

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

James C. A. Main Mickael Randour

UMONS – Université de Mons and F.R.S.-FNRS, Belgium

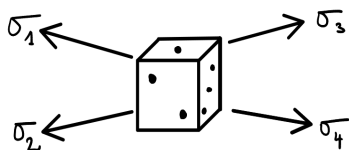
The logo for the University of Mons (UMONS) features the word "UMONS" in a bold, red, sans-serif font. The letter "U" is underlined with a horizontal red bar.

The logo for the National Research Fund (FNRS) features the lowercase letters "fnrs" in a bold, purple, sans-serif font. Below the letters, the tagline "LA LIBERTÉ DE CHERCHER" is written in a smaller, green, sans-serif font.

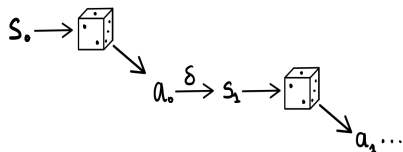
Journées du GT Vérif 2022 – July 11, 2022

Introduction

- In general, one can define **randomised strategies** in different ways.



Mixed strategies



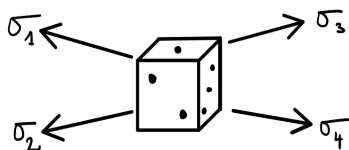
Behavioural strategies

- In general, these two classes of strategies are **not comparable**.

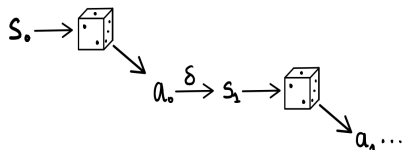
¹Aumann, "28. Mixed and Behavior Strategies in Infinite Extensive Games".

Introduction

- In general, one can define **randomised strategies** in different ways.



Mixed strategies



Behavioural strategies

- In general, these two classes of strategies are **not comparable**.
- Kuhn's theorem [Aum64]¹: in **games of perfect recall** any mixed strategy has an equivalent behavioural strategy and vice-versa.

Focus of this talk

A Kuhn-inspired classification of **finite-memory strategies**.

¹Aumann, "28. Mixed and Behavior Strategies in Infinite Extensive Games".

Table of contents

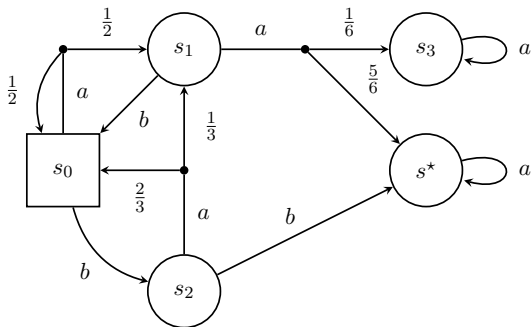
- 1 Games and strategies
- 2 Finite-memory strategies
- 3 Inclusions between classes
- 4 Differences between classes
- 5 Extending the classification

Table of contents

- 1 Games and strategies
- 2 Finite-memory strategies
- 3 Inclusions between classes
- 4 Differences between classes
- 5 Extending the classification

Setting: finite stochastic games

We consider two-player **stochastic games**.



Essential characteristics

- **Finite** state space $S = S_1 \uplus S_2$ and action space A .
- Probabilistic transition function $\delta: S \times A \rightarrow \mathcal{D}(S)$.
- No deadlocks.

Strategies

Definition

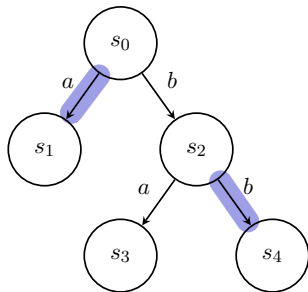
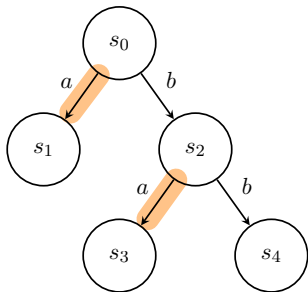
A **strategy** of \mathcal{P}_i is a function $\sigma_i: (SA)^*S_i \rightarrow \mathcal{D}(A)$.

Strategies

Definition

A **strategy** of \mathcal{P}_i is a function $\sigma_i: (SA)^*S_i \rightarrow \mathcal{D}(A)$.

- We compare strategies **independently of any objective or payoff**.
- **Equality is too restrictive**: two different strategies may induce the **same behaviour** in practice.

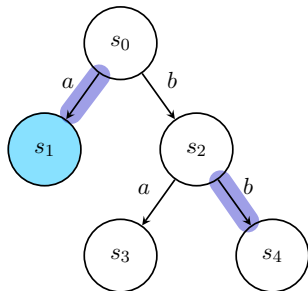
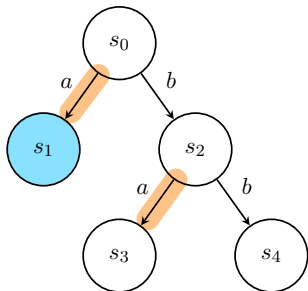


Strategies

Definition

A **strategy** of \mathcal{P}_i is a function $\sigma_i: (SA)^*S_i \rightarrow \mathcal{D}(A)$.

- We compare strategies **independently of any objective or payoff**.
- **Equality is too restrictive**: two different strategies may induce the **same behaviour** in practice.



Outcome-equivalence

Given two strategies σ_1 and σ_2 , and an initial state $s_{\text{init}} \in S$, we define a probability distribution on the set of plays in the usual way: for any history $h = s_0 a_0 s_1 \dots s_n$ with $s_0 = s_{\text{init}}$, we set

$$\mathbb{P}_s^{\sigma_1, \sigma_2}(\text{Cyl}(h)) = \prod_{k=0}^{n-1} \sigma_{i(k)}(s_0 a_0 \dots s_k) \cdot \delta(s_k, a_k, s_{k+1})$$

where $\text{Cyl}(h)$ is the set of plays with h as a prefix, and $i(k) = 1$ if $s_k \in S_1$ and 2 otherwise.

Outcome-equivalence

Given two strategies σ_1 and σ_2 , and an initial state $s_{\text{init}} \in S$, we define a probability distribution on the set of plays in the usual way: for any history $h = s_0 a_0 s_1 \dots s_n$ with $s_0 = s_{\text{init}}$, we set

$$\mathbb{P}_s^{\sigma_1, \sigma_2}(\text{Cyl}(h)) = \prod_{k=0}^{n-1} \sigma_{i(k)}(s_0 a_0 \dots s_k) \cdot \delta(s_k, a_k, s_{k+1})$$

where $\text{Cyl}(h)$ is the set of plays with h as a prefix, and $i(k) = 1$ if $s_k \in S_1$ and 2 otherwise.

Outcome-equivalence

Two strategies σ_1 and τ_1 of \mathcal{P}_1 are **outcome-equivalent** if for all strategies σ_2 of \mathcal{P}_2 and all initial states $s_{\text{init}} \in S$, we have

$$\mathbb{P}_{s_{\text{init}}}^{\sigma_1, \sigma_2} = \mathbb{P}_{s_{\text{init}}}^{\tau_1, \sigma_2}.$$

Table of contents

- 1 Games and strategies
- 2 Finite-memory strategies**
- 3 Inclusions between classes
- 4 Differences between classes
- 5 Extending the classification

Randomised finite-memory strategies

- In general, strategies can use unlimited memory, which is **unrealistic in practice**.

Randomised finite-memory strategies

- In general, strategies can use unlimited memory, which is **unrealistic in practice**.

Definition

A strategy σ_i of \mathcal{P}_i is **finite-memory** if it is induced by a stochastic **Mealy machine** $\mathcal{M} = (M, \mu_{\text{init}}, \alpha_{\text{next}}, \alpha_{\text{up}})$ where

- M is a finite set of memory states;
- $\mu_{\text{init}} \in \mathcal{D}(M)$ is an initial distribution;
- $\alpha_{\text{next}}: M \times S_i \rightarrow \mathcal{D}(A)$ is a stochastic next-move function;
- $\alpha_{\text{up}}: M \times S \times A \rightarrow \mathcal{D}(M)$ is a stochastic memory update function.

Randomised finite-memory strategies

- In general, strategies can use unlimited memory, which is **unrealistic in practice**.

Definition

A strategy σ_i of \mathcal{P}_i is **finite-memory** if it is induced by a stochastic **Mealy machine** $\mathcal{M} = (M, \mu_{\text{init}}, \alpha_{\text{next}}, \alpha_{\text{up}})$ where

- M is a finite set of memory states;
 - $\mu_{\text{init}} \in \mathcal{D}(M)$ is an initial distribution;
 - $\alpha_{\text{next}}: M \times S_i \rightarrow \mathcal{D}(A)$ is a stochastic next-move function;
 - $\alpha_{\text{up}}: M \times S \times A \rightarrow \mathcal{D}(M)$ is a stochastic memory update function.
-
- We can **classify Mealy machines** following whether their initialisation, updates and outputs are randomised or deterministic.

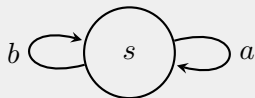
All classes of Mealy machines are not equally powerful

- Some classes of Mealy machines allow **richer behaviours** than others.

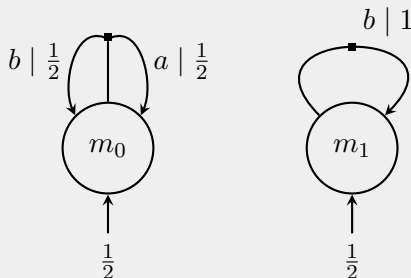
All classes of Mealy machines are not equally powerful

- Some classes of Mealy machines allow **richer behaviours** than others.
- For instance, the strategy illustrated on the right **cannot** be emulated with **randomisation only in the outputs**.

Game



Mealy machine example



Our results

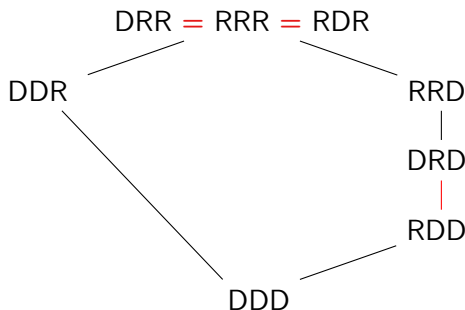
We use **acronyms** to define classes of Mealy machines: we use XYZ where $X, Y, Z \in \{D, R\}$ where D stands for deterministic and R for random, and

- X characterises initialisation,
- Y characterises outputs (next-move function),
- Z characterises updates.

Our results

We use **acronyms** to define classes of Mealy machines: we use XYZ where $X, Y, Z \in \{D, R\}$ where D stands for deterministic and R for random, and

- X characterises initialisation,
- Y characterises outputs (next-move function),
- Z characterises updates.



Illustrating a finite-memory strategy

In the sequel, we will illustrate fragments of Mealy machines for \mathcal{P}_i as follows.

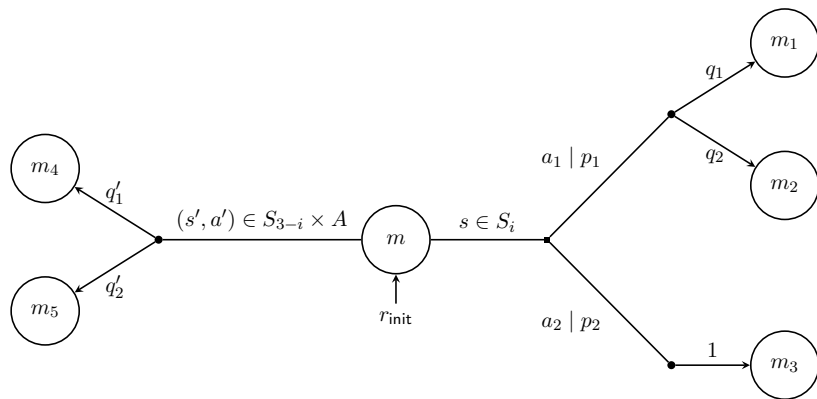


Table of contents

- 1 Games and strategies
- 2 Finite-memory strategies
- 3 Inclusions between classes**
- 4 Differences between classes
- 5 Extending the classification

RDD \subseteq DRD: trading random initialisation for outputs

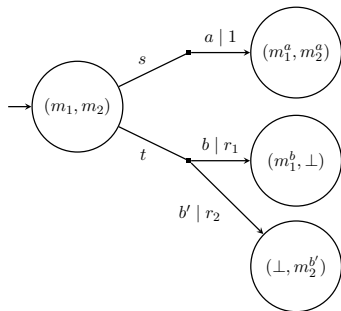
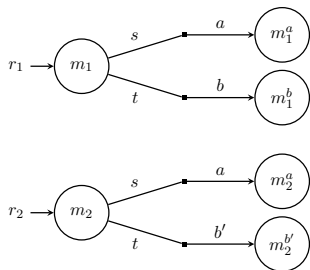
We fix an RDD Mealy machine $\mathcal{M} = (M, \mu_{\text{init}}, \alpha_{\text{next}}, \alpha_{\text{up}})$.

- We use an adaptation of the **subset construction** to go from \mathcal{M} to a DRD Mealy machine.
- State space of functions $f: \text{supp}(\mu_{\text{init}}) \rightarrow (M \cup \{\perp\})$:
 - We simulate the strategy from each initial state.
 - If an action is **inconsistent** with one of the simulations, we stop it (symbolised by \perp).

RDD \subseteq DRD: trading random initialisation for outputs

We fix an RDD Mealy machine $\mathcal{M} = (M, \mu_{\text{init}}, \alpha_{\text{next}}, \alpha_{\text{up}})$.

- We use an adaptation of the **subset construction** to go from \mathcal{M} to a DRD Mealy machine.
- State space of functions $f: \text{supp}(\mu_{\text{init}}) \rightarrow (M \cup \{\perp\})$:
 - We simulate the strategy from each initial state.
 - If an action is **inconsistent** with one of the simulations, we stop it (symbolised by \perp).



RRR \subseteq DRR: determinising initialisation

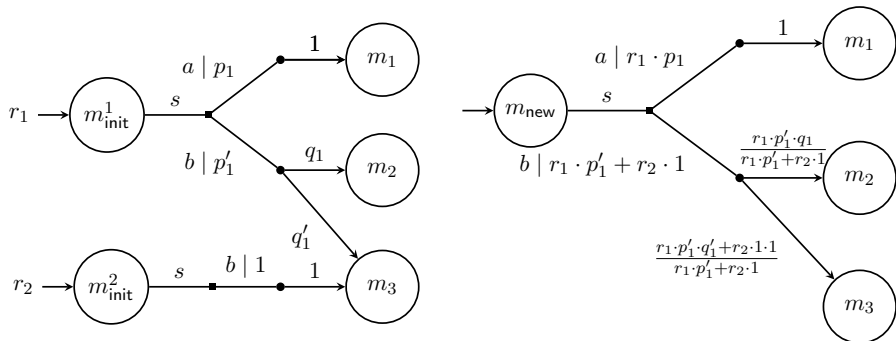
We fix an RRR Mealy machine $\mathcal{M} = (M, \mu_{\text{init}}, \alpha_{\text{next}}, \alpha_{\text{up}})$.

- To derive a DRR Mealy machine from \mathcal{M} , we add a **new initial state** m_{new} to the memory state space.
- We use **stochastic updates** to return to \mathcal{M} from m_{new} . Transition probabilities are chosen so the **distribution over memory states** is the same in \mathcal{M} and the DRR Mealy machine after the first step.

RRR \subseteq DRR: determinising initialisation

We fix an RRR Mealy machine $\mathcal{M} = (M, \mu_{\text{init}}, \alpha_{\text{next}}, \alpha_{\text{up}})$.

- To derive a DRR Mealy machine from \mathcal{M} , we add a **new initial state** m_{new} to the memory state space.
- We use **stochastic updates** to return to \mathcal{M} from m_{new} . Transition probabilities are chosen so the **distribution over memory states** is the same in \mathcal{M} and the DRR Mealy machine after the first step.



RRR \subseteq RDR: determinising outputs

We fix an RRR Mealy machine $\mathcal{M} = (M, \mu_{\text{init}}, \alpha_{\text{next}}, \alpha_{\text{up}})$.

- To derive a RDR Mealy machine from \mathcal{M} , we expand the state space by **augmenting memory states** with **pure memoryless strategies** $\sigma_i: S_i \rightarrow A$.
- We use stochastic initialisation and updates to **integrate the randomisation over actions in the transitions**.

RRR \subseteq RDR: determinising outputs

We fix an RRR Mealy machine $\mathcal{M} = (M, \mu_{\text{init}}, \alpha_{\text{next}}, \alpha_{\text{up}})$.

- To derive a RDR Mealy machine from \mathcal{M} , we expand the state space by **augmenting memory states** with **pure memoryless strategies** $\sigma_i: S_i \rightarrow A$.
- We use stochastic initialisation and updates to **integrate the randomisation over actions in the transitions**.

Naive construction \rightsquigarrow memory state space grows by a factor of $|A|^{|S_i|}$

\Leftrightarrow We can do better:

Theorem

There exists an RDR Mealy machine with $|M| \cdot |S_i| \cdot |A|$ states whose induced strategy is outcome-equivalent to \mathcal{M} .

RRR \subseteq RDR: choosing pure memoryless strategies

- Consider a game such that $S_i = \{s_1, s_2, s_3\}$, and $A = \{a_1, a_2, a_3\}$. Assume that for a memory state $m \in M$, we have:
 - $\alpha_{\text{next}}(m, s_1)(a_1) = \alpha_{\text{next}}(m, s_1)(a_2) = \frac{1}{2}$;
 - $\alpha_{\text{next}}(m, s_2)(a_1) = \alpha_{\text{next}}(m, s_2)(a_2) = \alpha_{\text{next}}(m, s_2)(a_3) = \frac{1}{3}$;
 - $\alpha_{\text{next}}(m, s_3)(a_1) = \frac{1}{3}$, $\alpha_{\text{next}}(m, s_3)(a_2) = \frac{1}{6}$ and $\alpha_{\text{next}}(m, s_3)(a_3) = \frac{1}{2}$.

RRR \subseteq RDR: choosing pure memoryless strategies

- Consider a game such that $S_i = \{s_1, s_2, s_3\}$, and $A = \{a_1, a_2, a_3\}$. Assume that for a memory state $m \in M$, we have:
 - $\alpha_{\text{next}}(m, s_1)(a_1) = \alpha_{\text{next}}(m, s_1)(a_2) = \frac{1}{2}$;
 - $\alpha_{\text{next}}(m, s_2)(a_1) = \alpha_{\text{next}}(m, s_2)(a_2) = \alpha_{\text{next}}(m, s_2)(a_3) = \frac{1}{3}$;
 - $\alpha_{\text{next}}(m, s_3)(a_1) = \frac{1}{3}$, $\alpha_{\text{next}}(m, s_3)(a_2) = \frac{1}{6}$ and $\alpha_{\text{next}}(m, s_3)(a_3) = \frac{1}{2}$.
- We represent the actions in a table to derive the pure memoryless strategies and their probabilities.

s_1	a_1		a_2
s_2	a_1	a_2	a_3
s_3	a_1	a_2	a_3

RRR \subseteq RDR: choosing pure memoryless strategies

- Consider a game such that $S_i = \{s_1, s_2, s_3\}$, and $A = \{a_1, a_2, a_3\}$. Assume that for a memory state $m \in M$, we have:
 - $\alpha_{\text{next}}(m, s_1)(a_1) = \alpha_{\text{next}}(m, s_1)(a_2) = \frac{1}{2}$;
 - $\alpha_{\text{next}}(m, s_2)(a_1) = \alpha_{\text{next}}(m, s_2)(a_2) = \alpha_{\text{next}}(m, s_2)(a_3) = \frac{1}{3}$;
 - $\alpha_{\text{next}}(m, s_3)(a_1) = \frac{1}{3}$, $\alpha_{\text{next}}(m, s_3)(a_2) = \frac{1}{6}$ and $\alpha_{\text{next}}(m, s_3)(a_3) = \frac{1}{2}$.
- We represent the actions in a table to derive the pure memoryless strategies and their probabilities.

s_1	a_1			a_2
s_2	a_1		a_2	a_3
s_3	a_1	a_2		a_3

RRR \subseteq RDR: choosing pure memoryless strategies

- Consider a game such that $S_i = \{s_1, s_2, s_3\}$, and $A = \{a_1, a_2, a_3\}$. Assume that for a memory state $m \in M$, we have:
 - $\alpha_{\text{next}}(m, s_1)(a_1) = \alpha_{\text{next}}(m, s_1)(a_2) = \frac{1}{2}$;
 - $\alpha_{\text{next}}(m, s_2)(a_1) = \alpha_{\text{next}}(m, s_2)(a_2) = \alpha_{\text{next}}(m, s_2)(a_3) = \frac{1}{3}$;
 - $\alpha_{\text{next}}(m, s_3)(a_1) = \frac{1}{3}$, $\alpha_{\text{next}}(m, s_3)(a_2) = \frac{1}{6}$ and $\alpha_{\text{next}}(m, s_3)(a_3) = \frac{1}{2}$.
- We represent the actions in a table to derive the pure memoryless strategies and their probabilities.

s_1	a_1			a_2
s_2	a_1	a_2		a_3
s_3	a_1	a_2		a_3
σ_k	σ_1	σ_2	σ_3	σ_4
	$x_1 = 0$	$x_2 = \frac{1}{3}$	$x_3 = \frac{1}{2}$	$x_4 = \frac{2}{3}$
				$x_5 = 1$

RRR \subseteq RDR: exploiting the memoryless strategies

- For each memory state $m \in M$, we determine **pure memoryless strategies** $\sigma_1^m, \dots, \sigma_{\ell(m)}^m$ and their respective **probabilities** $p_1^m, \dots, p_{\ell(m)}^m$.
- We **split transitions** that enter m into transitions that go to the states (m, σ_j^m) : a transition of probability q into m yields a transition with probability $q \cdot p_j^m$ into (m, σ_j^m) .

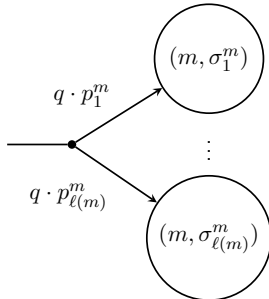
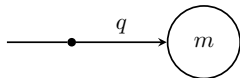


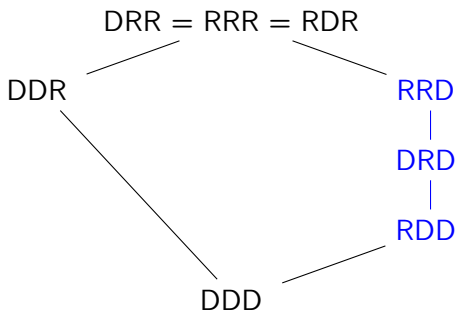
Table of contents

- 1 Games and strategies
- 2 Finite-memory strategies
- 3 Inclusions between classes
- 4 Differences between classes**
- 5 Extending the classification

Differences between classes

We discuss the following aspects:

- The chain of inclusions $DDD \subsetneq RDD \subsetneq DRD \subsetneq RRD \subsetneq RRR$ is strict.
- It holds that $DDR \not\subseteq RRD$ and $RDD \not\subseteq DDR$.



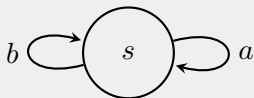
Strictness: $RDD \subsetneq DRD$

- In a one-player deterministic game, RDD strategies have **finitely many outcomes**.

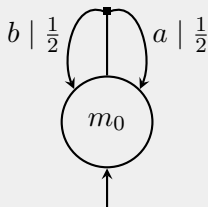
Strictness: $RDD \subsetneq DRD$

- In a one-player deterministic game, RDD strategies have **finitely many outcomes**.
- The DRD strategy depicted below has **no RDD equivalent**.

Game



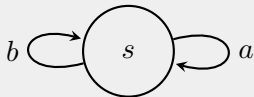
Witness of $RDD \subsetneq DRD$



Strictness: $DDR \not\subseteq RRD$

- The number of memory states in which we can find ourselves as a play goes on **cannot increase** for an RRD strategy..
- To have a **positive probability of never using a** , we must eventually be in a memory state m such that $\alpha_{\text{next}}(m, s)(a) = 0$ with positive probability.

Game



Can be adapted to witness $DDR \not\subseteq RRD$

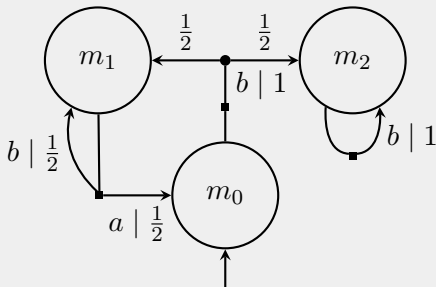


Table of contents

- 1 Games and strategies
- 2 Finite-memory strategies
- 3 Inclusions between classes
- 4 Differences between classes
- 5 Extending the classification**

Taxonomy in settings of partial information

- Up to now, we have discussed a classification of strategies in a setting of **perfect information**.

↪ Can we weaken this hypothesis ?

- It is not necessary to see the **states themselves**.
- For the inclusion $RDD \subseteq DRD$, we rely on the **visibility of actions** in our subset construction.
- For the inclusion $RRR \subseteq DRR$, we also use the **visibility of actions** in conditional probabilities.

Partial information

The classification holds in games where \mathcal{P}_i can **see their actions** and **distinguish the owner** of states from their observations.

References I

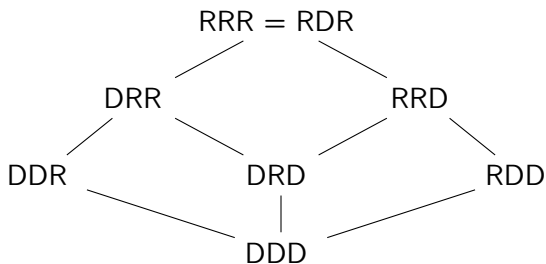
Aumann, Robert J . “28. Mixed and Behavior Strategies in Infinite Extensive Games”. In: *Advances in Game Theory. (AM-52), Volume 52*. Princeton University Press, 2016, pp. 627–650. DOI: doi:10.1515/9781400882014-029. URL: <https://doi.org/10.1515/9781400882014-029>.

Collapses – invisible actions

What happens to the lattice in full generality ? If we assume nothing on the visibility of actions ?

- Two inclusions of our lattice no longer hold. We have:
 - $RDD \not\subseteq DRD$;
 - $RRR \not\subseteq DRR$ (we even have $RDD \not\subseteq DRR$).
- Intuitively, for a strategy with **deterministic outputs** (i.e., in a subclass of RDR), the output actions are **encoded in the Mealy machine itself**.
 \rightsquigarrow such strategies allow the same behaviours **whether actions are visible or not**.

General lattice: no hypotheses on actions



Subgame perfect equilibria and Kuhn's theorem

- In the statement of Kuhn's theorem and our classification, the output of the strategies along **inconsistent branches** histories are completely disregarded.
- In other words, our classification approach is not relevant for the study of **subgame perfect equilibria**, for which these inconsistent histories are nonetheless taken in account.
- However, the output of a finite-memory strategy along an inconsistent history is **not well-defined**.