complexity results.

4.1 Qualitative determinacy

In this section we will present a polynomial time algorithm to compute the set $\mathsf{Positive}_1(\mathsf{LimInfAvg}(\lambda))$ and $\mathsf{Positive}_1(\mathsf{LimSupAvg}(\lambda))$ for $\lambda = 1$ in concurrent games with boolean reward functions, and the qualitative determinacy will also be a consequence of the algorithm. Again, like in Section 3, we will first present the algorithm as a μ-calculus expression. The algorithm is

$$\mathsf{Positive}_1(\mathsf{LimInfAvg}(1)) = \mathsf{Positive}_1(\mathsf{LimSupAvg}(1)) = \mu Y.\nu Z.\text{ASP}(S, Y, Z),$$

where $\text{ASP}(X, Y, Z)$ is as defined in Section 3. Let $Y^* = \mu Y.\nu Z.\text{ASP}(S, Y, Z)$ be the fixpoint.

Lemma 13. *There is a stationary strategy* σ_2 *for player 2 with patience at most* m *that ensures that for all states* $s \in S \setminus Y^*$*, all strategies* σ_1 *for player 1, we have that* $\Pr_s^{\sigma_1, \sigma_2^*}(\overline{\mathsf{LimSupAvg}}_{\leq}(1 - c)) = 1$*, where* $c = (\frac{\delta_{\min}}{m})^{n-1} \cdot \frac{1}{m}$

Proof. In the proof of Lemma 4 (Remark 5), we presented a witness stationary strategy σ_2^1 that ensured that the set $\overline{X}_1 = S \setminus \mu Y.\nu Z.\text{ASP}(S, Y, Z)$ was never left; and for all states $s \in \overline{X}_1$ and all strategies σ_1 for player 1 we have $\Pr_s^{\sigma_1,\sigma_2^*}(\overline{\mathsf{LimSupAvg}}_{\leq}(1-c)) = 1$. But notice that $S \setminus Y^* = \overline{X}_1$. Note also that σ_2^1 played uniformly over some subset of actions in $\Gamma_2(s)$ for any $s \in \overline{X}_1$. Hence, the patience of σ_2^1 is at most m. □

Lemma 14. *There is a Markov strategy σ_1^* for player 1 that ensures that for all states $s \in Y^*$ and all strategies σ_2 for player 2, we have that $\Pr_s^{\sigma_1^*,\sigma_2}(\mathsf{LimInfAvg}(1)) > 0$.*

Proof. Let $Y_0 = \emptyset$ and $Y_{i+1} = \nu Z.\text{ASP}(S, Y_i, Z)$. Also let ℓ be the smallest number such that $Y_{\ell+1} = Y_\ell$ and $Y^* = Y_\ell$. To construct σ_1^* we will first define a strategy σ_1^ϵ, for all $\epsilon > 0$. Fix $\epsilon > 0$ and we define σ_1^ϵ as follows: For $s \notin Y^*$ the strategy plays arbitrarily. For $s \in Y_i \setminus Y_{i-1}$ the strategy is as follows:

$$\sigma_1^\epsilon(s)(a) = \begin{cases} \frac{1-\epsilon}{Gd(s)} & \text{for } a \in \text{Good}_1(s, S, Y_{i-1}, Y_i) \text{ and } Gd(s) \neq Aw(s) \\ \frac{1}{Gd(s)} & \text{for } a \in \text{Good}_1(s, S, Y_{i-1}, Y_i) \text{ and } Gd(s) = Aw(s) \\ \frac{\epsilon}{Aw(s)-Gd(s)} & \text{for } a \notin \text{Good}_1(s, S, Y_{i-1}, Y_i), \end{cases}$$

where $Gd(s) = |\text{Good}_1(s, S, Y_{i-1}, Y_i)|$ and $Aw(s) = |\text{Allow}_1(s, S)| = |\Gamma_1(s)|$. By definition of Y_i, the set $\text{Good}_1(s, S, Y_{i-1}, Y_i)$ is not empty and hence this is well-defined.

The construction of the desired strategy σ_1^* is as follows: consider the sequence $\epsilon_1, \epsilon_2, \ldots$ such that $\epsilon_1 = \frac{1}{4}$ and $\epsilon_{i+1} = \frac{\epsilon_i}{2}$. In round k, the strategy σ_1^* will play according to $\sigma_1^{\epsilon_k}$. Note that this is a Markov strategy.

Let $s \in Y_i \setminus Y_{i-1}$. We will now show the statement using induction in i. More precisely, assume that we are in s in round j, we will show that either some state in Y_{i-1} is reached with positive probability or $\Pr_s^{\sigma_1^*,\sigma_2}(\mathsf{LimInfAvg}(1)) \geq \frac{1}{2}$.

For the base case, $i = 1$, notice that $Y_0 = \emptyset$. Hence we need to show that $\Pr_s^{\sigma_1^*,\sigma_2}(\mathsf{LimInfAvg}(1)) > 0$. By construction of σ_1^*, the probability for player 1 to ever play a action outside $\text{Good}_1(s, S, Y_0, Y_1)$ is $\sum_{k=j}^{\infty} \epsilon_k \leq \frac{1}{2}$. If no action outside $\text{Good}_1(s, S, Y_0, Y_1)$ is ever played we have by definition of $\text{Good}_1(s, S, Y_0, Y_1)$ that Y_1 is never left and we will in each step get a reward of 1.

For $i > 1$ there are two cases. Either player 2 plays an action in $\text{Bad}_2(s, S, Y_{i-1})$ with positive probability at some point or not. If not, the argument is identical to the base case (except that it is Y_i that will not be left with probability greater than a half). Otherwise, Y_{i-1} is reached with positive probability because all actions are played with positive probability by σ_1^* and the statement follows by induction. □

Theorem 15 (Qualitative determinacy and polynomial time computability)**.** *The following assertions hold for all concurrent game structures with boolean reward functions:*

1. *We have*

$$\begin{aligned} Y^* &= \mathsf{Positive}_1(\mathsf{LimInfAvg}(1)) = \mathsf{Positive}_1(\mathsf{LimSupAvg}(1)) \\ &= \bigcap_{\varepsilon>0} \mathsf{Positive}_1(\mathsf{LimInfAvg}(1-\varepsilon)) = \bigcap_{\varepsilon>0} \mathsf{Positive}_1(\mathsf{LimSupAvg}(1-\varepsilon)); \end{aligned}$$

and

$$\begin{aligned} S \setminus Y^* &= \mathsf{Almost}_2(\overline{\mathsf{LimInfAvg}}(1)) = \mathsf{Almost}_2(\overline{\mathsf{LimSupAvg}}(1)) \\ &= \bigcup_{c>0} \mathsf{Almost}_2(\overline{\mathsf{LimInfAvg}}(1-c)) = \bigcup_{c>0} \mathsf{Almost}_2(\overline{\mathsf{LimSupAvg}}(1-c)); \end{aligned}$$

where $Y^* = \mu Y.\nu Z.\text{ASP}(S, Y, Z)$.

2. *The set Y^* can be computed in quadratic time (in time $O(n \cdot |\delta|)$) where $|\delta| = \sum_{s \in S} \sum_{a \in \Gamma_1(s)} \sum_{b \in \Gamma_2(s)} |\mathrm{Dest}(s,a,b)|$, by the straight-forward computation of the μ-calculus formula $\mu Y.\nu Z.\mathrm{ASP}(S,Y,Z)$.*

Proof. The proof of the theorem is analogous to Theorem 6, and uses Lemma 14 and Lemma 13. □

4.2 Strategy complexity

In this section we will establish the complexities of the witness positive and almost-sure winning strategies for player 1 and player 2, from their respective winning sets.

Let $0 \leq \epsilon < 1$ be given. We will show that there exists games with a state s such that there exists σ_1^* such that for all σ_2 we have $\Pr_s^{\sigma_1^*,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon)) > 0$, but where no strategy with finite memory for player 1 ensures so.

Game with no finite-memory positive winning strategies. Let $\bar{G}$ be the following repeated game with absorbing states. The game $\bar{G}$ has 3 states, v^0, v^1 and v. For $i \in \{0,1\}$, state v^i is absorbing and has only one action for either player, a^i and b^i respectively and where $\mathrm{r}(v^i,a^i,b^i) = i$. The state v has two actions for either player. The actions are a_1 and a_2 for player 1 and b_1 and b_2 for player 2. Also $\mathrm{r}(v,a_i,b_j) = 0$ except for $i = j = 2$, for which $\mathrm{r}(v,a_2,b_2) = 1$. Furthermore $\delta(v,a_i,b_j) = v^0$ for $i \neq j$, $\delta(v,a_1,b_1) = v_1$ and $\delta(v,a_2,b_2) = v$. There is an illustration of $\bar{G}$ in Figure 5. Clearly state v^1 and state v are in Y^*, because a Markov strategy which in state v in round j plays action a_1 with probability $\frac{1}{2^{j+1}}$ and action a_2 with the remaining probability ensure $\mathsf{LimInfAvg}(1)$ with positive probability.

Lemma 16. *No finite-memory strategy σ_1^* for player 1 in $\bar{G}$ ensures that for all σ_2 and for all $0 \leq \epsilon < 1$ we have $\Pr_v^{\sigma_1^*,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon)) > 0$.*

Proof. The proof will be by contradiction. Consider $0 \leq \epsilon < 1$. Assume towards contradiction that a strategy σ_1 using finite memory for player 1 exists such that for all σ_2 we have $\Pr_v^{\sigma_1,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon)) > 0$. We will show that there is a σ_2 such that $\Pr_v^{\sigma_1,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon)) = 0$ to establish the contradiction.

We will divide the memory states of player 1 into two types. The two types are memory states of type 1, where σ_1 plays a_2 with probability 1 and memory states of type 2, where σ_1 plays a_2 with probability less than 1. The strategy σ_2 is then to play b_1, if, conditioned on the history so far, σ_1 is in a memory state of type 1, otherwise play b_2. Let p be the smallest non-zero probability with which σ_1 plays a_1. We see that in each round, if player 1 follows σ_1 and player 2 follows σ_2, there is a probability of at least p to reach v^0 and otherwise the plays stays in v. Clearly, we must therefore reach v^0 after some number of steps with probability 1, which will ensure that all remaining rewards are 0. Hence $\Pr_v^{\sigma_1,\sigma_2}(\mathsf{LimInfAvg}(1-\epsilon)) = 0$. This is a contradiction and the desired result follows. □

For completeness we will now show that there exists games with states s such that there exists σ_2 such that for all σ_1 we have $\Pr_s^{\sigma_1,\sigma_2}(\overline{\mathsf{LimInfAvg}}(1)) = 1$, but where no stationary strategy σ_2 with patience less than m exists.

Let m be some fixed number. The game $\underline{G^m}$ has 1 state, v. The state v has m actions for both players. The actions are $a_1, a_2, \ldots, a_m$ for player 1 and $b_1, b_2, \ldots, b_m$ for player 2. Also $\mathrm{r}(v,a_i,b_j) = 1$ for $i \neq j$ and $\mathrm{r}(v,a_i,b_i) = 0$. Furthermore $\delta(v,a_i,b_j) = v$. There is an illustration of $\underline{G^2}$ in Figure 6. Clearly state v is in $S \setminus Y^*$.

Lemma 17. *For all $m > 0$, no stationary strategy σ_2 for player 2 with patience less than m in $\underline{G^m}$ ensures that for all σ_1 we have $\Pr_v^{\sigma_1,\sigma_2}(\overline{\mathsf{LimInfAvg}}(1)) = 1$.*

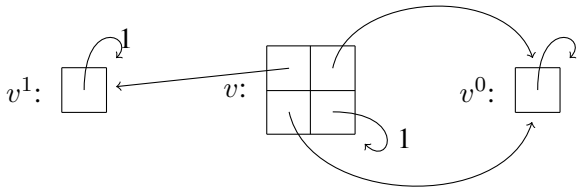

Figure 5: The figure shows $\bar{G}$ that will be used to show infinite-memory requirement for positive winning strategies.

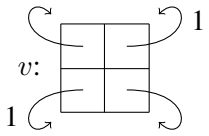

Figure 6: The example shows $\underline{G}^2$ of the game family $\underline{G}^n$ that will be used to show that patience m is required by player 2.

Proof. The proof will be by contradiction. Assume that such a strategy σ_2 for player 2 exists. Clearly it must play some action b_i with probability 0. Hence, if σ_1 plays a_i with probability 1, we have $\Pr_v^{\sigma_1,\sigma_2}(\mathsf{LimInfAvg}(1)) = 1$. That is a contradiction. □

Theorem 18 (Strategy complexity)**.** *The following assertions hold for concurrent games with boolean reward functions:*

1. *Positive winning strategies for player 1 for objectives* $\mathsf{LimInfAvg}(1)$ *and* $\mathsf{LimInfAvg}(1-\epsilon)$*, for* $\epsilon > 0$*, (also* $\mathsf{LimSupAvg}(1)$ *and* $\mathsf{LimSupAvg}(1-\epsilon)$*, for* $\epsilon > 0$*) require infinite-memory in general; whenever such positive winning strategies exist, Markov strategies are sufficient and the optimal bound for time-dependent memory is* T*, for all rounds* $T > 0$*.*

2. *Stationary almost-sure winning strategies for player 2 exist for objectives* $\overline{\mathsf{LimInfAvg}}(1)$ *and* $\overline{\mathsf{LimInfAvg}}(1-c)$*, for some constant* $c > 0$*, (also* $\overline{\mathsf{LimSupAvg}}(1)$ *and* $\overline{\mathsf{LimSupAvg}}(1-c)$*, for some constant* $c > 0$*) and the optimal bound for patience of such stationary almost-sure winning strategies is the size of the action space.*

Proof. The proofs are as follows:

1. Lemma 16 shows that infinite-memory is required, and Lemma 14 shows that Markov strategies are sufficient for positive winning. The sufficiency of Markov strategies establishes the T upper bound for time-dependent memory; and Lemma 16 and Lemma 7 establish the T lower bound for time-dependent memory.

2. The existence of stationary positive winning strategies with m patience follows from Lemma 13. The lower bound of m for patience follows from Lemma 17.

The desired result follows. □

Remark 19. *For all the results for almost-sure and positive winning with exact and limit qualitative constraints, we considered boolean reward functions. For general real-valued reward functions without loss of generality we can consider that the rewards are in the interval* $[0, 1]$ *by shifting and scaling of the rewards. Observe that all our results for boolean reward functions with* $\mathsf{LimInfAvg}(1)$ *and* $\mathsf{LimInfAvg}(1-\epsilon)$*, for all* $\epsilon > 0$ *(and also for* $\mathsf{LimSupAvg}$*) also hold for reward functions with rewards in the interval* $[0, 1]$*, since in our proof we can replace reward 0 by the maximal reward that is strictly less than 1.*

Remark 20. *Note that both for positive and almost-sure winning with exact and limit qualitative constraints we have presented quadratic time algorithms. For almost-sure winning the bound of our algorithm matches the best known bound for the special case of concurrent reachability games. For positive winning, concurrent reachability games can be solved in linear time. However for the special case of turn-based deterministic mean-payoff games with boolean rewards, the almost-sure and the positive winning sets for the exact and the limit qualitative constraints coincide with the winning set for coBüchi games, where the goal is to ensure that eventually a set* C *is reached and never left (the set* C *is the set of states with reward 1). The current best known algorithms for turn-based deterministic coBüchi games are quadratic, and our algorithm matches the quadratic bound known for the special case of turn-based deterministic games.*

5 Almost and Positive Winning for Quantitative Path Constraints

In this section our goal is to establish hardness results for polynomial time computability of $\mathsf{Almost}_1(\mathsf{LimInfAvg}(\lambda))$ and $\mathsf{Positive}_1(\mathsf{LimInfAvg}(\lambda))$, given λ is a rational number in the interval $(0, 1)$, for turn-based stochastic games with boolean reward functions. We first mention several related polynomial time computability results: (1) turn-based deterministic games with boolean reward functions can be solved in polynomial time (follows from [34] as the pseudo-polynomial time algorithm is polynomial for boolean rewards); (2) turn-based stochastic reachability games can be solved in polynomial time for almost-sure and positive winning (follows from the results of [10] that show a polynomial reduction to turn-based deterministic Büchi games for almost-sure and positive winning); and (3) turn-based stochastic and concurrent stochastic games can be solved in polynomial time if $\lambda = 1$ as established in the previous sections for almost-sure and positive winning. Hence our hardness result for almost-sure and positive winning for turn-based stochastic boolean reward games with $\lambda \neq 1$ is tight in the sense that relaxation to deterministic games, or reachability objectives, or $\lambda = 1$ ensures polynomial time computability. Our hardness result will be a reduction from the problem of deciding if $\mathrm{val}(s) \geq c$, given a constant $c \geq 0$ and a state s in a turn-based deterministic mean-payoff game with *arbitrary* rewards to the problem of deciding whether $t \in \mathsf{Almost}_1(\mathsf{LimInfAvg}(\lambda))$ in turn-based stochastic games with *boolean* rewards, for $\lambda \in (0, 1)$. Our reduction will also ensure that in the game obtained we have $\mathsf{Almost}_1(\mathsf{LimInfAvg}(\lambda)) = \mathsf{Positive}_1(\mathsf{LimInfAvg}(\lambda))$. Hence the hardness also follows for the problem of deciding whether $t \in \mathsf{Positive}_1(\mathsf{LimInfAvg}(\lambda))$ in turn-based stochastic games with boolean rewards. The polynomial computability of optimal values in turn-based deterministic mean-payoff games with arbitrary rewards is a long-standing open problem (the decision problem is in NP $\cap$ coNP, but no deterministic sub-exponential time algorithm is known). To present the reduction we first present an equivalent and convenient notation for turn-based deterministic and turn-based stochastic games.

Equivalent convenient notation for turn-based games. An equivalent formulation for turn-based stochastic games is as follows: a turn-based stochastic game $G = ((S, E), (S_1, S_2, S_P), \delta)$ consists of a finite set S of states, E of edges, a partition of the state space into player 1, player 2 and probabilistic states, (S_1, S_2, S_P, respectively) and a probabilistic transition function $\delta : S_P \to \mathcal{D}(S)$ such that for all $s \in S_P$ and $t \in S$ we have $(s, t) \in E$ iff $\delta(s)(t) > 0$. In a turn-based stochastic game, in player 1 states the successor state

is chosen by player 1 and likewise for player 2 states. In probabilistic states the successor state is chosen according to the probabilistic transition function δ. For a turn-based deterministic game we have $S_P = \emptyset$, and hence we do not need the transition function δ, and simply represent them as $G = ((S, E), (S_1, S_2))$.

Optimal values in DMPGs. A DMPG (deterministic mean-payoff game) consists of a turn-based deterministic game $G = ((S, E), (S_1, S_2))$ with a reward function $\mathrm{r} : E \to \{0, 1, \ldots, M\}$, (note that the rewards are non-negative integers and not necessarily boolean). The *optimal value* for a state s, denoted as $\mathrm{val}(s)$, is the maximal limit-inf-average value that player 1 can ensure with a positional strategy against all positional strategies of the opponent. Formally, given two positional strategies σ_1 and σ_2, and a starting state s, an unique cycle C is executed infinitely often, and the mean-payoff value for σ_1 and σ_2 from s, denoted $\mathrm{val}(s, \sigma_1, \sigma_2)$, is $\frac{\sum_{e \in C} \mathrm{r}(e)}{|C|}$, the sum of the rewards in C, divided by the length of C. Then $\mathrm{val}(s)$ is the value of state $s \in S$, that is, $\mathrm{val}(s) = \max_{\sigma_1} \min_{\sigma_2} \mathrm{val}(s, \sigma_1, \sigma_2)$, where σ_1 and σ_2 ranges over positional strategies of player 1 and player 2, respectively. Given a rational number λ, the decision problem of whether $\mathrm{val}(s) \geq \lambda$ lies in NP $\cap$ coNP and can be computed in pseudo-polynomial time (in time $O(|S| \cdot |E| \cdot M)$) for DMPGs [34, 6]. Finding an algorithm that runs in polynomial time and solves that decision problem is a long-standing open problem. We will reduce the computation of the value problem for DMPGs to almost-sure winning in turn-based stochastic games with boolean rewards but quantitative path constraints, i.e., $\mathsf{Almost}_1(\mathsf{LimInfAvg}(\lambda))$, for $\lambda \in (0, 1)$.

Reduction. Given the DMPG $G = ((S, E), (S_1, S_2))$ with non-negative integral rewards, with largest reward M, the construction of a turn-based stochastic game $G' = ((S', E'), (S'_1, S'_2, S_P), \delta')$ is as follows: $S'_1 = S_1$ and $S'_2 = S_2$; and for every edge $e = (s, t)$ in G with reward $\mathrm{r}(e)$, we replace the edge between s and t with a gadget with four additional states (namely, $v^1(e), v^2(e), v^3(e)$ and $v^4(e)$) along with s and t and eight edges with boolean rewards. The gadget is as follows: each state $v^i(e)$ is a probabilistic state for $i \in \{1, 2, 3, 4\}$, and from state s there is an edge corresponding to the edge e that goes to $v^1(e)$. The other transitions from the states in the gadget are specified below: (i) from $v^1(e)$ the next state is state $v^2(e)$ with probability $\frac{\mathrm{r}(e)}{M}$ and state $v^3(e)$ with probability $\frac{M - \mathrm{r}(e)}{M}$, and the edges have reward 0; (ii) the next state from either state $v^2(e)$ or state $v^3(e)$ is $v^4(e)$ with probability $1 - \frac{1}{M}$ and state t with probability $\frac{1}{M}$, and the edges from $v^2(e)$ have reward 1 and the edges from $v^3(e)$ have reward 0; and (iii) the next state from state $v^4(e)$ is $v^1(e)$ with edge reward 0. There is a illustration of the gadget in Figure 7. We refer to the boolean reward turn-based stochastic game obtained by the above reduction from a DMPG G as $G' = \mathsf{Red}(G)$. Also note that the reduction is polynomial as all the probabilities can be expressed in polynomial size given the input DMPG G.

Property of the reduction. Let $e = (s, t)$ be some edge in G with reward $\mathrm{r}(e)$. It is easy to verify that in the gadget, if the edge to $v^1(e)$ is taken from s in G', then we eventually reach t with probability 1, while we expect to get $\mathrm{r}(e)$ rewards of value 1 and $3 \cdot M - \mathrm{r}(e)$ rewards of value 0, before reaching t. The expected number of steps to reach t from s is thus always $3 \cdot M$ and is independent of the reward value $\mathrm{r}(e)$. Hence the total expected reward is $\mathrm{r}(e)$ and one step of the game G is simulated by $3 \cdot M$ steps in G'. We will show that if a state s in G has optimal value $\mathrm{val}(s)$, then the corresponding state in G' is in $\mathsf{Almost}_1(\mathsf{LimInfAvg}(\frac{\mathrm{val}(s)}{3M}))$. Also, we will show that if a state in G' is in $\mathsf{Almost}_1(\mathsf{LimInfAvg}(\lambda))$, then the corresponding state in G has optimal value of at least $3 \cdot M \cdot \lambda$. We present the results in the following two lemmas.

One basic property of Markov chains. In both the lemmas we will use the following *basic property* of a Markov chain. Consider a Markov chain with arbitrary rewards, and closed recurrent set C of the Markov chain. Let α be the expected mean-payoff value from a starting state s in C (the expected mean-payoff value is independent of the start state since C is a closed recurrent set). Then for all $s \in C$, we have

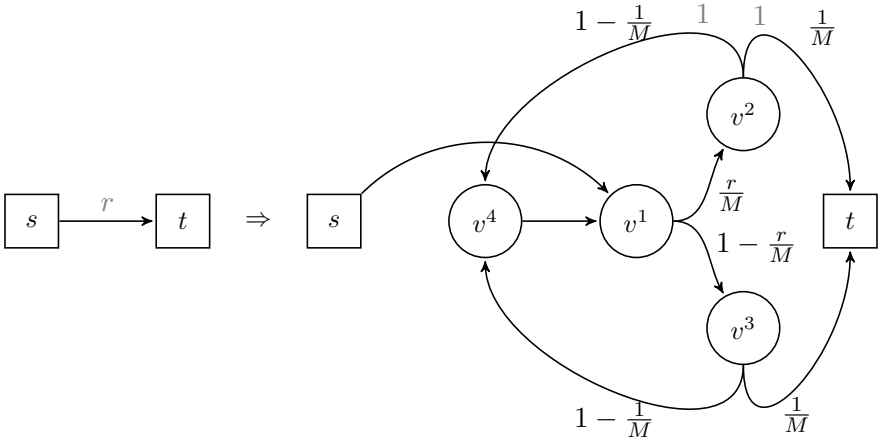

Figure 7: Our gadget for reducing a DMPG to a turn-based stochastic boolean reward game: The edges that go to more than one state have the probability annotated on them in black; and any non-zero reward is annotated on the corresponding edge in gray.

$s \in \mathsf{Almost}_1(\mathsf{LimInfAvg}(\alpha))$ and for all $\alpha' > \alpha$ we have $s \notin \mathsf{Positive}_1(\mathsf{LimSupAvg}(\alpha'))$, i.e., almost-surely the mean-payoff value is at least α, and for every $\alpha' > \alpha$ the mean-payoff is at least α' with probability 0. The above basic property result follows by the almost-sure convergence to the invariant distribution (or Cesaro limit) for a closed recurrent set of a Markov chain.

Lemma 21. *Given a DMPG G with largest reward M for a state s in G we have that the corresponding state in G' belongs to* $\mathsf{Almost}_1(\mathsf{LimInfAvg}(\frac{\mathrm{val}(s)}{3M}))$ *and* $\mathsf{Positive}_1(\mathsf{LimInfAvg}(\frac{\mathrm{val}(s)}{3M}))$*, where* $G' = \mathsf{Red}(G)$.

Proof. Consider an optimal positional (pure and stationary) strategy σ_1 for player 1 in G (such an optimal strategy exists in DMPGs [15]). The strategy ensures that LimInfAvg is at least $\mathrm{val}(s)$ if the play starts in s against any strategy for player 2. Consider the corresponding strategy σ_1' of σ_1 in G'. Consider a positional best response strategy σ_2' for player 2 in G' to σ_1', if the play starts in the state that corresponds to state s. The play in G' given σ_1' and σ_2' reaches an unique closed recurrent set C' with probability 1 (i.e., the set C' corresponds to the unique cycle C reachable from s given strategies σ_1 and the corresponding strategy σ_2 of σ_2', and the states introduced by the gadget). We have the following desired properties. First, in the closed recurrent set C' of G' the expected limit-average payoff is at least $\frac{\mathrm{val}(s)}{3 \cdot M}$, since the average reward of the cycle C in G is at least $\mathrm{val}(s)$, and in G' every step of G is simulated by $3 \cdot M$ steps with the same total reward value in expectation. Second, since we have a Markov chain, the expectation and almost-sure satisfaction coincide for closed recurrent set (the basic property of Markov chains). Finally, in G' the closed recurrent set C' is reached with probability 1 given the strategies σ_1' and σ_2', from the starting state corresponding to s. This shows that the corresponding state to s in G' belongs to $\mathsf{Almost}_1(\mathsf{LimInfAvg}(\frac{\mathrm{val}(s)}{3M}))$ and $\mathsf{Positive}_1(\mathsf{LimInfAvg}(\frac{\mathrm{val}(s)}{3M}))$. □

Lemma 22. *Given a DMPG G with largest reward M for a state s in G, if the corresponding state in G' belongs to either* $\mathsf{Almost}_1(\mathsf{LimInfAvg}(\lambda))$ *or* $\mathsf{Positive}_1(\mathsf{LimInfAvg}(\lambda))$*, then* $\mathrm{val}(s) \geq 3 \cdot M \cdot \lambda$*, where* $G' = \mathsf{Red}(G)$.

Proof. Consider a positional strategy for player 1 in G' (such a strategy exists since we consider turn-based stochastic games) to ensure $\mathsf{LimInfAvg}(\lambda)$ with probability 1 from the state corresponding to s. Consider the corresponding strategy σ_1 in G and a positional best response strategy σ_2 of player 2 in G, and consider the corresponding strategy σ_2' of σ_2 in G'. Let the unique cycle executed in G given σ_1 and σ_2 from s be C. The unique closed recurrent set reached with probability 1 in G' from s given σ_1' and σ_2' is C'. Hence, the set C' consists of the states in C along with the gadget states of C the reduction. Since the state s belongs to $\mathsf{Almost}_1(\mathsf{LimInfAvg}(\lambda))$ or $\mathsf{Positive}_1(\mathsf{LimInfAvg}(\lambda))$ in G', it follows from the basic property of Markov chains that the expected average reward of the closed recurrent set C' is at least λ. Since every step of G is simulated by $3 \cdot M$ steps in G' it follows that the average reward of the cycle C must be at least $3 \cdot M \cdot \lambda$. This completes the proof. □

The following theorem follows from the two previous lemmas and establish the desired hardness result.

Theorem 23 (Hardness for quantitative constraints)**.** *Given a DMPG G, a state s and a rational value λ, we have* $\mathrm{val}(s) \geq \lambda$ *in G iff* $s \in \mathsf{Almost}_1(\mathsf{LimInfAvg}(\frac{\lambda}{3 \cdot M})) = \mathsf{Positive}_1(\mathsf{LimInfAvg}(\frac{\lambda}{3 \cdot M}))$ *in* $G' = \mathsf{Red}(G)$.

6 Conclusion

In this work we considered qualitative analysis of concurrent mean-payoff games. For qualitative constraints, we established the qualitative determinacy results; presented quadratic algorithms to compute almost-sure and positive winning sets (matching the best known bounds for the simpler case of reachability objectives or turn-based deterministic games); and presented a complete characterization of the strategy complexity. We established a hardness result for qualitative analysis with quantitative path constraints.

References

[1] C. Baier, M. Größer, and N. Bertrand. Probabilistic omega-automata. *J. ACM*, 59(1), 2012.

[2] N. Bertrand, B. Genest, and H. Gimbert. Qualitative determinacy and decidability of stochastic games with signals. In *Proc. of LICS*, pages 319–328, 2009.

[3] T. Bewley and E. Kohlberg. The asymptotic behavior of stochastic games. *Math Oper Research*, (1), 1976.

[4] D. Blackwell and T.S. Ferguson. The big match. *Annals of Mathematical Statistics*, 39:159–163, 1968.

[5] T. Brázdil, V. Brozek, A. Kucera, and J. Obdrzálek. Qualitative reachability in stochastic bpa games. *Inf. Comput.*, 209(8):1160–1183, 2011.

[6] L. Brim, J. Chaloupka, L. Doyen, R. Gentilini, and J-F. Raskin. Faster algorithms for mean-payoff games. *Formal Methods in System Design*, 38(2):97–118, 2011.

[7] K. Chatterjee, M. Chmelik, and M. Tracol. What is decidable about partially observable Markov decision processes with omega-regular objectives. In *Proceedings of CSL 2013: Computer Science Logic*, 2013.

[8] K. Chatterjee and L. Doyen. Partial-observation stochastic games: How to win when belief fails. In *LICS*, 2012.

[9] K. Chatterjee, L. Doyen, T. A. Henzinger, and J.-F. Raskin. Algorithms for omega-regular games of incomplete information. *Logical Methods in Computer Science*, 3(3:4), 2007.

[10] K. Chatterjee, M. Jurdziński, and T.A. Henzinger. Simple stochastic parity games. In *CSL'03*, volume 2803 of *LNCS*, pages 100–113. Springer, 2003.

[11] K. Chatterjee, R. Majumdar, and T. A. Henzinger. Stochastic limit-average games are in exptime. *Int. J. Game Theory*, 37(2):219–234, 2008.

[12] A. Condon. The complexity of stochastic games. *Information and Computation*, 96(2):203–224, 1992.

[13] L. de Alfaro, M. Faella, R. Majumdar, and V. Raman. Code-aware resource management. In *EMSOFT 05*. ACM, 2005.

[14] L. de Alfaro, T.A. Henzinger, and O. Kupferman. Concurrent reachability games. In *FOCS'98*, pages 564–575. IEEE, 1998.

[15] A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *Int. Journal of Game Theory*, 8(2):109–113, 1979.

[16] K. Etessami and M. Yannakakis. Recursive Markov decision processes and recursive stochastic games. In *ICALP'05*, LNCS 3580, Springer, pages 891–903, 2005.

[17] K. Etessami and M. Yannakakis. Recursive concurrent stochastic games. In *ICALP'06 (2)*, LNCS 4052, Springer, pages 324–335, 2006.

[18] H. Everett. Recursive games. In *Contributions to the Theory of Games III*, volume 39 of *Annals of Mathematical Studies*, pages 47–78, 1957.

[19] D. Gillette. Stochastic games with zero stop probabilitites. In *Contributions to the Theory of Games III*, pages 179–188. Princeton University Press, 1957.

[20] V. A. Gurvich, A. V. Karzanov, and L. G. Khachiyan. Cyclic games and an algorithm to find minimax cycle means in directed graphs. *USSR Comput. Math. Math. Phys.*, 28(5):85–91, April 1990.

[21] K. A. Hansen, R. Ibsen-Jensen, and P. B. Miltersen. The complexity of solving reachability games using value and strategy iteration. In *CSR*, pages 77–90, 2011.

[22] K. A. Hansen, M. Koucký, N. Lauritzen, P. B. Miltersen, and E. P. Tsigaridas. Exact algorithms for solving stochastic games: extended abstract. In *STOC*, pages 205–214, 2011.

[23] K. A. Hansen, M. Koucký, and P. B. Miltersen. Winning concurrent reachability games requires doubly-exponential patience. In *LICS*, pages 332–341, 2009.

[24] A.J. Hoffman and R.M. Karp. On nonterminating stochastic games. *Management Sciences*, 12(5):359–370, 1966.

[25] M. Jurdzinski. Small progress measures for solving parity games. In *STACS'00*, pages 290–301. LNCS 1770, Springer, 2000.

[26] M. Kwiatkowska, G. Norman, and D. Parker. Verifying randomized distributed algorithms with prism. In *Workshop on Advances in Verification (WAVE'00)*, 2000.

[27] T. A. Liggett and S. A. Lippman. Stochastic games with perfect information and time average payoff. *Siam Review*, 11:604–607, 1969.

[28] J.F. Mertens and A. Neyman. Stochastic games. *International Journal of Game Theory*, 10:53–66, 1981.

[29] S. Nain and M. Y. Vardi. Solving partial-information stochastic parity games. In *LICS*, pages 341–348, 2013.

[30] A. Pogosyants, R. Segala, and N. Lynch. Verification of the randomized consensus algorithm of Aspnes and Herlihy: a case study. *Distributed Computing*, 13(3):155–186, 2000.

[31] L.S. Shapley. Stochastic games. *Proc. Nat. Acad. Sci. USA*, 39:1095–1100, 1953.

[32] M.I.A. Stoelinga. Fun with FireWire: Experiments with verifying the IEEE1394 root contention protocol. In *Formal Aspects of Computing*, 2002.

[33] M.Y. Vardi. Automatic verification of probabilistic concurrent finite-state systems. In *FOCS'85*, pages 327–338. IEEE Computer Society Press, 1985.

[34] U. Zwick and M. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158:343–359, 1996.