Different Strokes in Randomised Strategies: Revisiting Kuhn's Theorem Under Finite-Memory Assumptions

James C. A. Main
F.R.S.-FNRS & UMONS – Université de Mons, Belgium

Mickael Randour
F.R.S.-FNRS & UMONS – Université de Mons, Belgium

Abstract

Two-player (antagonistic) games on (possibly stochastic) graphs are a prevalent model in theoretical computer science, notably as a framework for reactive synthesis.

Optimal strategies may require randomisation when dealing with inherently probabilistic goals, balancing multiple objectives, or in contexts of partial information. There is no unique way to define randomised strategies. For instance, one can use so-called *mixed* strategies or *behavioural* ones. In the most general settings, these two classes do not share the same expressiveness. A seminal result in game theory – *Kuhn's theorem* – asserts their equivalence in games of perfect recall.

This result crucially relies on the possibility for strategies to use *infinite memory*, i.e., unlimited knowledge of all past observations. However, computer systems are finite in practice. Hence it is pertinent to restrict our attention to *finite-memory* strategies, defined as automata with outputs. Randomisation can be implemented in these in different ways: the *initialisation*, *outputs* or *transitions* can be randomised or deterministic respectively. Depending on which aspects are randomised, the expressiveness of the corresponding class of finite-memory strategies differs.

In this work, we study two-player turn-based stochastic games and provide a complete taxonomy of the classes of finite-memory strategies obtained by varying which of the three aforementioned components are randomised. Our taxonomy holds both in settings of perfect and imperfect information, and in games with more than two players.

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

Keywords and phrases two-player games on graphs, stochastic games, Markov decision processes, finite-memory strategies, randomised strategies

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2022.22

Related Version *Full Version*: `https://arxiv.org/abs/2201.10825` [31]

Funding *James C. A. Main*: F.R.S.-FNRS Research Fellow.
Mickael Randour: F.R.S.-FNRS Research Associate, member of the TRAIL Institute.

1 Introduction

Games on graphs. Games on (possibly stochastic) graphs have been studied for decades, both for their own interest (e.g., [25, 20, 27]) and for their value as a framework for *reactive synthesis* (e.g., [28, 35, 9, 3]). The core problem is almost always to find *optimal strategies* for the players: strategies that guarantee winning for Boolean winning conditions (e.g., [26, 39, 12, 10]), or strategies that achieve the best possible payoff in quantitative contexts (e.g., [25, 5, 13]). In multi-objective settings, one is interested in *Pareto-optimal* strategies (e.g., [19, 38, 36, 23]), but the bottom line is the same: players are looking for strategies that guarantee the best possible results.

In reactive synthesis, we model the interaction between a system and its uncontrollable environment as a two-player antagonistic game, and we represent the specification to ensure as a winning objective. An optimal strategy for the system in this game then constitutes a formal blueprint for a *controller* to implement in the real world [3].

Randomness in strategies. In essence, a *pure strategy* is simply a function mapping histories (i.e., the past and present of a play) to an action deterministically.

Optimal strategies may require *randomisation* when dealing with inherently probabilistic goals, balancing multiple objectives, or in contexts of partial information: see, e.g., [15, 36, 2, 23]. There are different ways of randomising strategies. For instance, a *mixed* strategy is essentially a probability distribution over a set of pure strategies. That is, the player randomly selects a pure strategy at the beginning of the game and then follows it for the entirety of the play without resorting to randomness ever again. By contrast, a *behavioural* strategy randomly selects an action at each step: it thus maps histories to probability distributions over actions.

Kuhn's theorem. In full generality, these two definitions yield different classes of strategies (e.g., [21], [34, Chapter 11]). Nonetheless, Kuhn's theorem [1] proves their equivalence under a mild hypothesis: in games of *perfect recall*, for any mixed strategy there is an equivalent behavioural strategy and vice-versa. A game is said to be of perfect recall for a given player if said player never forgets their previous knowledge and the actions they have played (i.e., they can observe their own actions). Let us note that perfect recall and *perfect information* are two different notions: perfect information is not required to have perfect recall.

Let us highlight that Kuhn's theorem crucially relies on two elements. First, mixed strategies can be distributions over an *infinite* set of pure strategies. Second, strategies can use *infinite memory*, i.e., they are able to remember the past completely, however long it might be. Indeed, consider a game in which a player can choose one of two actions in each round. One could define a (memoryless) behavioural strategy that selects one of the two actions by flipping a coin each round. This strategy generates infinitely many sequences of actions, therefore any equivalent mixed strategy needs the ability to randomise between infinitely many different sequences, and thus, infinitely many pure strategies. Moreover, infinitely many of these sequences require infinite memory to be generated (due to their non-regularity).

Finite-memory strategies. From the point of view of reactive synthesis, infinite-memory strategies, along with randomised ones relying on infinite supports, are undesirable for implementation. This is why a plethora of recent advances has focused on *finite-memory* strategies, usually represented as (a variation on) Mealy machines, i.e., finite automata with outputs. See, e.g., [27, 19, 11, 23, 4, 6]. Randomisation can be implemented in these finite-memory strategies in different ways: the *initialisation*, *outputs* or *transitions* can be randomised or deterministic respectively.

Depending on which aspects are randomised, the expressiveness of the corresponding class of finite-memory strategies differs: in a nutshell, *Kuhn's theorem crumbles when restricting ourselves to finite memory.* For instance, we show that some finite-memory strategies with only randomised outputs (i.e., the natural equivalent of behavioural strategies) cannot be emulated by finite-memory strategies with only randomised initialisation (i.e., the natural equivalent of mixed strategies) – see Lemma 7. Similarly, it is known that some finite-memory strategies that are encoded by Mealy machines using randomisation in all three components admit no equivalent using randomisation only in outputs [22, 21].

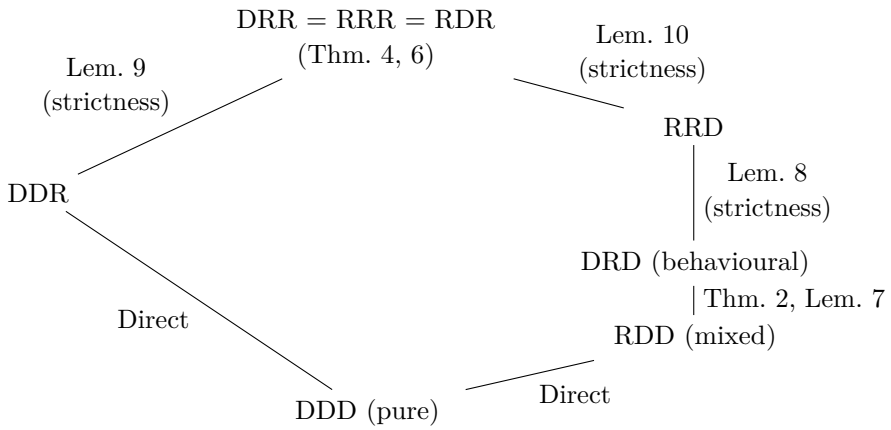


Figure 1.1 Lattice of strategy classes in terms of expressible probability distributions over plays against all strategies of the other player. In the three-letter acronyms, the letters, in order, refer to the initialisation, outputs and updates of the Mealy machines: D and R respectively denote deterministic and randomised components.

Our contributions. We consider *two-player zero-sum stochastic games* (e.g., [37, 20, 32, 6]), encompassing two-player (deterministic) games and Markov decision processes as particular subcases. We establish a *Kuhn-like taxonomy* of the classes of finite-memory strategies obtained by varying which of the three aforementioned components are randomised: we illustrate it in Figure 1.1, and describe it fully in Section 3.

Let us highlight a few elements. Naturally, the least expressive model corresponds to pure strategies. In contrast to what happens with infinite memory, and as noted in the previous paragraph, we see that mixed strategies are strictly less expressive than behavioural ones. We also observe that allowing randomness both in initialisation and in outputs (RRD strategies) yields an even more expressive class – and incomparable to what is obtained by allowing randomness in updates only. Finally, the most expressive class is obviously obtained when allowing randomness in all components; yet it may be dropped in initialisation or in outputs without reducing the expressiveness – but not in both simultaneously.

To compare the expressiveness of strategy classes, we consider *outcome-equivalence*, as defined in Section 2. Intuitively, two strategies are outcome-equivalent if, against any strategy of the opponent, they yield identical probability distributions (i.e., they induce identical Markov chains). Hence we are agnostic with regard to the objective, winning condition, payoff function, or preference relation of the game, and with regard to how they are defined (e.g., colours on actions, states, transitions, etc).

Finally, let us note that in our setting of two-player stochastic games, the perfect recall hypothesis holds. Most importantly, we assume that actions are visible. Lifting this hypothesis drastically changes the relationships between the different models. While our main presentation considers two-player perfect-information games for the sake of simplicity, we argue in Section 6 that *our results hold in games of imperfect information* too, assuming visible actions, and that our results hold in games with more than two players.

Related work. There are three main axes of research related to our work.

The first one deals with the *various types of randomness* one can inject in strategies and their consequences. Obviously, Kuhn's theorem [1] is a major inspiration, as well as the examples of differences between strategy models presented in [21]. On a different but related note, [16] studies when randomness is not helpful in games nor strategies (as it can be simulated by other means).

The second direction focuses on trying to characterise the *power of finite-memory strategies*, with or without randomness. One can notably cite [27] for memoryless strategies, and [30, 4, 6], and [7] for finite-memory ones in deterministic, stochastic, and infinite-arena games respectively.

The third axis concentrates on the use of *randomness as a means to simplify strategies* and/or reduce their memory requirements. Examples of this endeavour can be found in [14, 17, 29, 19, 33]. These are further motivations to understand randomised strategies even in contexts where randomness is not needed a priori to play optimally.

Outline. Due to space constraints, we only provide an overview of our work. All technical details and proofs can be found in the full version of this paper [31]. Section 2 summarises all preliminary notions. In Section 3, we present the taxonomy illustrated in Figure 1.1 and comment on it. We divide its proofs into two sections: Section 4 establishes the inclusions, and Section 5 proves their strictness. Finally, we discuss extensions of our results to games with imperfect information and multi-player games in Section 6.

2 Preliminaries

Probability. Given any finite or countable set A, we write $\mathcal{D}(A)$ for the set of probability distributions over A, i.e., the set of functions $p\colon A \to [0,1]$ such that $\sum_{a\in A} p(a) = 1$. Similarly, given some set A and some σ-algebra $\mathcal{F}$ over A, we denote by $\mathcal{D}(A,\mathcal{F})$ the set of probability distributions over the measurable space $(A,\mathcal{F})$.

Games. We consider two-player stochastic games of perfect information played on graphs. We denote the two players by $\mathcal{P}_1$ and $\mathcal{P}_2$. In such a game, the set of states is partitioned between the two players. At the start of a play, a pebble is placed on some initial state and each round, the owner of the current state selects an action available in said state and the next state is chosen randomly following a distribution depending on the current state and chosen action. The game proceeds for an infinite number of rounds, yielding an infinite play.

Formally, a (two-player) *stochastic game* (of perfect information) is defined as a tuple $\mathcal{G} = (S_1, S_2, A, \delta)$ where $S = S_1 \uplus S_2$ is a non-empty finite set of states partitioned into a set S_1 of states of $\mathcal{P}_1$ and a set S_2 of states of $\mathcal{P}_2$, A is a finite set of actions and $\delta\colon S \times A \to \mathcal{D}(S)$ is a (partial) probabilistic transition function. For any state $s \in S$, we write $A(s)$ for the set of actions available in s, which are the actions $a \in A$ such that $\delta(s,a)$ is defined. We assume that for all $s \in S$, $A(s)$ is non-empty, i.e., there are no deadlocks in the game.

A *play* of $\mathcal{G}$ is a sequence $s_0a_0s_1\ldots \in (SA)^\omega$ such that for all $k \in \mathbb{N}$, $\delta(s_k,a_k)(s_{k+1}) > 0$. A *history* is a finite prefix of a play ending in a state. Given a play $\pi = s_0a_0s_1a_1\ldots$ and $k \in \mathbb{N}$, we write $\pi_{|k}$ for the history $s_0a_0\ldots a_{k-1}s_k$. For any history $h = s_0a_0\ldots a_{k-1}s_k$, we let $\mathsf{last}(h) = s_k$. We write $\mathsf{Plays}(\mathcal{G})$ to denote the set of plays of $\mathcal{G}$, $\mathsf{Hist}(\mathcal{G})$ to denote the set of histories of $\mathcal{G}$ and $\mathsf{Hist}_i(\mathcal{G}) = \mathsf{Hist}(\mathcal{G}) \cap (SA)^*S_i$ for the set of histories ending in states controlled by $\mathcal{P}_i$. Given some initial state $s_{\mathsf{init}} \in S$, we write $\mathsf{Plays}(\mathcal{G}, s_{\mathsf{init}})$ and $\mathsf{Hist}(\mathcal{G}, s_{\mathsf{init}})$ for the set of plays and histories starting in state s_{init} respectively.

An interesting class of stochastic games which has been extensively studied is that of *deterministic games*; a game $\mathcal{G} = (S_1, S_2, A, \delta)$ is a deterministic game if for all $s \in S$ and $a \in A(s)$, $\delta(s,a)$ is a Dirac distribution. Another interesting class of games is that of one-player games. A game $\mathcal{G} = (S_1, S_2, A, \delta)$ is a one-player game of $\mathcal{P}_i$ if S_{3-i} is empty, i.e., all states belong to $\mathcal{P}_i$. These one-player games are the equivalent of *Markov decision processes* in our context, and will be referred to as such.

Strategies and outcomes. A strategy is a function that describes how a player should act based on a history. Players need not act in a deterministic fashion: they can use randomisation to select an action. Formally, a *strategy* of $\mathcal{P}_i$ is a function $\sigma_i \colon \mathsf{Hist}_i(\mathcal{G}) \to \mathcal{D}(A)$ such that for all histories h and all actions $a \in A$, $\sigma_i(h)(a) > 0$ implies $a \in A(\mathsf{last}(h))$. In other words, a strategy assigns to any history ending in a state controlled by $\mathcal{P}_i$ a distribution over the actions available in this state.

When both players fix a strategy and an initial state is decided, the game becomes a purely stochastic process (a Markov chain). Let us recall the relevant σ-algebra for the definition of probabilities over plays. For any history h, we define $\mathsf{Cyl}(h) = \{\pi \in \mathsf{Plays}(\mathcal{G}) \mid h \text{ is a prefix of } \pi\}$, the *cylinder* of h, consisting of plays that extend h. Let us denote by $\mathcal{F}_{\mathcal{G}}$ the σ-algebra generated by all cylinder sets.

Let σ_1 and σ_2 be strategies of $\mathcal{P}_1$ and $\mathcal{P}_2$ respectively and $s_{\mathsf{init}} \in S$ be an initial state. We define the probability measure (over $(\mathsf{Plays}(\mathcal{G}), \mathcal{F}_{\mathcal{G}})$) induced by playing σ_1 and σ_2 from s_{init} in $\mathcal{G}$, written $\mathbb{P}^{\sigma_1,\sigma_2}_{s_{\mathsf{init}}}$, in the following way: for any history $h = s_0 a_0 \dots s_n \in \mathsf{Hist}(\mathcal{G}, s_{\mathsf{init}})$, the probability assigned to $\mathsf{Cyl}(h)$ is given by the product

$$\mathbb{P}^{\sigma_1,\sigma_2}_{s_{\mathsf{init}}}(\mathsf{Cyl}(h)) = \prod_{k=0}^{n-1} \tau_k(s_0 a_0 \dots s_k)(a_k) \cdot \delta(s_k, a_k, s_{k+1}),$$

where $\tau_k = \sigma_1$ if $s_k \in S_1$ and $\tau_k = \sigma_2$ otherwise. For any history $h \in \mathsf{Hist}(\mathcal{G}) \setminus \mathsf{Hist}(\mathcal{G}, s_{\mathsf{init}})$, we set $\mathbb{P}^{\sigma_1,\sigma_2}_{s_{\mathsf{init}}}(\mathsf{Cyl}(h)) = 0$. By Carathéodory's extension theorem [24, Theorem A.1.3], the measure described above can be extended in a unique fashion to $(\mathsf{Plays}(\mathcal{G}), \mathcal{F}_{\mathcal{G}})$.

Let σ_1 be a strategy of $\mathcal{P}_1$. A play or prefix of play $s_0 a_0 s_1 \dots$ is said to be *consistent* with σ_1 if for all indices k, $s_k \in S_1$ implies $\sigma_1(s_0 a_0 \dots s_k)(a_k) > 0$.[1] Consistency with respect to strategies of $\mathcal{P}_2$ is defined analogously.

Outcome-equivalence of strategies. In the next sections, we study the expressiveness of finite-memory strategy models depending on the type of randomisation allowed. Two strategies may yield the same outcomes despite being different: the actions suggested by a strategy in an inconsistent history can be changed without affecting the outcome. Therefore, instead of using the equality of strategies as a measure of equivalence, we consider some weaker notion of equivalence, referred to as outcome-equivalence.

We say that two strategies σ_1 and τ_1 of $\mathcal{P}_1$ are *outcome-equivalent* if for any strategy σ_2 of $\mathcal{P}_2$ and for any initial state s_{init}, the probability distributions $\mathbb{P}^{\sigma_1,\sigma_2}_{s_{\mathsf{init}}}$ and $\mathbb{P}^{\tau_1,\sigma_2}_{s_{\mathsf{init}}}$ coincide. Outcome-equivalence of strategies can also be established without invoking induced probability distributions; two strategies are outcome-equivalent if and only if they coincide over the set of histories consistent with (one of) them. This can be shown by exploiting the definition of the probability of cylinder sets.

▶ **Lemma 1.** *Let σ_1 and τ_1 be two strategies of $\mathcal{P}_1$. These two strategies are outcome-equivalent if and only if for all histories $h \in \mathsf{Hist}_1(\mathcal{G})$, h consistent with σ_1 implies $\sigma_1(h) = \tau_1(h)$.*

Proof. Let us assume that σ_1 and τ_1 are outcome-equivalent. Let $h \in \mathsf{Hist}_1(\mathcal{G})$ be a history that is consistent with σ_1. Let s_{init} denote the first state of h and let σ_2 be a strategy of $\mathcal{P}_2$ consistent with h. It follows from the definition of the probability distribution of cylinders that for any $a \in A(\mathsf{last}(h))$ and any $s \in \mathsf{supp}(\delta(\mathsf{last}(h), a))$, we have

[1] We use the terminology of consistency not only for plays and histories, but also for prefixes of plays that end with an action.

$$\sigma_1(h)(a) = \frac{\mathbb{P}^{\sigma_1,\sigma_2}_{s_{\mathsf{init}}}(\mathsf{Cyl}(has))}{\mathbb{P}^{\sigma_1,\sigma_2}_{s_{\mathsf{init}}}(\mathsf{Cyl}(h)) \cdot \delta(\mathsf{last}(h),a)(s)} = \frac{\mathbb{P}^{\tau_1,\sigma_2}_{s_{\mathsf{init}}}(\mathsf{Cyl}(has))}{\mathbb{P}^{\tau_1,\sigma_2}_{s_{\mathsf{init}}}(\mathsf{Cyl}(h)) \cdot \delta(\mathsf{last}(h),a)(s)} = \tau_1(h)(a),$$

which shows that $\sigma_1(h) = \tau_1(h)$. This ends the proof of the first direction.

Let us now assume that σ_1 and τ_1 coincide over histories consistent with σ_1. Let σ_2 be a strategy of $\mathcal{P}_2$ and $s_{\mathsf{init}} \in S$ be an initial state. It suffices to study the probability of cylinder sets. Let $h \in \mathsf{Hist}(\mathcal{G})$ be a history starting in s_{init}. If h is consistent with σ_1, then all prefixes of h also are, therefore the definition of the probability of a cylinder ensures that $\mathbb{P}^{\sigma_1,\sigma_2}_{s_{\mathsf{init}}}(h) = \mathbb{P}^{\tau_1,\sigma_2}_{s_{\mathsf{init}}}(h)$. Otherwise, if h is not consistent with σ_1, then h is necessarily of the form $h'ah''$ with h' consistent with σ_1 and $\sigma_1(h')(a) = 0$. It follows that $\tau_1(h')(a) = 0$, thus $\mathbb{P}^{\sigma_1,\sigma_2}_{s_{\mathsf{init}}}(h) = \mathbb{P}^{\tau_1,\sigma_2}_{s_{\mathsf{init}}}(h) = 0$. This shows that σ_1 and σ_2 are outcome-equivalent, ending the proof. ◀

Subclasses of strategies. A strategy is called *pure* if it does not use randomisation; a pure strategy can be viewed as a function $\mathsf{Hist}_i(\mathcal{G}) \to A$. A strategy that only uses information on the current state of the play is called *memoryless*: a strategy σ_i of $\mathcal{P}_i$ is memoryless if for all histories $h, h' \in \mathsf{Hist}_i(\mathcal{G})$, $\mathsf{last}(h) = \mathsf{last}(h')$ implies $\sigma_i(h) = \sigma_i(h')$. Memoryless strategies can be viewed as functions $S_i \to \mathcal{D}(A)$. Strategies that are both memoryless and pure can be viewed as functions $S_i \to A$.

A strategy σ is said to be *finite-memory* (FM) if it can be encoded by a Mealy machine, i.e., an automaton with outputs along its edges. We can include randomisation in the initialisation, outputs and updates (i.e., transitions) of the Mealy machine. Formally, a *stochastic Mealy machine* of $\mathcal{P}_i$ is a tuple $\mathcal{M} = (M, \mu_{\mathsf{init}}, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$, where M is a finite set of memory states, $\mu_{\mathsf{init}} \in \mathcal{D}(M)$ is an initial distribution, $\alpha_{\mathsf{next}} \colon M \times S_i \to \mathcal{D}(A)$ is the (stochastic) next-move function and $\alpha_{\mathsf{up}} \colon M \times S \times A \to \mathcal{D}(M)$ is the (stochastic) update function.

Before we explain how to define the strategy induced by a Mealy machine, let us first describe how these machines work. Fix a Mealy machine $\mathcal{M} = (M, \mu_{\mathsf{init}}, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$. Let $s_0 \in S$. At the start of a play, an initial memory state m_0 is selected randomly following μ_{init}. Then, at each step of the play such that $s_k \in S_i$, an action a_k is chosen following the distribution $\alpha_{\mathsf{next}}(m_k, s_k)$, and otherwise an action is chosen following the other player's strategy. The memory state m_{k+1} is then randomly selected following the distribution $\alpha_{\mathsf{up}}(m_k, s_k, a_k)$ and the game state s_{k+1} is chosen following the distribution $\delta(s_k, a_k)$, both choices being made independently.

Let us now explain how a strategy can be derived from a Mealy machine. As explained previously, when in a certain memory state $m \in M$ and game state $s \in S_i$, the probability of an action $a \in A(s)$ being chosen is given by $\alpha_{\mathsf{next}}(m, s)(a)$. Therefore, the probability of choosing the action $a \in A$ after some history $h = ws$ (where $w \in (SA)^*$ and $s = \mathsf{last}(h)$) is given by the sum, for each memory state $m \in M$, of the probability that m was reached after $\mathcal{M}$ processes w, multiplied by $\alpha_{\mathsf{next}}(m, s)(a)$.

To provide a formal definition of the strategy induced by $\mathcal{M}$, we must first describe the distribution over memory states after $\mathcal{M}$ processes elements of $(SA)^*$. We formally define this distribution inductively. Details on how to derive the formulae for the update of these distributions, which use conditional probabilities, are presented in the full paper [31].

The distribution μ_ε over memory states after reading the empty word ε is by definition μ_{init}. Assume inductively we know the distribution μ_w for $w = s_0 a_0 \dots s_{k-1} a_{k-1}$ and let us explain how one derives $\mu_{ws_ka_k}$ from μ_w for any state $s_k \in \mathsf{supp}(\delta(s_{k-1}, a_{k-1}))$ and for any action $a_k \in A(s_k)$.

If $s_k \in S_{3-i}$, i.e., s_k is not controlled by the owner of the strategy, the action a_k does not introduce any conditions on the current memory state. Therefore, we set, for any memory state $m \in M$,

$$\mu_{ws_ka_k}(m) = \sum_{m' \in M} \mu_w(m') \cdot \alpha_{\mathsf{up}}(m', s_k, a_k)(m),$$

which consists in checking for each predecessor state m', what the probability of moving to memory state m is and weighing the sum by the probability of being in m'.

If $s_k \in S_i$, i.e., s_k is controlled by the owner of the strategy, then the choice of an action conditions what the predecessor memory states could be. If we have, for all memory states $m' \in M$ such that $\mu_w(m') > 0$, that $\alpha_{\mathsf{next}}(m', s_k)(a_k) = 0$, then the action a_k is actually never chosen. In this case, to ensure a complete definition, we perform an update as in the previous case. Otherwise, we condition updates on the likelihood of being in a memory state knowing that the action a_k was chosen. We define, for any memory state $m \in M$,

$$\mu_{ws_ka_k}(m) = \frac{\sum_{m' \in M} \mu_w(m') \cdot \alpha_{\mathsf{up}}(m', s_k, a_k)(m) \cdot \alpha_{\mathsf{next}}(m', s_k)(a_k)}{\sum_{m' \in M} \mu_w(m') \cdot \alpha_{\mathsf{next}}(m', s_k)(a_k)}.$$

This quotient is not well-defined whenever for all $m' \in \mathsf{supp}(\mu_w)$, $\alpha_{\mathsf{next}}(m', s_k)(a_k) = 0$, justifying the distinction above.

Using these distributions, we formally define the strategy $\sigma_i^{\mathcal{M}}$ induced by the Mealy machine $\mathcal{M} = (M, \mu_{\mathsf{init}}, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$ as the strategy $\sigma_i^{\mathcal{M}} \colon \mathsf{Hist}_i(\mathcal{G}) \to \mathcal{D}(A)$ such that for all histories $h = ws$, for all actions $a \in A(s)$, $\sigma_i^{\mathcal{M}}(h)(a) = \sum_{m \in M} \mu_w(m) \cdot \alpha_{\mathsf{next}}(m, s)(a)$.

Classifying finite-memory strategies. In the sequel, we investigate the relationships between different classes of finite-memory strategies with respect to expressive power. We classify finite-memory strategies following the type of stochastic Mealy machines that can induce them. We introduce a concise notation for each class: we use three-letter acronyms of the form XXX with X $\in$ {D,R}, where the letters, in order, refer to the initialisation, outputs and updates of the Mealy machines, with D and R respectively denoting deterministic and randomised components. For instance, we will write RRD to denote the class of Mealy machines that have randomised initialisation and outputs, but deterministic updates. We also apply this terminology to FM strategies: we will say that an FM strategy is in the class XXX – i.e., it is an XXX strategy – if it is induced by an XXX Mealy machine.

Moreover, in the remainder of the paper, we will abusively identify Mealy machines and their induced FM strategies. For instance, we will say that $\mathcal{M}$ is an XXX strategy to mean that $\mathcal{M}$ is an XXX Mealy machine (thus inducing an XXX strategy). As a by-product of this identification, we apply the terminology introduced previously for strategies to Mealy machines, without explicitly referring to the strategy they induce. For instance, we may say a history is consistent with some Mealy machine, or that two Mealy machines are outcome-equivalent. Let us note however that we will not use a Mealy machine in lieu of its induced strategy whenever we are interested in the strategy itself as a function. This choice lightens notations; the strategy induced by a Mealy machine need not be introduced unless it is required as a function.

We close this section by commenting on some of the classes, and discuss previous appearances in the literature, under different names. Pure strategies use no randomisation: hence, the class DDD corresponds to pure FM strategies, which can be represented by Mealy machines that do not rely on randomisation.

Strategies in the class DRD have been referred to as *behavioural* FM strategies in [21]. The name comes from the randomised outputs, reminiscent of behavioural strategies that output a distribution over actions after a history. We note that stochastic Mealy machines that induce DRD strategies are such that their distributions over memory states are Dirac due to the deterministic initialisation and updates.

Similarly, RDD strategies have been referred to as *mixed* FM strategies [21]. The general definition of a mixed strategy is a distribution over pure strategies: under a mixed strategy, a player randomly selects a pure strategy at the start of a play and plays according to it for the whole play. RDD strategies are similar in the way that the random initialisation can be viewed as randomly selecting some DDD strategy (i.e., a pure FM strategy) among a *finite* selection of such strategies.

The elements of RRR, the broadest class of FM strategies, have been referred to as general FM strategies [21] and stochastic-update FM strategies [8, 18]. The latter name highlights the random nature of updates and insists on the difference with models that rely on deterministic updates, more common in the literature.

3 Taxonomy of finite-memory strategies

In this section, we comment on the relationships between the classes of finite-memory strategies in terms of expressiveness. We say that a class $\mathcal{C}_1$ of FM strategies is no less expressive than a class $\mathcal{C}_2$ if for all games $\mathcal{G}$, for all FM strategies $\mathcal{M} \in \mathcal{C}_2$ in $\mathcal{G}$, one can find some FM strategy $\mathcal{M}' \in \mathcal{C}_1$ of $\mathcal{G}$ such that $\mathcal{M}$ and $\mathcal{M}'$ are outcome-equivalent strategies. For the sake of brevity, we will say that $\mathcal{C}_2$ is included in $\mathcal{C}_1$, and write $\mathcal{C}_2 \subseteq \mathcal{C}_1$.

Figure 1.1 summarises our results. In terms of set inclusion, each line in the figure indicates that the class below is strictly included in the class above. Each line is decorated with a reference to the relevant results. The strictness results hold in one-player deterministic games. In particular, there are no collapses in the diagram in either two-player deterministic games and in Markov decision processes.

Some inclusions follow purely from syntactic arguments. For instance, the inclusion DRD $\subseteq$ RRD follows from the fact that RRD Mealy machines have more randomisation power than DRD ones. The inclusions RDD $\subseteq$ DRD, RRR $\subseteq$ DRR and RRR $\subseteq$ RDR, which do not follow from such arguments, are covered in Section 4.

Pure strategies are strictly less expressive than any other class of FM strategies; pure strategies cannot induce any non-Dirac distributions on plays in deterministic one-player games. The strictness of the other inclusions is presented in Section 5.

We close this section by comparing our results with Kuhn's theorem, which asserts that the classes of behavioural strategies and mixed strategies in games of perfect recall share the same expressiveness. Games of perfect recall have two traits: players never forget the sequence of histories controlled by them that have taken place and they can see their own actions. In particular, stochastic games of perfect information are a special case of games of perfect recall. Recall that mixed strategies are distributions over pure strategies. We comment briefly on the techniques used in the proof of Kuhn's theorem, and compare them with the finite-memory setting. Let us fix a game $\mathcal{G} = (S_1, S_2, A, \delta)$.

On the one hand, the emulation of mixed strategies with behavioural strategies is performed as follows. Let p_i be a mixed strategy of $\mathcal{P}_i$, i.e., a distribution over pure strategies of $\mathcal{G}$. An outcome-equivalent behavioural strategy σ_i is constructed such that, for all histories $h \in \mathsf{Hist}_i(\mathcal{G})$ and actions $a \in A(\mathsf{last}(h))$,

$$\sigma_i(h)(a) = \frac{p_i(\{\tau_i \text{ pure strategy} \mid \tau_i \text{ consistent with } h \text{ and } \tau_i(h) = a\})}{p_i(\{\tau_i \text{ pure strategy} \mid \tau_i \text{ consistent with } h\})}.$$

In the finite-memory case, similar ideas can be used to show that RDD $\subseteq$ DRD. In the proof of Theorem 2, from some RDD strategy (i.e., a so-called mixed FM strategy), we construct a DRD strategy (i.e., a so-called behavioural FM strategy) that keeps track of the finitely many pure FM strategies that the RDD strategy mixes and that are consistent with the current history. An adaption of the quotient above is used in the next-move function of the DRD strategy.

On the other hand, the emulation of behavioural strategies by mixed strategies exploits the fact that mixed strategies may randomise over *infinite* sets. In a finite-memory setting, the same techniques cannot be applied. As a consequence, the class of RDD strategies is strictly included in the class of DRD strategies. In a certain sense, one could say that Kuhn's theorem only partially holds in the case of FM strategies.

4 Non-trivial inclusions

This section covers the non-trivial inclusions that are asserted in the lattice of Figure 1.1. The structure of this section is as follows. Section 4.1 covers the inclusion RDD $\subseteq$ DRD. The inclusion RRR $\subseteq$ DRR is presented in Section 4.2. Finally, we close this section with the inclusion RRR $\subseteq$ RDR in Section 4.3. Full proofs and details of this section are presented in the full paper [31].

4.1 Simulating RDD strategies with DRD ones

We argue that for all RDD strategies in any game, one can find some outcome-equivalent DRD strategy (Theorem 2). The converse inclusion is not true; this discussion is relegated to Section 5.1. The outlined construction yields a DRD strategy that has a state space of size exponential in the size of the state space of the original RDD strategy; we complement Theorem 2 by proving that there are some RDD strategies for which this exponential blow-up in the number of states is necessary for any outcome-equivalent DRD strategy (Lemma 3). We argue that this blow-up is unavoidable in both deterministic two-player games and Markov decision processes.

Let $\mathcal{G} = (S_1, S_2, A, \delta)$ be a game. Fix an RDD strategy $\mathcal{M} = (M, \mu_{\mathsf{init}}, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$ of $\mathcal{P}_i$. Let us sketch how to emulate $\mathcal{M}$ with a DRD strategy $\mathcal{B} = (B, b_{\mathsf{init}}, \beta_{\mathsf{next}}, \beta_{\mathsf{up}})$ built with a subset construction-like approach. The memory states of $\mathcal{B}$ are functions $f \colon \mathsf{supp}(\mu_{\mathsf{init}}) \to M \cup \{\bot\}$. A memory state f is interpreted as follows. For all initial memory states $m_0 \in \mathsf{supp}(\mu_{\mathsf{init}})$, we have $f(m_0) = \bot$ if the history seen up to now is not consistent with the pure FM strategy $(M, m_0, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$, and otherwise $f(m_0)$ is the memory state reached in the same pure FM strategy after processing the current history. Updates are naturally derived from these semantics.

Using this state space and update scheme, we can compute the likelihood of each memory state of the mixed FM strategy $\mathcal{M}$ after some sequence $w \in (SA)^*$ has taken place. Indeed, we keep track of each initial memory state from which it was possible to be consistent with w, and, for each such initial memory state m_0, the memory state reached after w was processed starting in m_0. Therefore, this likelihood can be inferred from μ_{init}; the probability of $\mathcal{M}$ being in $m \in M$ after w has been processed is given by the (normalised) sum of the probability of each initial memory state $m_0 \in \mathsf{supp}(\mu_{\mathsf{init}})$ such that $f(m_0) = m$.

The definition of the next-move function of $\mathcal{B}$ is directly based on the distribution over states of $\mathcal{M}$ described in the previous paragraph, and ensures that the two strategies select actions with the same probabilities at any given state. For any action $a \in A(s)$, the probability

of a being chosen in game state s and in memory state f is determined by the probability of $\mathcal{M}$ being in some memory state m such that $\alpha_{\mathsf{next}}(m, s) = a$, where this probability is inferred from f. It follows from Lemma 1 that $\mathcal{M}$ and $\mathcal{B}$ are outcome-equivalent.

Intuitively, we postpone the initial randomisation and instead randomise at each step in an attempt of replicating the initial distribution in the long run.

▶ **Theorem 2.** *Let $\mathcal{G} = (S_1, S_2, A, \delta)$ be a game. Let $\mathcal{M} = (M, \mu_{\mathsf{init}}, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$ be an RDD strategy of $\mathcal{P}_i$. There exists a DRD strategy $\mathcal{B} = (B, b_{\mathsf{init}}, \beta_{\mathsf{next}}, \beta_{\mathsf{up}})$ such that $\mathcal{B}$ and $\mathcal{M}$ are outcome-equivalent.*

The DRD strategy outlined prior to Theorem 2 leads to an exponential blow-up of the memory state space. For an RDD strategy $\mathcal{M} = (M, \mu_{\mathsf{init}}, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$, we have described an outcome-equivalent DRD strategy with a state space consisting of functions $\mathsf{supp}(\mu_{\mathsf{init}}) \to M \cup \{\bot\}$, therefore with a state space of size $(|M| + 1)^{|\mathsf{supp}(\mu_{\mathsf{init}})|}$.

An exponential blow-up in the number of initial memory states cannot be avoided in general. Intuitively, due to the other player's actions (or stochastic transitions in Markov decision processes), it may be the case that for each subset of the initial states, there are histories ending in some fixed state s that are consistent with the pure FM strategies starting in these initial states. If these pure strategies each prescribe a different action in s, then there must be at least one memory state per non-empty subset of initial states in an outcome-equivalent DRD Mealy machine; this can be deduced by counting the number of necessary next-move functions $\alpha_{\mathsf{next}}(\cdot, s)$. We obtain the following result.

▶ **Lemma 3.** *For all $n \in \mathbb{N}$, there exists a two-player deterministic game (respectively a Markov decision process) $\mathcal{G}_n$ with $n+2$ states, $4n+2$ transitions, $n+1$ actions, and an RDD strategy $\mathcal{M}_n$ of $\mathcal{P}_1$ with n states such that any outcome-equivalent DRD strategy must have at least $2^n - 1$ states.*

4.2 Simulating RRR strategies with DRR ones

We establish that DRR strategies are as expressive as RRR strategies, i.e., randomness in the initialisation can be removed. We outline the ideas behind the construction of a DRR strategy that is outcome-equivalent to a given RRR strategy. The rough idea behind the construction is to simulate the behaviour of the RRR strategy at the start of the play using a new initial memory state and then move back into the RRR strategy we simulate.

We substitute the random selection of an initial memory element in two stages. To ensure the first action is selected in the same way under both the supplied strategy and the strategy we construct, we rely on the randomised outputs. The probability of selecting an action a in a given state s of the game in our new initial memory state is given as the sum of selecting action a in state s in each memory state m weighed by the initial probability of m.

We then leverage the stochastic updates to behave as though we had been using the supplied FM strategy from the start. If the first game state was controlled by the player who does not own the strategy, the probability of moving into a memory state m is also described by a weighted sum similar in spirit to the case of the first action (albeit by considering the update function in place of the next-move function). Whenever the owner of the strategy controls the first state of the game, the chosen action conditions which possible initial memory states we could have found ourselves in. The reasoning in this case is similar to the one for the update of the distribution over memory states (denoted by μ_w in Section 2) after processing some sequence in $(SA)^*$. In light of the above and Lemma 1, we obtain the following expressiveness result.

▶ **Theorem 4.** *Let $\mathcal{G} = (S_1, S_2, A, \delta)$ be a game. Let $\mathcal{M} = (M, \mu_{\mathsf{init}}, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$ be an RRR strategy owned by $\mathcal{P}_i$. There exists a DRR strategy $\mathcal{B} = (B, b_{\mathsf{init}}, \beta_{\mathsf{next}}, \beta_{\mathsf{up}})$ such that $\mathcal{B}$ and $\mathcal{M}$ are outcome-equivalent, and such that $|B| = |M| + 1$.*

4.3 Simulating RRR strategies with RDR ones

We are concerned with the simulation of RRR strategies by RDR strategies, i.e., with substituting randomised outputs with deterministic outputs. The idea behind the removal of randomisation in outputs is to simulate said randomisation by means of both stochastic initialisation and updates. These are used to preemptively perform the random selection of an action, simultaneously with the selection of an initial or successor memory state.

Let $\mathcal{G} = (S_1, S_2, A, \delta)$ be a stochastic game and let $\mathcal{M} = (M, \mu_{\mathsf{init}}, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$ be an RRR strategy of $\mathcal{P}_i$. We construct an RDR strategy $\mathcal{B} = (B, \beta_{\mathsf{init}}, \beta_{\mathsf{next}}, \beta_{\mathsf{up}})$ that is outcome-equivalent to $\mathcal{M}$ and such that $|B| \leq |M| \cdot |S| \cdot |A|$. The state space of $\mathcal{B}$ consists of pairs (m, σ_i) where $m \in M$ and σ_i is a pure memoryless strategy of $\mathcal{P}_i$. To achieve our bound on the size of B, we cannot take all pure memoryless strategies of $\mathcal{P}_i$. To illustrate how we perform the selection of these pure memoryless strategies, we provide a simple example of the construction on a DRD strategy (which is a special case of RRR strategies) with a single memory state (i.e., a memoryless randomised strategy).

▶ **Example 5.** We consider a game $\mathcal{G} = (S_1, S_2, A, \delta)$ where $S_1 = \{s_1, s_2, s_3\}$, $S_2 = \emptyset$, $A = \{a_1, a_2, a_3\}$ and all actions are enabled in all states. We need not specify δ exactly for our purposes. For our construction, we fix an order on the actions of $\mathcal{G}$: $a_1 < a_2 < a_3$.

Let $\mathcal{M} = (\{m\}, m, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$ be the DRD strategy such that $\alpha_{\mathsf{next}}(m, s_1)$ and $\alpha_{\mathsf{next}}(m, s_2)$ are uniform distributions over $\{a_1, a_2\}$ and A respectively and $\alpha_{\mathsf{next}}(m, s_3)$ is defined by $\alpha_{\mathsf{next}}(m, s_3)(a_1) = \frac{1}{3}$, $\alpha_{\mathsf{next}}(m, s_3)(a_2) = \frac{1}{6}$ and $\alpha_{\mathsf{next}}(m, s_3)(a_3) = \frac{1}{2}$.

Figure 4.1 illustrates the probability of each action being chosen in each state as the length of a segment. Let us write $0 = x_1 < x_2 < x_3 < x_4 < x_5 = 1$ for all of the endpoints of the segments appearing in the illustration. For each index $k \in \{1, \ldots, 4\}$, we define a pure memoryless strategy σ_k that assigns to each state the action lying in the segment above it in the figure. For instance, σ_2 is such that $\sigma_2(s_1) = a_1$ and $\sigma_2(s_2) = \sigma_2(s_3) = a_2$. Furthermore, for all $k \in \{1, \ldots, 4\}$, the length $x_{k+1} - x_k$ of its corresponding interval denotes the probability of the strategy being chosen during stochastic updates.

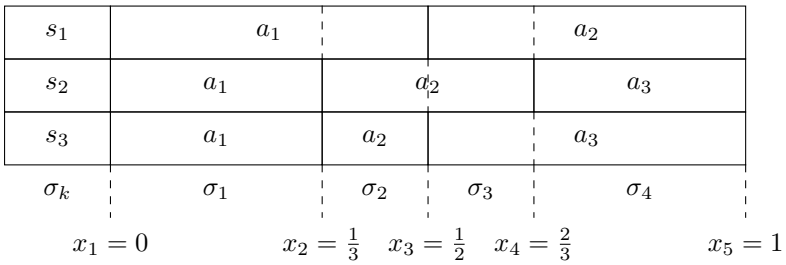


Figure 4.1 Representation of cumulative probability of actions under strategy $\mathcal{M}$ and derived memoryless strategies.

We construct an RDR strategy $\mathcal{B} = (B, \beta_{\mathsf{init}}, \beta_{\mathsf{next}}, \beta_{\mathsf{up}})$ that is outcome-equivalent to $\mathcal{M}$ in the following way. We let $B = \{m\} \times \{\sigma_1, \sigma_2, \sigma_3, \sigma_4\}$. The initial distribution is given by $\beta_{\mathsf{init}}(m, \sigma_k) = x_{k+1} - x_k$, i.e., the probability of σ_k in the illustration. We set, for any $j, k \in \{1, \ldots, 4\}$, $s \in S$ and $a \in A$, $\beta_{\mathsf{up}}((m, \sigma_k), s, a)((m, \sigma_j)) = x_{j+1} - x_j$. Finally, we let $\beta_{\mathsf{next}}((m, \sigma_k), s) = \sigma_k(s)$ for all $k \in \{1, \ldots, 4\}$ and $s \in S$.

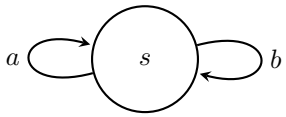


Figure 5.1 A (one-player) game with a single state and two actions.

The argument for the outcome-equivalence of $\mathcal{B}$ and $\mathcal{M}$ is the following; for any state $s \in S_1$, the probability of moving into a memory state (m, σ_k) such that $\sigma_k(s) = a$ is by construction the probability $\alpha_{\mathsf{next}}(m, s)$. ◀

In the previous example, we had a unique memory state m and we defined some memoryless strategies from the next-move function partially evaluated in this state (i.e., from $\alpha_{\mathsf{next}}(m, \cdot)$). In general, each memory state may have a different partially evaluated next-move function, and therefore we must define some memoryless strategies for each individual memory state. For each memory state, we can bound the number of derived memoryless strategies by $|S_i| \cdot |A|$; we look at cumulative probabilities over actions (of which there are at most $|A|$) for each state of $\mathcal{P}_i$. This explains our announced bound on $|B|$.

Furthermore, in general, the memory update function is not trivial. Generalising the construction above can be done in a straightforward manner to handle updates. Intuitively, the probability to move to some memory state of the form (m, σ) is defined by the probability of moving into m multiplied by the probability of σ (in the sense of Figure 4.1).

We now formally state our result in the general setting.

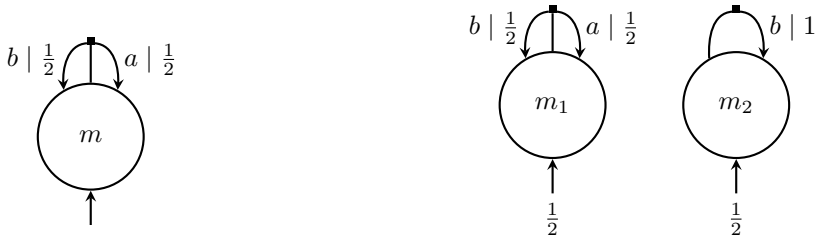
▶ **Theorem 6.** *Let $\mathcal{G} = (S_1, S_2, A, \delta)$ be a game. Let $\mathcal{M} = (M, \mu_{\mathsf{init}}, \alpha_{\mathsf{next}}, \alpha_{\mathsf{up}})$ be an RRR strategy owned by $\mathcal{P}_i$. There exists an RDR strategy $\mathcal{B} = (B, \beta_{\mathsf{init}}, \beta_{\mathsf{next}}, \beta_{\mathsf{up}})$ such that $\mathcal{B}$ and $\mathcal{M}$ are outcome-equivalent, and such that $|B| \leq |M| \cdot |S_i| \cdot |A|$.*

5 Strictness of inclusions

We now discuss the strictness of inclusions in the lattice of Figure 1.1. Section 5.1 complements the previous Section 4.1 and presents a DRD strategy that has no outcome-equivalent RDD counterpart. The strict inclusion of the class DRD in the class of RRD strategies is covered in Section 5.2. Finally, we provide the necessary results to establish that the class DDR is incomparable to the classes of RDD, DRD and RRD strategies in Section 5.3. Technical details are presented in the full paper [31].

All strictness results hold in one-player deterministic games with a single state and two actions. This is one of the simplest possible settings to show that strategy classes are distinct. Indeed, in a game with a single state and a single action, the only strategy is to always play the unique action, and therefore all strategy classes collapse into one. For the entirety of this section, we let $\mathcal{G}$ denote the game depicted in Figure 5.1, and only consider strategies of $\mathcal{G}$ in the upcoming statements.

We illustrate FM strategies witnessing the strictness of inclusions asserted in the lattice of Figure 1.1 in Figures 5.2 and 5.3. The Mealy machines are interpreted as follows. Edges that exit memory states read a game state (omitted in these figures due to s being the sole involved game state) and split into edges labelled by an action and a probability of this action being played, e.g., for $c \in \{a, b\}$ and $p \in [0, 1]$, the notation $c \mid p$ indicates that the probability of playing action c in the current memory state is p. In Figure 5.3, the edges are further split after the choice of an action for randomised updates. The edge labels following this second split represent the probabilities of stochastic updates. This second split is omitted whenever an update is deterministic.



(a) RDD $\subsetneq$ DRD. **(b)** DRD $\subsetneq$ RRD.

Figure 5.2 Depictions of Mealy machines witnessing the strictness of two inclusions asserted in Figure 1.1. For the sake of readability, we do not label transitions by s as it is the sole state the Mealy machines can read in $\mathcal{G}$.

5.1 RDD strategies are strictly less expressive than DRD ones

We argue that there exists a DRD strategy that cannot be emulated by any RDD strategy. Let us first explain some intuition behind this statement. Intuitively, an RDD strategy can only randomise once at the start between a finite number of pure FM (DDD) strategies. After this initial randomisation, the sequence of actions prescribed by the RDD strategy is fixed relative to the play in progress. Any DRD strategy that chooses an action randomly at each step, such as the strategy depicted in Figure 5.2a, i.e., the strategy playing actions a and b with uniform probability at each step in $\mathcal{G}$, cannot be reproduced by an RDD strategy. Indeed, this randomisation generates an infinite number of patterns of actions. These patterns cannot all be captured by an RDD strategy due to the fact that its initial randomisation is over a finite set.

▶ **Lemma 7.** *There exists a DRD strategy of $\mathcal{P}_1$ such that there is no outcome-equivalent RDD strategy.*

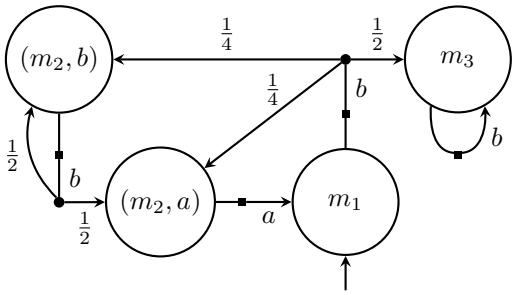
5.2 DRD strategies are strictly less expressive than RRD ones

We argue that there exists an RRD strategy that has no outcome-equivalent DRD strategy. The example we provide is based on results of [21, 22]; the authors of [21] illustrate that some behaviour proven to not be achievable by DRD strategies in concurrent reachability games by [22] can be achieved using RRD strategies.

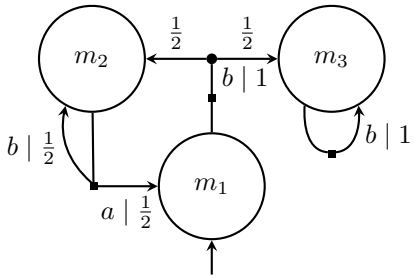
Consider the Mealy machine depicted in Figure 5.2b. The main idea underlying its induced strategy is the following. This strategy attempts the action a at all steps with a positive probability due to memory state m_1. It also has a positive probability of never playing a due to memory state m_2.

This behaviour cannot be achieved with a DRD strategy. The distribution over memory states of a DRD strategy following a history is a Dirac distribution due to the deterministic initialisation and deterministic updates. It follows that DRD strategies suggest actions with probabilities given directly by the next-move function. In particular, if an action is attempted at each round by a DRD strategy, then there exists a positive lower bound on the probability of the action being chosen (as there are finitely many memory states), therefore the action is eventually selected almost-surely. It follows that there is no DRD strategy that is outcome-equivalent to the strategy depicted in Figure 5.2b.

▶ **Lemma 8.** *There exists an RRD strategy of $\mathcal{P}_1$ such that there is no outcome-equivalent DRD strategy.*



(a) A DDR strategy witnessing DDR $\not\subseteq$ RRD.



(b) An outcome-equivalent RRR strategy with fewer states.

Figure 5.3 Outcome-equivalent strategies witnessing the non-inclusion DDR $\not\subseteq$ RRD. For the sake of readability, we do not label transitions by s as it is the sole state the Mealy machines can read in $\mathcal{G}$. We omit the probability of actions in Figure 5.3a as outputs are deterministic.

5.3 RRD and DDR strategies are incomparable

We argue that the classes RRD and DDR of finite-memory strategies are incomparable. While we have shown that RDR and DRR strategies are as powerful as RRR strategies, DDR strategies are not because they lack the ability to provide a random output at the first step of a game. Due to this trait, one can even construct some RDD strategy that cannot be emulated by any DDR strategy; any strategy that randomises between two pure FM strategies prescribing action a and action b respectively at the first turn in $\mathcal{G}$ has no outcome-equivalent DDR strategy. The following result follows immediately.

▶ **Lemma 9.** *There exists an RDD strategy of $\mathcal{P}_1$ such that there is no outcome-equivalent DDR strategy.*

On the other hand, one can construct a DDR strategy that has no outcome-equivalent RRD strategy. For instance, the DDR strategy that is depicted in Figure 5.3a has no outcome-equivalent RRD strategy. For ease of analysis, we illustrate in Figure 5.3b a DRR strategy with fewer states that is outcome-equivalent to the Mealy machine depicted in Figure 5.3a. Note that the DDR strategy of Figure 5.3a can be obtained by applying the construction of Theorem 6 to Figure 5.3b.

Intuitively, these strategies have a non-zero probability of never using action a after any history, while they have a positive probability of using action a at any time besides the first round and right after a use of the action a. The behaviour described above cannot be reproduced by an RRD strategy. There are two reasons to this.

First, along any play consistent with an RRD strategy, the support of the distribution over memory states cannot increase in size. Because of deterministic updates, the probability carried by a memory state m can only be transferred to at most one other state, and may be lost if the used action cannot be used while in m. This is not the case for strategies that have stochastic updates, such as those of Figure 5.3.

Second, one can force situations in which the size of the support of the distribution over memory states of an RRD strategy must decrease. If after a given history h, the action a has a positive probability of never being used despite being assigned a positive probability at each round after h, then at some point there must be some memory state of the RRD strategy that has positive probability and that assigns (via the next-move function) probability zero to action a. For instance, this is the case from the start with the RRD strategy depicted in

Figure 5.2b. Intuitively, in general, if at all times all memory states in the support of the distribution over memory states after the current history assign a positive probability to action a, the probability of using a at each round after h would be bounded from below by the smallest positive probability assigned to a by the next-move function. Therefore a would eventually be played almost-surely assuming h has taken place, contradicting the fact that there was a positive probability of never using action a after h. By using action a at a point in which some memory state in the support of the distribution over memory states assigns probability zero to a, the size of the support of the strategy decreases.

By design of our DDR strategy, if one assumes the existence of an outcome-equivalent RRD strategy, then it is possible to construct a play along which the size of the support of the distribution over memory states of the RRD strategy decreases between two consecutive steps infinitely often. Because this size cannot increase along a play, this is not possible, i.e., there is no such RRD strategy. We obtain the following lemma.

▶ **Lemma 10.** *There exists a DDR strategy of $\mathcal{P}_1$ such that there is no outcome-equivalent RRD strategy.*

6 Extensions

In the following, we discuss settings beyond two-player games with perfect information to which our results carry over.

Multi-player games. In this work, we have considered two-player games. Our result also extend to games with more than two players. The definition of outcome-equivalence naturally extends to multi-player games; instead of quantifying universally over strategies of the other player as is done in the two-player setting, one quantifies universally over strategy profiles of all other players when defining outcome-equivalence with more players.

In practice, when studying the outcome-equivalence of strategies of some given player $\mathcal{P}_i$, there being more than one other player is no different than having a fictitious single other player obtained as a coalition of all players besides $\mathcal{P}_i$. Therefore, all of our results carry over to the multi-player setting directly. Furthermore, due to outcome-equivalence being a criterion that depends solely on the studied player, as highlighted by the equivalent formulation given in Lemma 1, the way players are arranged in coalitions does not affect our taxonomy in multi-player games with coalitions.

Imperfect information. Our presentation assumes perfect information; the players are fully informed of the sequence of witnessed states and used actions. Our results also hold in a context of partial observation with observable actions. A formalisation of the following is provided in the full paper [31].

First, it is not necessary to have full knowledge of the states; an observation suffices. Intuitively, the Mealy machines are formally agnostic to the nature of the inputs, therefore states can be substituted with observations while preserving correctness of all of our constructions.

Second, we need not be capable to observe the actions of the other player for our constructions. However, it is required to observe the actions of the player we consider; we exploit this in the memory update functions in the constructions relevant to Theorems 2 and 4. In addition to the visibility of actions, these constructions need also the capability of distinguishing from a sequence of observations whose turn it was at each step; this can be achieved either by requiring that the two players have disjoint action sets or by encoding in the state observations to whom the states belong.

References

1 Robert J . Aumann. *28. Mixed and Behavior Strategies in Infinite Extensive Games*, pages 627–650. Princeton University Press, 2016. doi:doi:10.1515/9781400882014-029.

2 Raphaël Berthon, Mickael Randour, and Jean-François Raskin. Threshold constraints with guarantees for parity objectives in Markov decision processes. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 121:1–121:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.121.

3 Roderick Bloem, Krishnendu Chatterjee, and Barbara Jobstmann. Graph games and reactive synthesis. In Edmund M. Clarke, Thomas A. Henzinger, Helmut Veith, and Roderick Bloem, editors, *Handbook of Model Checking*, pages 921–962. Springer, 2018. doi:10.1007/978-3-319-10575-8_27.

4 Patricia Bouyer, Stéphane Le Roux, Youssouf Oualhadj, Mickael Randour, and Pierre Vandenhove. Games where you can play optimally with arena-independent finite memory. In Igor Konnov and Laura Kovács, editors, *31st International Conference on Concurrency Theory, CONCUR 2020, September 1-4, 2020, Vienna, Austria (Virtual Conference)*, volume 171 of *LIPIcs*, pages 24:1–24:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CONCUR.2020.24.

5 Patricia Bouyer, Nicolas Markey, Mickael Randour, Kim G. Larsen, and Simon Laursen. Average-energy games. *Acta Inf.*, 55(2):91–127, 2018. doi:10.1007/s00236-016-0274-1.

6 Patricia Bouyer, Youssouf Oualhadj, Mickael Randour, and Pierre Vandenhove. Arena-independent finite-memory determinacy in stochastic games. In Serge Haddad and Daniele Varacca, editors, *32nd International Conference on Concurrency Theory, CONCUR 2021, August 24-27, 2021, Virtual Conference*, volume 203 of *LIPIcs*, pages 26:1–26:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CONCUR.2021.26.

7 Patricia Bouyer, Mickael Randour, and Pierre Vandenhove. Characterizing omega-regularity through finite-memory determinacy of games on infinite graphs. In Petra Berenbrink and Benjamin Monmege, editors, *39th International Symposium on Theoretical Aspects of Computer Science, STACS 2022, March 15-18, 2022, Marseille, France (Virtual Conference)*, volume 219 of *LIPIcs*, pages 16:1–16:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.STACS.2022.16.

8 Tomás Brázdil, Václav Brozek, Krishnendu Chatterjee, Vojtech Forejt, and Antonín Kucera. Markov decision processes with multiple long-run average objectives. *Log. Methods Comput. Sci.*, 10(1), 2014. doi:10.2168/LMCS-10(1:13)2014.

9 Romain Brenguier, Lorenzo Clemente, Paul Hunter, Guillermo A. Pérez, Mickael Randour, Jean-François Raskin, Ocan Sankur, and Mathieu Sassolas. Non-zero sum games for reactive synthesis. In Adrian-Horia Dediu, Jan Janousek, Carlos Martín-Vide, and Bianca Truthe, editors, *Language and Automata Theory and Applications – 10th International Conference, LATA 2016, Prague, Czech Republic, March 14-18, 2016, Proceedings*, volume 9618 of *Lecture Notes in Computer Science*, pages 3–23. Springer, 2016. doi:10.1007/978-3-319-30000-9_1.

10 Thomas Brihaye, Florent Delgrange, Youssouf Oualhadj, and Mickael Randour. Life is random, time is not: Markov decision processes with window objectives. In Wan Fokkink and Rob van Glabbeek, editors, *30th International Conference on Concurrency Theory, CONCUR 2019, August 27-30, 2019, Amsterdam, the Netherlands*, volume 140 of *LIPIcs*, pages 8:1–8:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CONCUR.2019.8.

11 Véronique Bruyère, Emmanuel Filiot, Mickael Randour, and Jean-François Raskin. Meet your expectations with guarantees: Beyond worst-case synthesis in quantitative games. *Inf. Comput.*, 254:259–295, 2017. doi:10.1016/j.ic.2016.10.011.

12 Véronique Bruyère, Quentin Hautem, and Mickael Randour. Window parity games: an alternative approach toward parity games with time bounds. In Domenico Cantone and Giorgio Delzanno, editors, *Proceedings of the Seventh International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2016, Catania, Italy, 14-16 September 2016*, volume 226 of *EPTCS*, pages 135–148, 2016. doi:10.4204/EPTCS.226.10.

13 Véronique Bruyère, Quentin Hautem, Mickael Randour, and Jean-François Raskin. Energy mean-payoff games. In Wan Fokkink and Rob van Glabbeek, editors, *30th International Conference on Concurrency Theory, CONCUR 2019, August 27-30, 2019, Amsterdam, the Netherlands*, volume 140 of *LIPIcs*, pages 21:1–21:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CONCUR.2019.21.

14 Krishnendu Chatterjee, Luca de Alfaro, and Thomas A. Henzinger. Trading memory for randomness. In *1st International Conference on Quantitative Evaluation of Systems (QEST 2004), 27-30 September 2004, Enschede, The Netherlands*, pages 206–217. IEEE Computer Society, 2004. doi:10.1109/QEST.2004.1348035.

15 Krishnendu Chatterjee and Laurent Doyen. Partial-observation stochastic games: How to win when belief fails. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25-28, 2012*, pages 175–184. IEEE Computer Society, 2012. doi:10.1109/LICS.2012.28.

16 Krishnendu Chatterjee, Laurent Doyen, Hugo Gimbert, and Thomas A. Henzinger. Randomness for free. *Inf. Comput.*, 245:3–16, 2015. doi:10.1016/j.ic.2015.06.003.

17 Krishnendu Chatterjee, Thomas A. Henzinger, and Vinayak S. Prabhu. Trading infinite memory for uniform randomness in timed games. In Magnus Egerstedt and Bud Mishra, editors, *Hybrid Systems: Computation and Control, 11th International Workshop, HSCC 2008, St. Louis, MO, USA, April 22-24, 2008. Proceedings*, volume 4981 of *Lecture Notes in Computer Science*, pages 87–100. Springer, 2008. doi:10.1007/978-3-540-78929-1_7.

18 Krishnendu Chatterjee, Zuzana Kretínská, and Jan Kretínský. Unifying two views on multiple mean-payoff objectives in Markov decision processes. *Log. Methods Comput. Sci.*, 13(2), 2017. doi:10.23638/LMCS-13(2:15)2017.

19 Krishnendu Chatterjee, Mickael Randour, and Jean-François Raskin. Strategy synthesis for multi-dimensional quantitative objectives. *Acta Inf.*, 51(3-4):129–163, 2014. doi:10.1007/s00236-013-0182-6.

20 Anne Condon. The complexity of stochastic games. *Inf. Comput.*, 96(2):203–224, 1992. doi:10.1016/0890-5401(92)90048-K.

21 Julien Cristau, Claire David, and Florian Horn. How do we remember the past in randomised strategies? In Angelo Montanari, Margherita Napoli, and Mimmo Parente, editors, *Proceedings First Symposium on Games, Automata, Logic, and Formal Verification, GANDALF 2010, Minori (Amalfi Coast), Italy, 17-18th June 2010*, volume 25 of *EPTCS*, pages 30–39, 2010. doi:10.4204/EPTCS.25.7.

22 Luca de Alfaro, Thomas A. Henzinger, and Orna Kupferman. Concurrent reachability games. *Theor. Comput. Sci.*, 386(3):188–217, 2007. doi:10.1016/j.tcs.2007.07.008.

23 Florent Delgrange, Joost-Pieter Katoen, Tim Quatmann, and Mickael Randour. Simple strategies in multi-objective mdps. In Armin Biere and David Parker, editors, *Tools and Algorithms for the Construction and Analysis of Systems – 26th International Conference, TACAS 2020, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2020, Dublin, Ireland, April 25-30, 2020, Proceedings, Part I*, volume 12078 of *Lecture Notes in Computer Science*, pages 346–364. Springer, 2020. doi:10.1007/978-3-030-45190-5_19.

24 Rick Durrett. *Probability: Theory and Examples*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 5th edition, 2019. doi:10.1017/9781108591034.

25 Andrzej Ehrenfeucht and Jan Mycielski. Positional strategies for mean payoff games. *Int. Journal of Game Theory*, 8(2):109–113, 1979.

26 E. Allen Emerson and Charanjit S. Jutla. The complexity of tree automata and logics of programs. In *FOCS*, pages 328–337. IEEE Computer Society, 1988. doi:10.1109/SFCS.1988.21949.

27 Hugo Gimbert and Wieslaw Zielonka. Games where you can play optimally without any memory. In Martín Abadi and Luca de Alfaro, editors, *CONCUR 2005 – Concurrency Theory, 16th International Conference, CONCUR 2005, San Francisco, CA, USA, August 23-26, 2005, Proceedings*, volume 3653 of *Lecture Notes in Computer Science*, pages 428–442. Springer, 2005. doi:10.1007/11539452_33.

28 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002. doi:10.1007/3-540-36387-4.

29 Florian Horn. Random fruits on the Zielonka tree. In Susanne Albers and Jean-Yves Marion, editors, *26th International Symposium on Theoretical Aspects of Computer Science, STACS 2009, February 26-28, 2009, Freiburg, Germany, Proceedings*, volume 3 of *LIPIcs*, pages 541–552. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Germany, 2009. doi:10.4230/LIPIcs.STACS.2009.1848.

30 Stéphane Le Roux, Arno Pauly, and Mickael Randour. Extending finite-memory determinacy by Boolean combination of winning conditions. In Sumit Ganguly and Paritosh K. Pandya, editors, *38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2018, December 11-13, 2018, Ahmedabad, India*, volume 122 of *LIPIcs*, pages 38:1–38:20. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.FSTTCS.2018.38.

31 James C. A. Main and Mickael Randour. Different strokes in randomised strategies: Revisiting Kuhn's theorem under finite-memory assumptions. *CoRR*, abs/2201.10825, 2022. arXiv:2201.10825.

32 A. Maitra and W. Sudderth. Stochastic games with Borel payoffs. In Abraham Neyman and Sylvain Sorin, editors, *Stochastic Games and Applications*, pages 367–373, Dordrecht, 2003. Springer Netherlands.

33 Benjamin Monmege, Julie Parreaux, and Pierre-Alain Reynier. Reaching your goal optimally by playing at random with no memory. In Igor Konnov and Laura Kovács, editors, *31st International Conference on Concurrency Theory, CONCUR 2020, September 1-4, 2020, Vienna, Austria (Virtual Conference)*, volume 171 of *LIPIcs*, pages 26:1–26:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CONCUR.2020.26.

34 Martin J. Osborne and Ariel Rubinstein. *A course in game theory*. The MIT Press, Cambridge, USA, 1994. electronic edition.

35 Mickael Randour. Automated synthesis of reliable and efficient systems through game theory: A case study. In *Proc. of ECCS 2012*, Springer Proceedings in Complexity XVII, pages 731–738. Springer, 2013. doi:10.1007/978-3-319-00395-5_90.

36 Mickael Randour, Jean-François Raskin, and Ocan Sankur. Percentile queries in multi-dimensional Markov decision processes. *Formal Methods Syst. Des.*, 50(2-3):207–248, 2017. doi:10.1007/s10703-016-0262-7.

37 L. S. Shapley. Stochastic games. *Proceedings of the National Academy of Sciences*, 39(10):1095–1100, 1953. doi:10.1073/pnas.39.10.1095.

38 Yaron Velner, Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, Alexander Moshe Rabinovich, and Jean-François Raskin. The complexity of multi-mean-payoff and multi-energy games. *Inf. Comput.*, 241:177–196, 2015. doi:10.1016/j.ic.2015.03.001.

39 Wieslaw Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theor. Comput. Sci.*, 200(1-2):135–183, 1998. doi:10.1016/S0304-3975(98)00009-7.