

Asynchronous distributed games

P. Gastin, B. Lerman, M. Zeitoun

LIAFA, CNRS & Univ. Paris 7

Motivations & contributions of this work

- ★ Framework for **asynchronous** multi-players distributed games.
- ★ **Communication** to cooperate: 2 teams of players with shared memory.
Some memory locations can be read by some players of both teams.

- ★ Definition of **asynchronous distributed games (ADG)**
- ★ Relate ADG and sequential games + reset moves.
- ★ Discuss existence of winning distributed strategies.

- ★ Definition of **asynchronous alternating automata (AAA)**.
- ★ Natural definition of **runs** by prime event-structures.
- ★ Link acceptance in **AAA** and existence of winning strategies in **ADG**.

Motivations & contributions of this work

- ★ Framework for **asynchronous** multi-players distributed games.
- ★ **Communication** to cooperate: 2 teams of players with shared memory.
Some memory locations can be read by some players of both teams.

- ★ Definition of **asynchronous distributed games (ADG)**
- ★ Relate ADG and sequential games + reset moves.
- ★ Discuss existence of winning distributed strategies.

- ★ Definition of **asynchronous alternating automata (AAA)**.
- ★ Natural definition of **runs** by prime event-structures.
- ★ Link acceptance in **AAA** and existence of winning strategies in **ADG**.

Motivations & contributions of this work

- ★ Framework for **asynchronous** multi-players distributed games.
- ★ **Communication** to cooperate: 2 teams of players with shared memory.
Some memory locations can be read by some players of both teams.

- ★ Definition of **asynchronous distributed games (ADG)**
- ★ Relate ADG and sequential games + reset moves.
- ★ Discuss existence of winning distributed strategies.

- ★ Definition of **asynchronous alternating automata (AAA)**.
- ★ Natural definition of **runs** by prime event-structures.
- ★ Link acceptance in **AAA** and existence of winning strategies in **ADG**.

Distributed games

Control synthesis in the Ramadge-Wonham setting

In the sequential setting, regular system \mathcal{S} with

- controllable/uncontrollable actions.
- observable/unobservable actions.
- Desired behaviors (= regular language) \mathcal{G} .
- Look for an **optimal** (less restrictive) controller \mathcal{C} st. $\mathcal{S} \times \mathcal{C} \subseteq \mathcal{G}$.
- Looking for any controller, with no minimal requirements, is easy: disable every controllable action and check it works.

In our context, two **restrictions** for now:

- Everything is observable.
- One looks for **some** controller (perhaps not the optimal one).
The controller might be forced to enable actions anyway.

Distributed architectures

$(\Sigma, \mathcal{P}, R, W)$:

- Σ : finite set of actions or players
- \mathcal{P} : finite set of processes (memory cells)
- $R : \Sigma \rightarrow 2^{\mathcal{P}}$ assigns to each $a \in \Sigma$ its read domain $R(a)$
- $W : \Sigma \rightarrow 2^{\mathcal{P}}$ assigns to each $a \in \Sigma$ its write domain $W(a)$

Distributed architectures

$(\Sigma, \mathcal{P}, R, W)$:

- Σ : finite set of actions or players
- \mathcal{P} : finite set of processes (memory cells)
- $R : \Sigma \rightarrow 2^{\mathcal{P}}$ assigns to each $a \in \Sigma$ its read domain $R(a)$
- $W : \Sigma \rightarrow 2^{\mathcal{P}}$ assigns to each $a \in \Sigma$ its write domain $W(a)$

$$\Sigma = \{a \ b\}$$

Distributed architectures

$(\Sigma, \mathcal{P}, R, W)$:

- Σ : finite set of actions or players
- \mathcal{P} : finite set of processes (memory cells)
- $R : \Sigma \rightarrow 2^{\mathcal{P}}$ assigns to each $a \in \Sigma$ its read domain $R(a)$
- $W : \Sigma \rightarrow 2^{\mathcal{P}}$ assigns to each $a \in \Sigma$ its write domain $W(a)$

$$\Sigma = \{a \ b\}$$

1

2

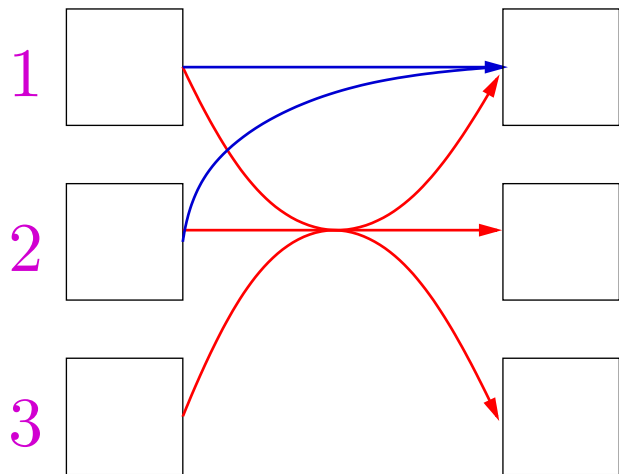
3

Distributed architectures

$(\Sigma, \mathcal{P}, R, W)$:

- Σ : finite set of actions or players
- \mathcal{P} : finite set of processes (memory cells)
- $R : \Sigma \rightarrow 2^{\mathcal{P}}$ assigns to each $a \in \Sigma$ its read domain $R(a)$
- $W : \Sigma \rightarrow 2^{\mathcal{P}}$ assigns to each $a \in \Sigma$ its write domain $W(a)$

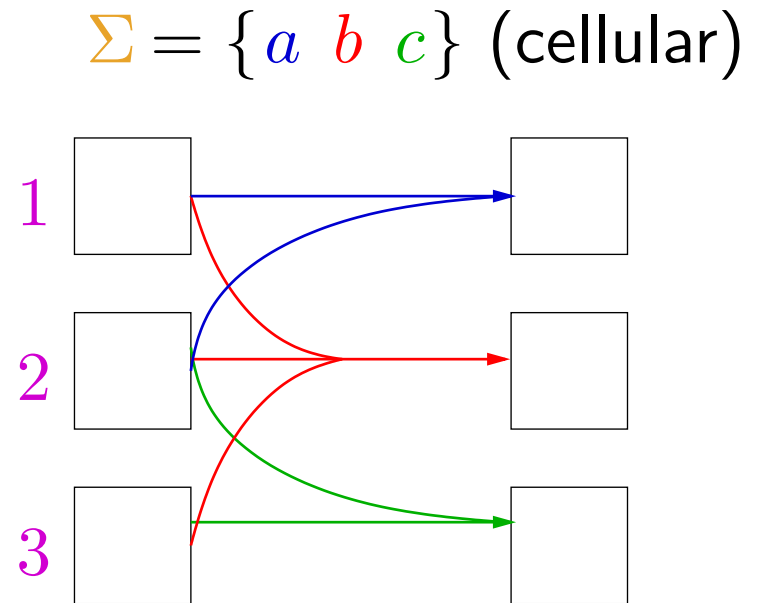
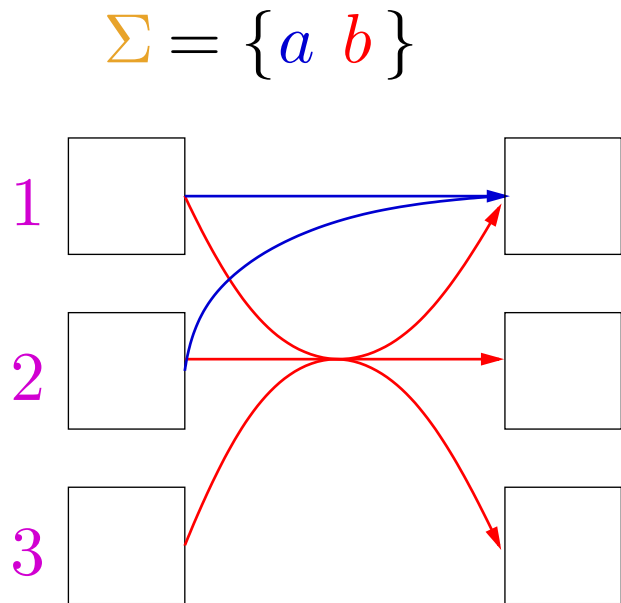
$$\Sigma = \{a \ b\}$$



Distributed architectures

$(\Sigma, \mathcal{P}, R, W)$:

- Σ : finite set of **actions** or **players**
- \mathcal{P} : finite set of **processes** (**memory cells**)
- $R : \Sigma \rightarrow 2^{\mathcal{P}}$ assigns to each $a \in \Sigma$ its read domain $R(a)$
- $W : \Sigma \rightarrow 2^{\mathcal{P}}$ assigns to each $a \in \Sigma$ its write domain $W(a)$



Asynchronous distributed systems (ADS)

$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

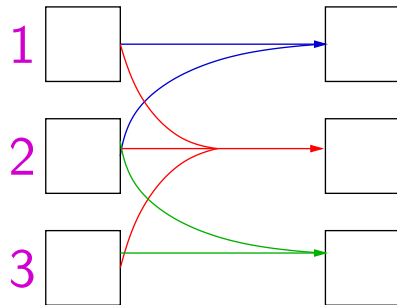
- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

Asynchronous distributed systems (ADS)

$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

Architecture

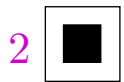
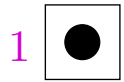


Asynchronous distributed systems (ADS)

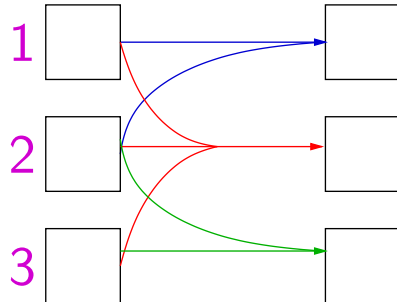
$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

Global state



Architecture

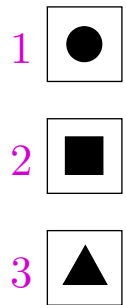


Asynchronous distributed systems (ADS)

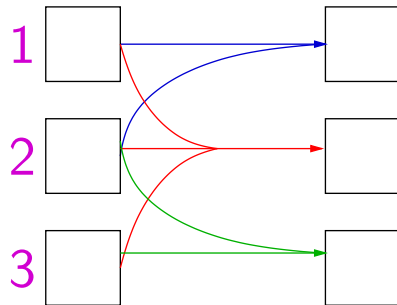
$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

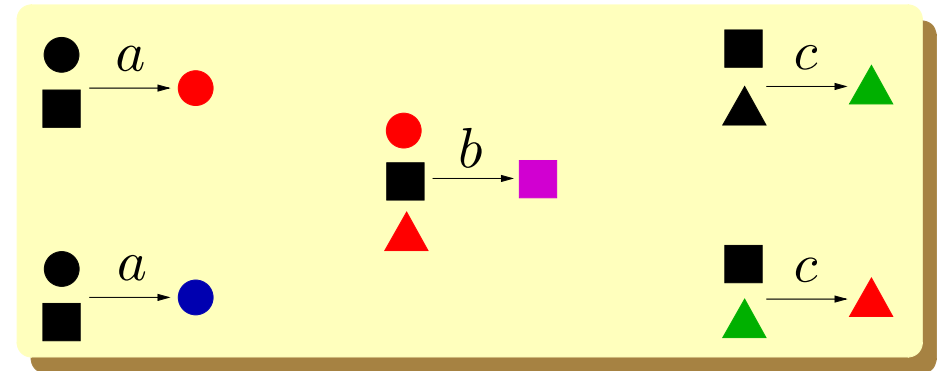
Global state



Architecture



Transitions

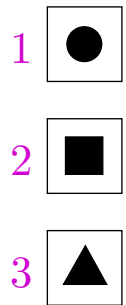


Asynchronous distributed systems (ADS)

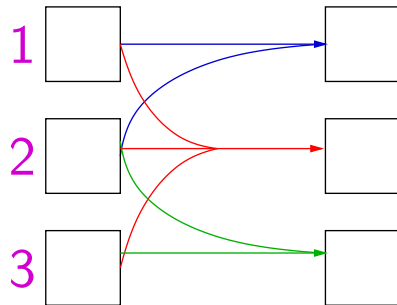
$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

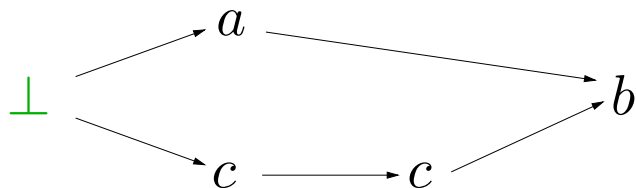
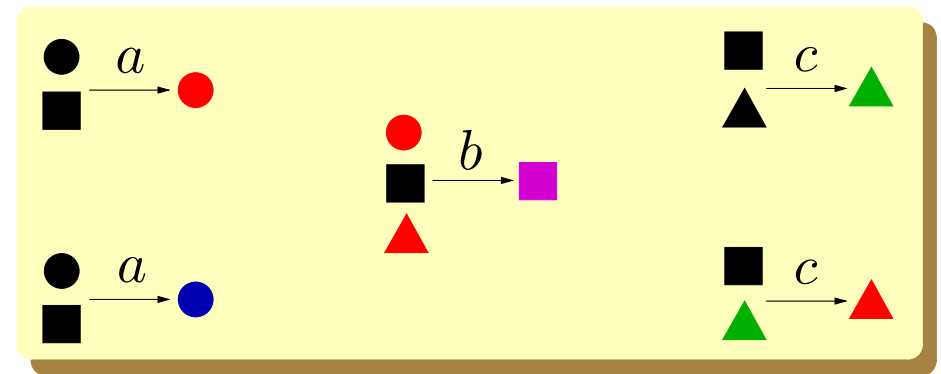
Global state



Architecture



Transitions

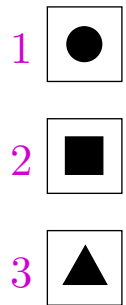


Asynchronous distributed systems (ADS)

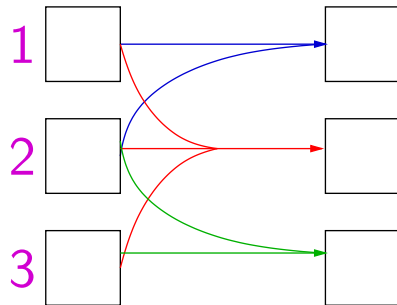
$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

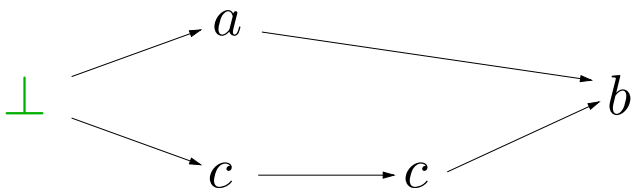
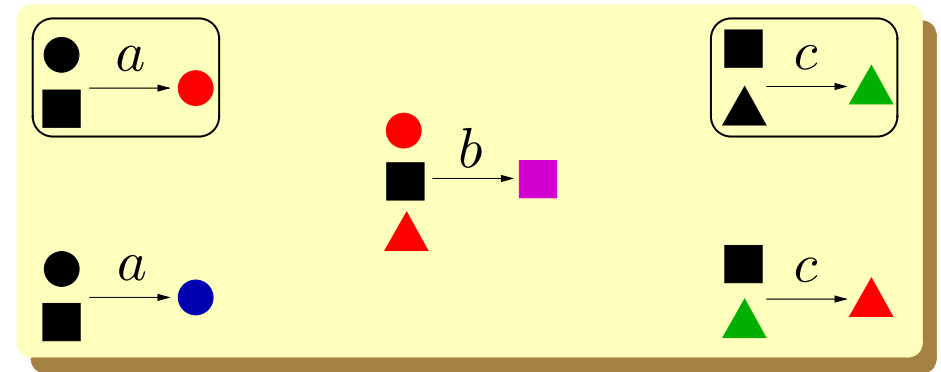
Global state



Architecture



Transitions

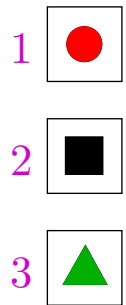


Asynchronous distributed systems (ADS)

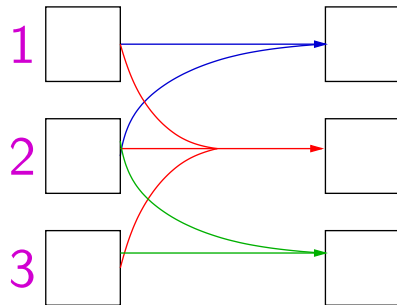
$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

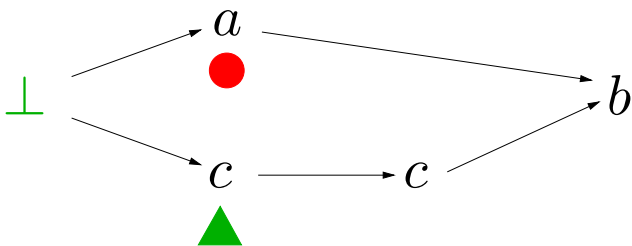
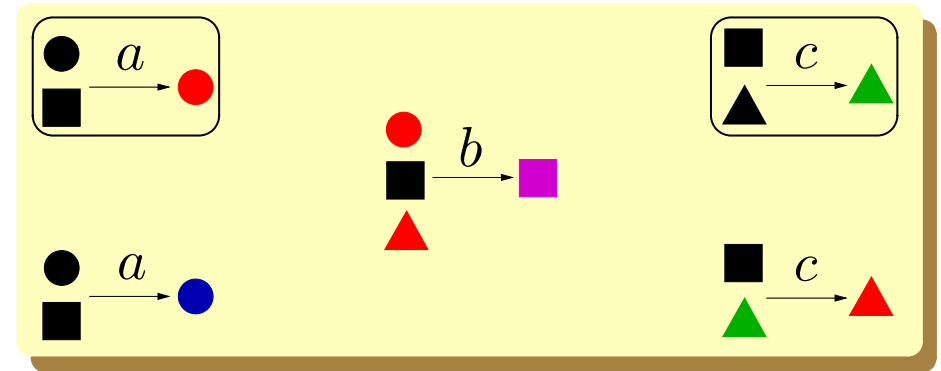
Global state



Architecture



Transitions

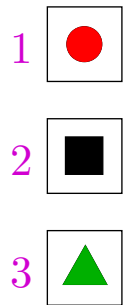


Asynchronous distributed systems (ADS)

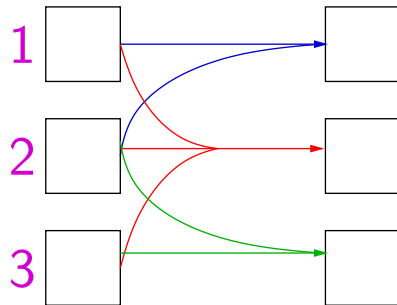
$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

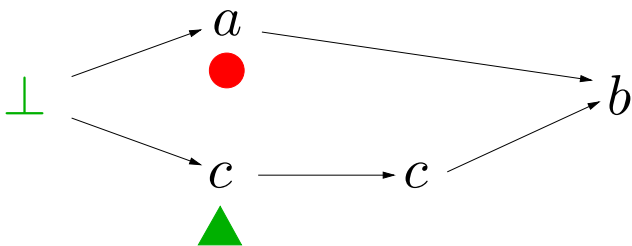
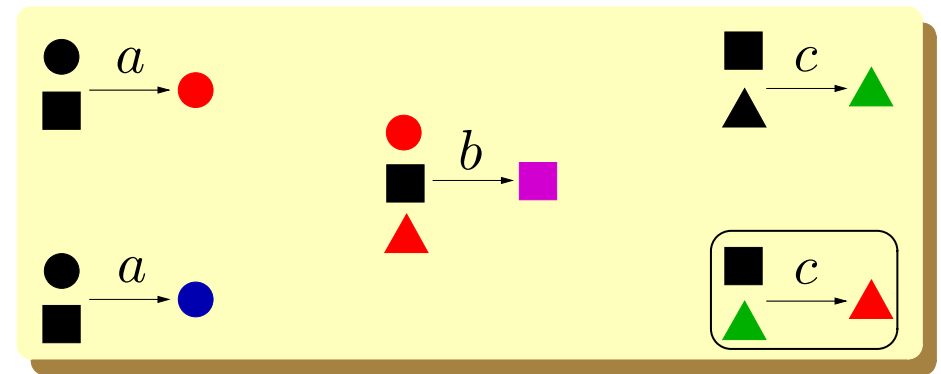
Global state



Architecture



Transitions

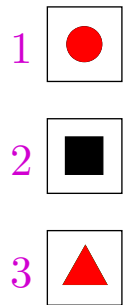


Asynchronous distributed systems (ADS)

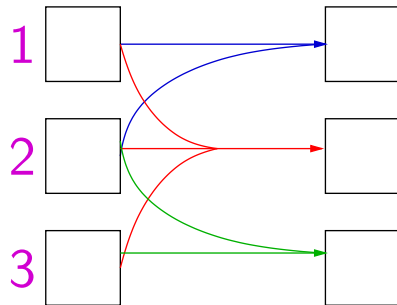
$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

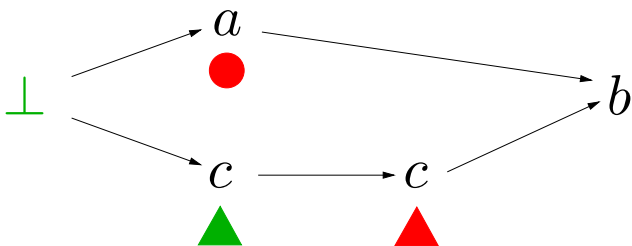
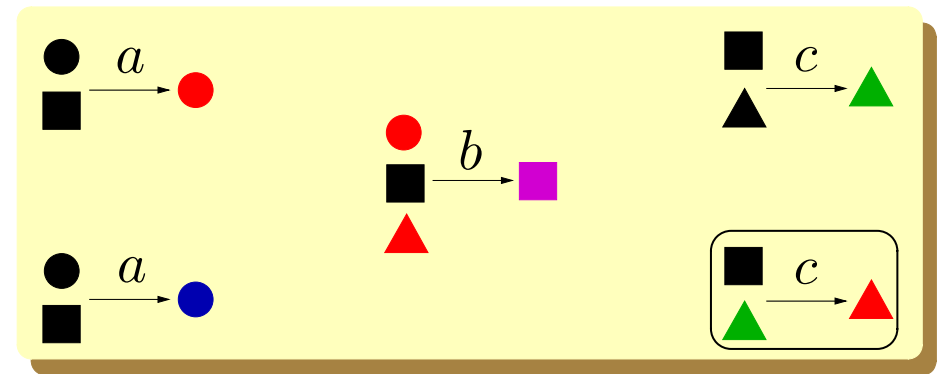
Global state



Architecture



Transitions

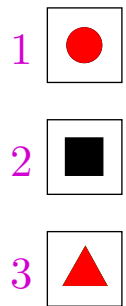


Asynchronous distributed systems (ADS)

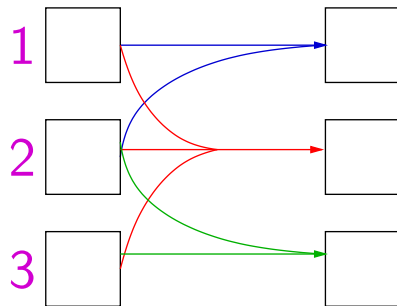
$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

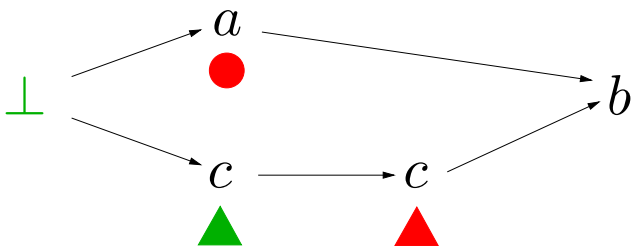
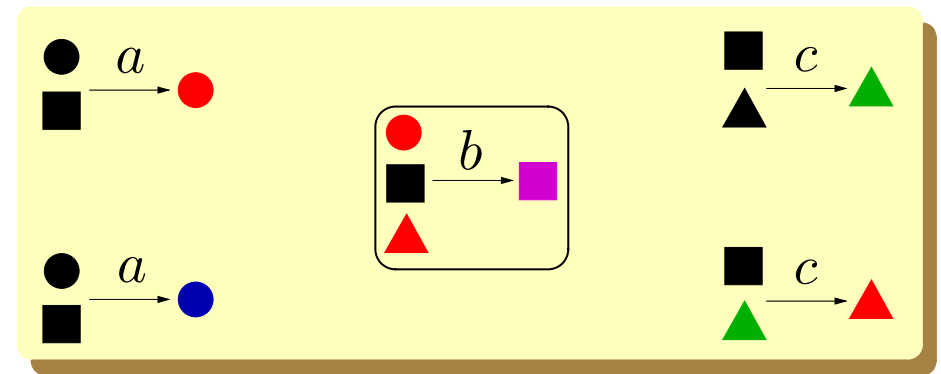
Global state



Architecture



Transitions

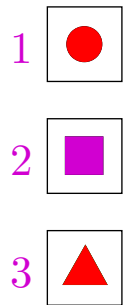


Asynchronous distributed systems (ADS)

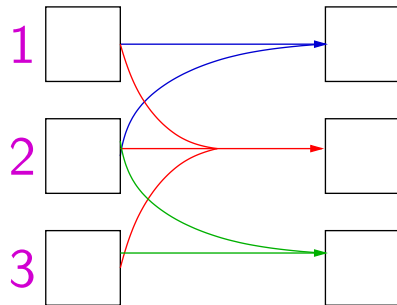
$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

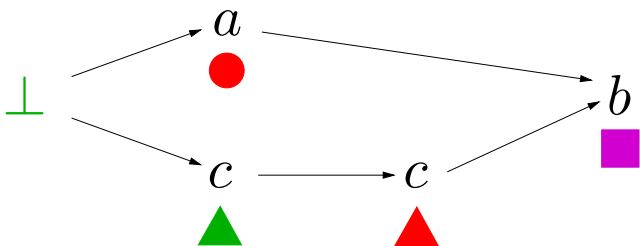
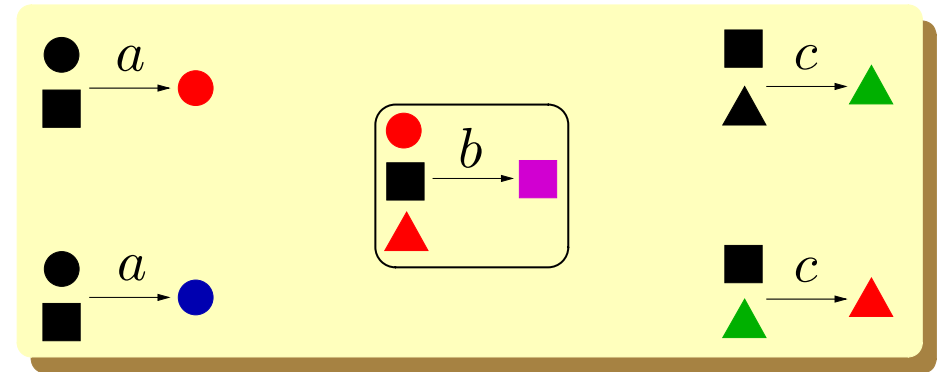
Global state



Architecture



Transitions

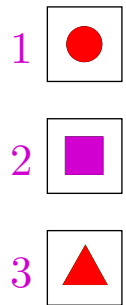


Asynchronous distributed systems (ADS)

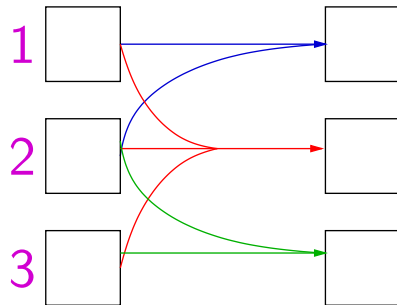
$\langle (Q_i)_{i \in \mathcal{P}}, (T_a)_{a \in \Sigma}, q^0 \rangle$ asynchronous distributed system over $(\Sigma, \mathcal{P}, R, W)$

- $\forall i \in \mathcal{P}$, Q_i is the set of local states for process i
- $\forall a \in \Sigma$, $T_a \subseteq Q_{R(a)} \times Q_{W(a)}$, possible local moves of action a
- $q^0 \in Q = \prod_{i \in \mathcal{P}} Q_i$ is the global starting position of G

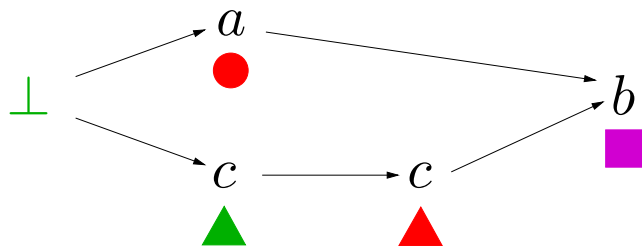
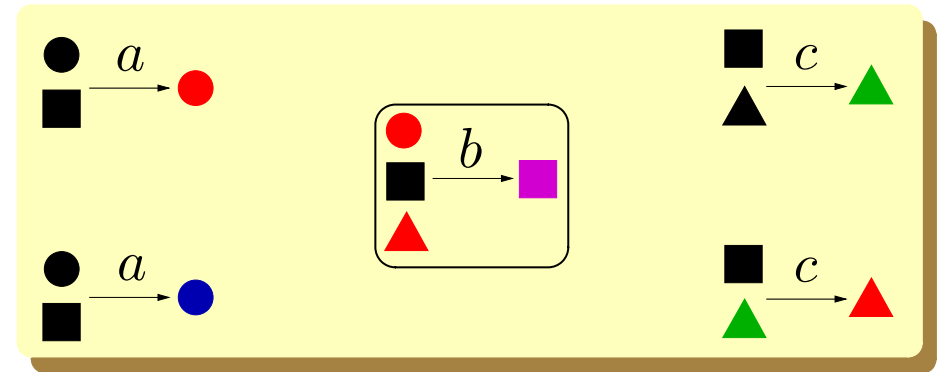
Global state



Architecture



Transitions



Mazurkiewicz trace over $\Sigma' = \Sigma \times \prod Q_i$

Mazurkiewicz traces

Dependence alphabet (Σ, D) with D reflexive & symmetric.

Mazurkiewicz trace $t = (V, \leq, \lambda)$ where (V, \leq) poset and $\lambda : V \rightarrow \Sigma$ st.

$$\begin{cases} x \prec y \Rightarrow \lambda(x) D \lambda(y) \\ \lambda(x) D \lambda(y) \Rightarrow x \leq y \text{ or } y \leq x \end{cases}$$

The labeling and the order relations are related.

Several special properties (eg uniqueness of a trace coming from a linearization)

Mazurkiewicz traces

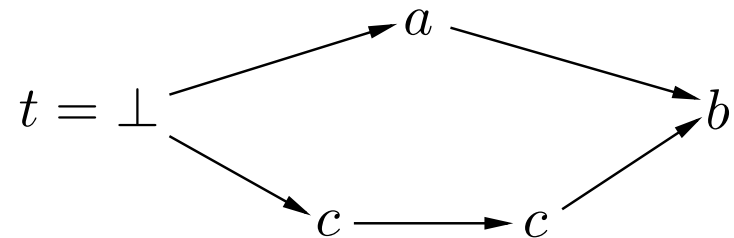
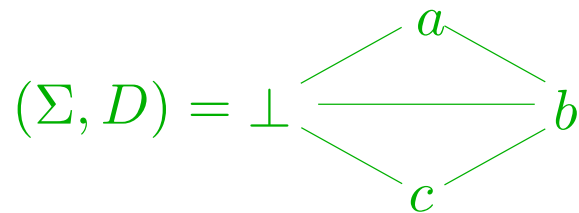
Dependence alphabet (Σ, D) with D reflexive & symmetric.

Mazurkiewicz trace $t = (V, \leq, \lambda)$ where (V, \leq) poset and $\lambda : V \rightarrow \Sigma$ st.

$$\begin{cases} x \prec y \Rightarrow \lambda(x) D \lambda(y) \\ \lambda(x) D \lambda(y) \Rightarrow x \leq y \text{ or } y \leq x \end{cases}$$

The labeling and the order relations are related.

Several special properties (eg uniqueness of a trace coming from a linearization)



Mazurkiewicz traces

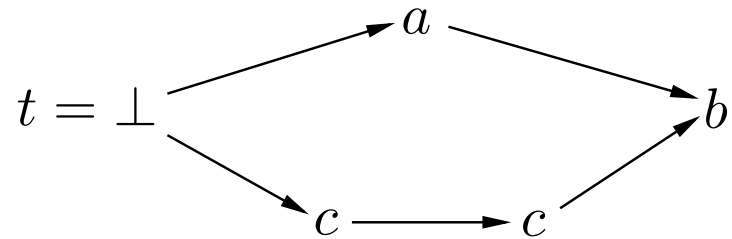
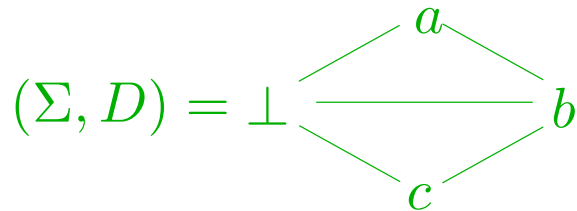
Dependence alphabet (Σ, D) with D reflexive & symmetric.

Mazurkiewicz trace $t = (V, \leq, \lambda)$ where (V, \leq) poset and $\lambda : V \rightarrow \Sigma$ st.

$$\begin{cases} x \prec y \Rightarrow \lambda(x) D \lambda(y) \\ \lambda(x) D \lambda(y) \Rightarrow x \leq y \text{ or } y \leq x \end{cases}$$

The labeling and the order relations are related.

Several special properties (eg uniqueness of a trace coming from a linearization)



Technical restriction

$$\begin{aligned} \forall a \in \Sigma, \quad \emptyset \neq W(a) \subseteq R(a) \\ \forall a, b \in \Sigma, \quad R(a) \cap W(b) = \emptyset \iff R(b) \cap W(a) = \emptyset \end{aligned} \quad (\mathcal{S})$$

Dependence relation over Σ : $D = \{(a, b) \mid R(a) \cap W(b) \neq \emptyset\}$

From ADS to asynchronous distributed games (ADG)

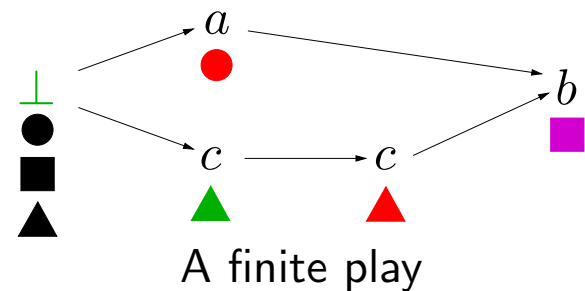
- $\Sigma = \Sigma_0 \uplus \Sigma_1$: partition of the set of players (actions) in teams 0 and 1.
Players of team 0 cooperate together against team 1.
- Each player only has a partial view of the global history.

From ADS to asynchronous distributed games (ADG)

- $\Sigma = \Sigma_0 \uplus \Sigma_1$: partition of the set of players (actions) in teams 0 and 1. Players of team 0 cooperate together against team 1.
- Each player only has a partial view of the global history.

$$\Sigma' = \{(a, p) \mid a \in \Sigma, p \in Q_{W(a)}\} \cup \{(\perp, q^0)\}, \quad (a, p) D (b, q) \Leftrightarrow a D b$$

- Positions of the game $V \subseteq \mathbb{M}(\Sigma', D)$.
- Play: finite or infinite trace of $\mathbb{R}(\Sigma', D)$.
- **Rules** of the game for player a given by T_a .

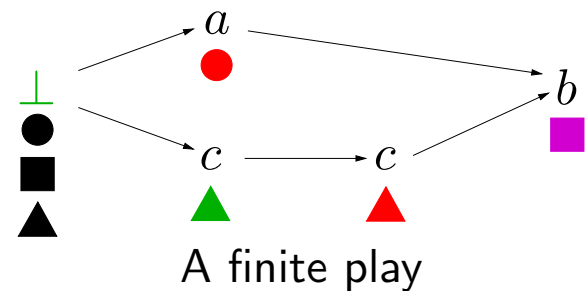


From ADS to asynchronous distributed games (ADG)

- $\Sigma = \Sigma_0 \uplus \Sigma_1$: partition of the set of players (actions) in teams **0** and **1**.
Players of team **0** cooperate together against team **1**.
- Each player only has a partial view of the global history.

$$\Sigma' = \{(a, p) \mid a \in \Sigma, p \in Q_{W(a)}\} \cup \{(\perp, q^0)\}, \quad (a, p) D (b, q) \Leftrightarrow a D b$$

- Positions of the game $V \subseteq \mathbb{M}(\Sigma', D)$.
- Play: finite or infinite trace of $\mathbb{R}(\Sigma', D)$.
- **Rules** of the game for player a given by T_a .



- Winning condition: set of finite or infinite traces $\mathcal{W} \subseteq \mathbb{R}(\Sigma', D)$.
Team **0** wins plays of \mathcal{W} and loses plays of $\mathbb{R}(\Sigma', D) \setminus \mathcal{W}$.
- **Remarks** Teams **0** and **1** might be simultaneously enabled at $v \in V$.
One can recover the global state from a distributed play.

Deterministic distributed strategies (DS)

- A **distributed strategy (DS)** for team 0 tells players of team 0 how to play.
- A distributed strategy f gives possible moves, hence must follow the rules.
- A strategy f is **winning (WDS)** if all plays played according to f are in \mathcal{W} .
- Advised moves only depend on the player's **view**, **not** on the **full history**.
- We assume all actions are **observable**.

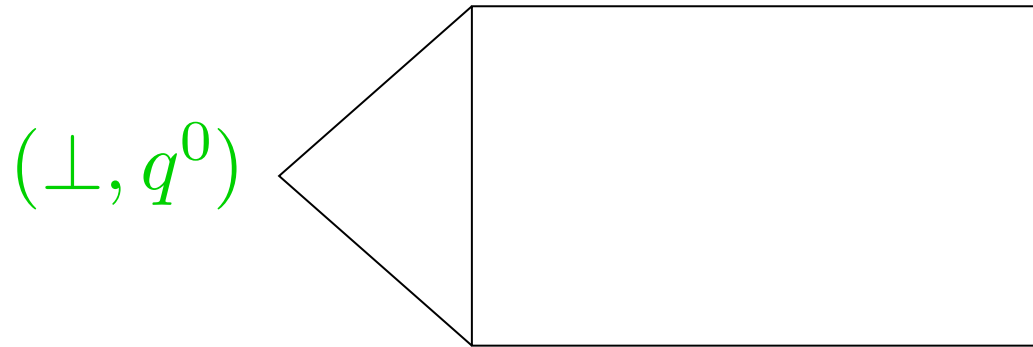
Deterministic distributed strategies (DS)

- A **distributed strategy (DS)** for team 0 tells players of team 0 how to play.
- A distributed strategy f gives possible moves, hence must follow the rules.
- A strategy f is **winning (WDS)** if all plays played according to f are in \mathcal{W} .
- Advised moves only depend on the player's **view**, **not** on the **full history**.
- We assume all actions are **observable**.

- As in the classical setting, one can see
 - an **ADG** as a distributed **program** embedded in an **environment**.
 - ▷ Actions of Σ_0 can be controlled, actions of Σ_1 cannot.
 - a **distributed play** as an **execution** of the program.
 - the **winning condition** as a specification of **good behaviors**.
 - Solving a distributed game = computing a distributed controller.

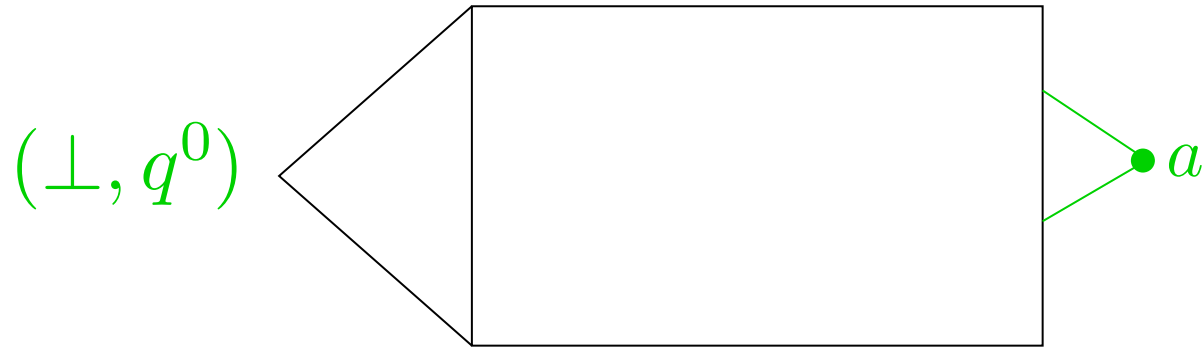
Distributed strategies with perfect memory

Recall that a **distributed play** is a rooted trace of $\mathbb{R}(\Sigma', D)$.



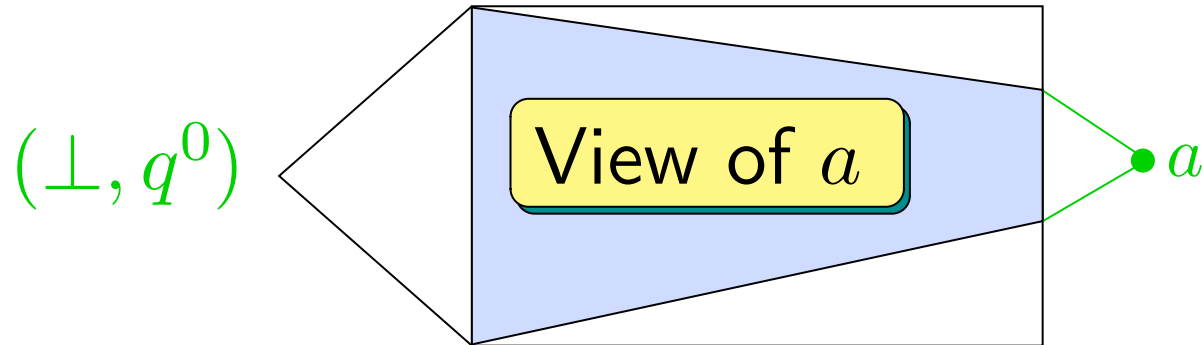
Distributed strategies with perfect memory

Recall that a **distributed play** is a rooted trace of $\mathbb{R}(\Sigma', D)$.



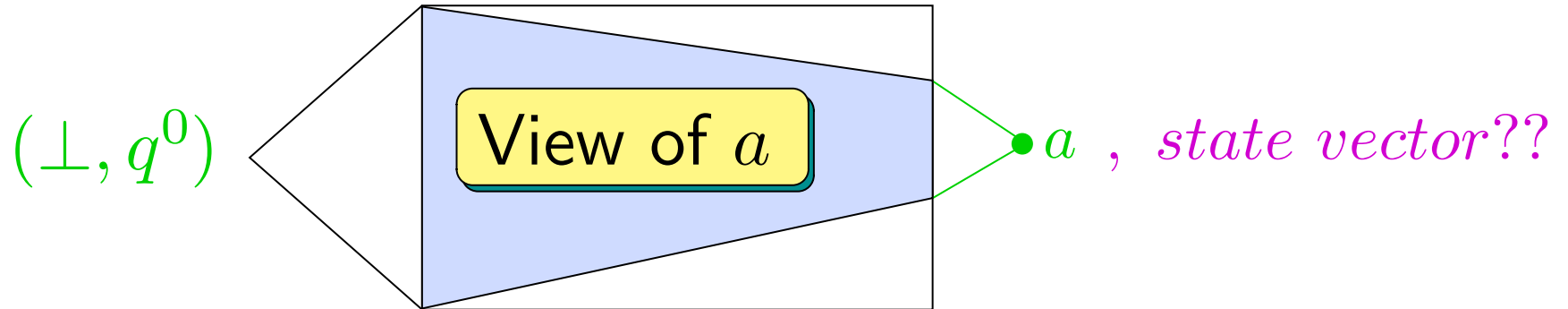
Distributed strategies with perfect memory

Recall that a **distributed play** is a rooted trace of $\mathbb{R}(\Sigma', D)$.



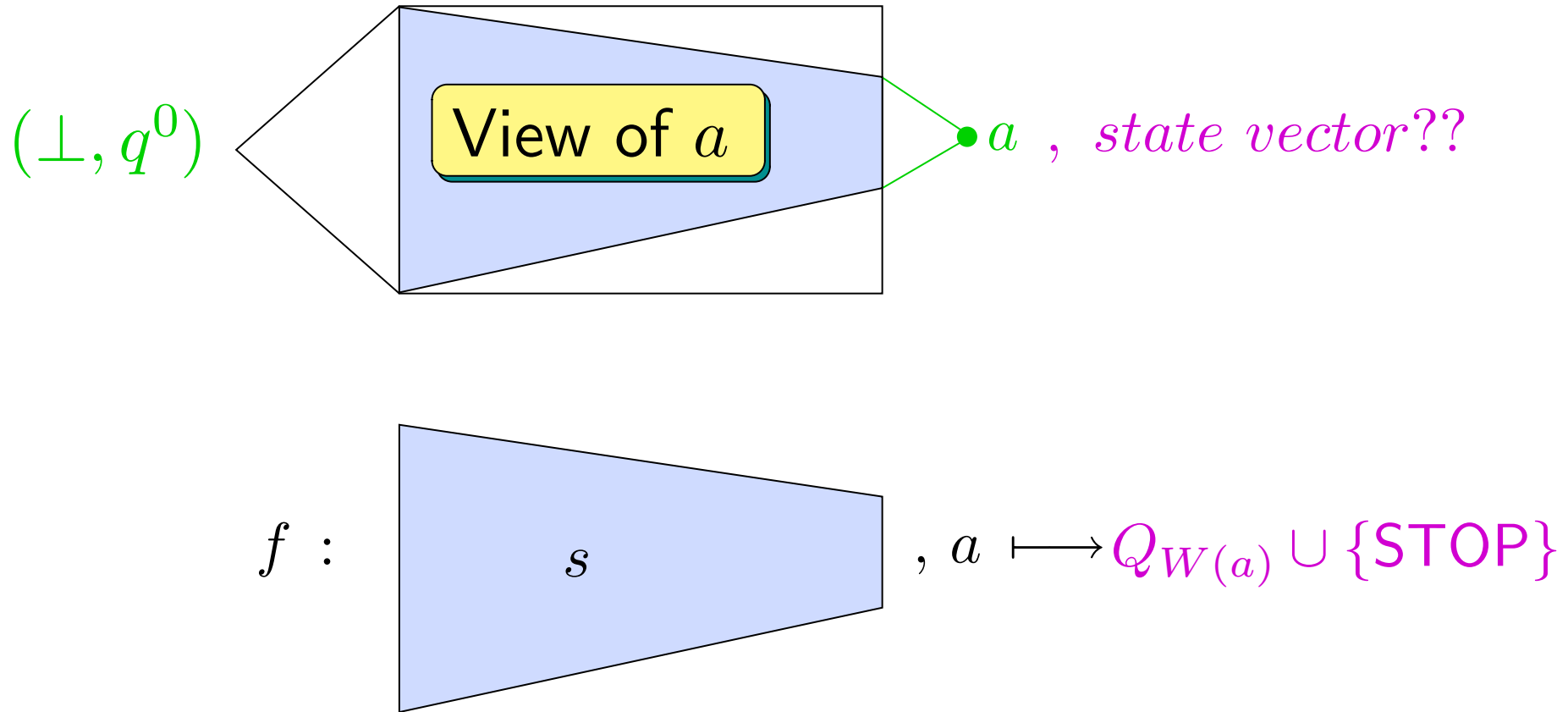
Distributed strategies with perfect memory

Recall that a **distributed play** is a rooted trace of $\mathbb{R}(\Sigma', D)$.



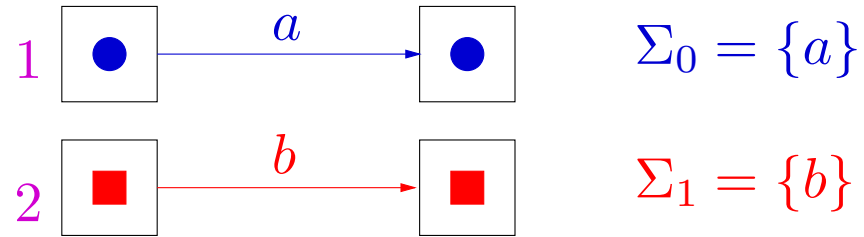
Distributed strategies with perfect memory

Recall that a **distributed play** is a rooted trace of $\mathbb{R}(\Sigma', D)$.

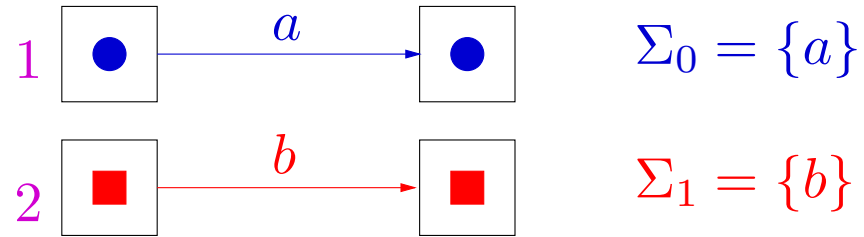


- The strategy for a does **not** use the **whole history of the game**, only a 's view.
- The strategy has the ability to advise a to **stop**.
- All DS are abstractions of this perfect memory DS.

Distributed strategies: the need to stop

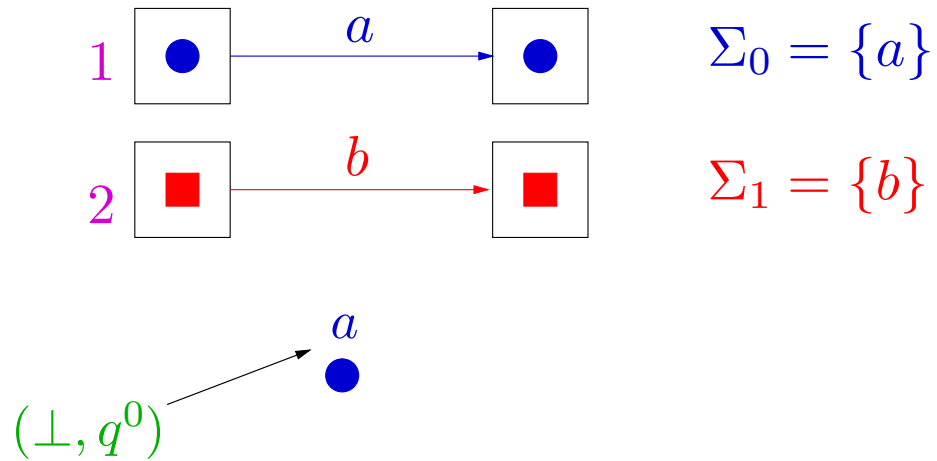


Distributed strategies: the need to stop

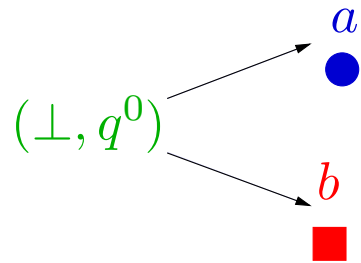
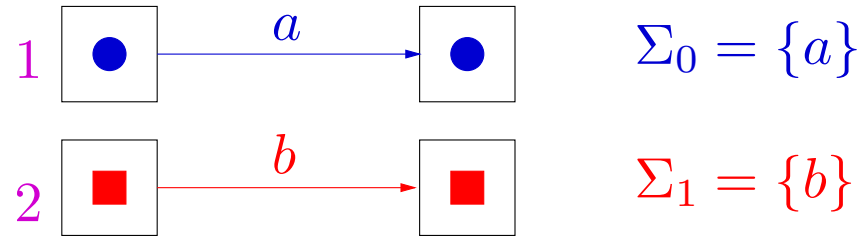


(\perp, q^0)

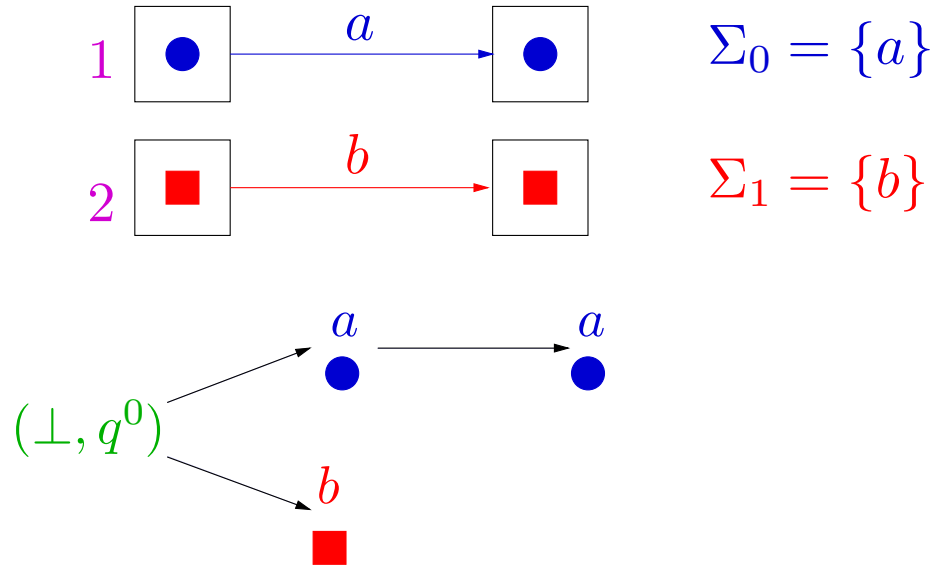
Distributed strategies: the need to stop



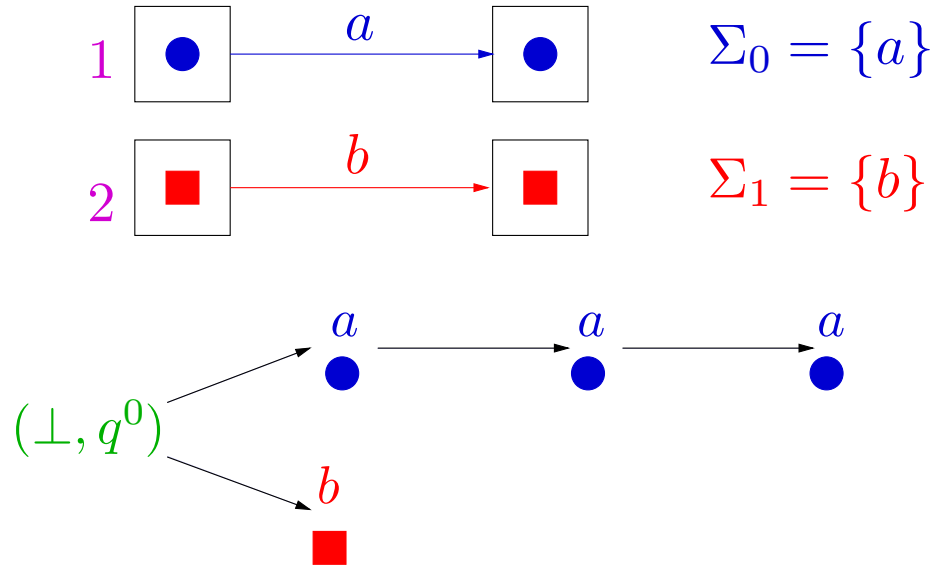
Distributed strategies: the need to stop



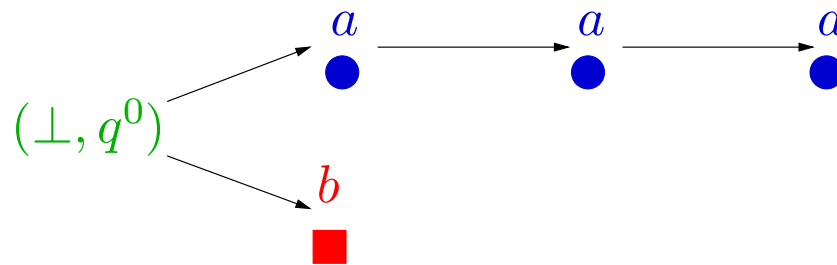
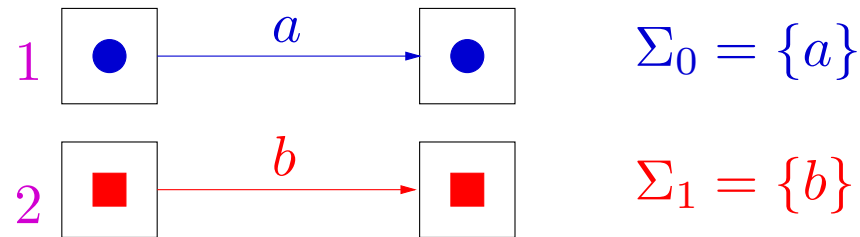
Distributed strategies: the need to stop



Distributed strategies: the need to stop



Distributed strategies: the need to stop



$$\mathcal{W} = \mathbb{M}(\Sigma', D) \cup \{(\perp, q^0)(a, \bullet)^\omega (b, \blacksquare)^\omega\}$$

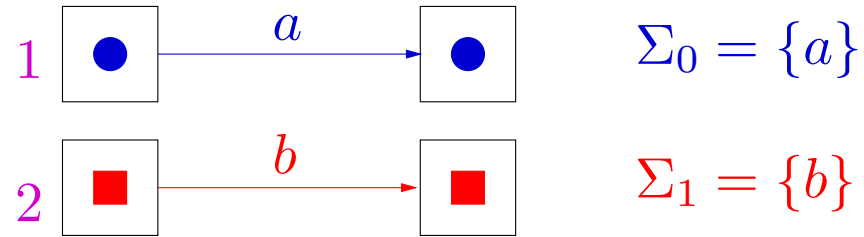
Neither team 0 nor team 1 has a winning distributed strategy.

Goal: determine whether team 0 (the system) has a winning distributed strategy.

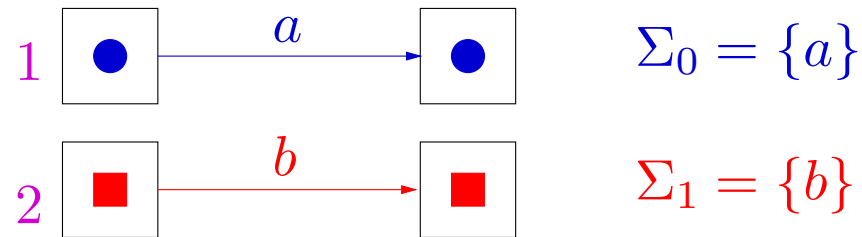
$$\mathcal{W} = \{t \in \mathbb{R}(\Sigma', D) \mid |t|_a = 7\}$$

A WDS for team 0 has to advise a to **stop** after 7 moves.

Distributed strategies: maximality condition



Distributed strategies: maximality condition



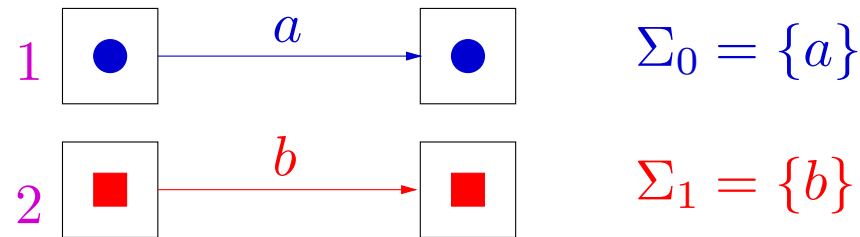
$$\mathcal{W} = \{t \in \mathbb{R}(\Sigma', D) \mid |t|_a = 7\}$$

We want the WDS f for team 0 advising exactly 7 a 's to be winning.

Moves of Σ_1 cannot block Σ_0 since a and b are independent.

The play $a^6 b^\omega$ is not f -maximal.

Distributed strategies: maximality condition

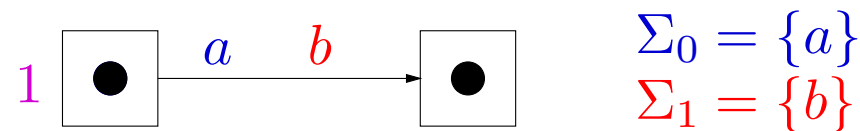


$$\mathcal{W} = \{t \in \mathbb{R}(\Sigma', D) \mid |t|_a = 7\}$$

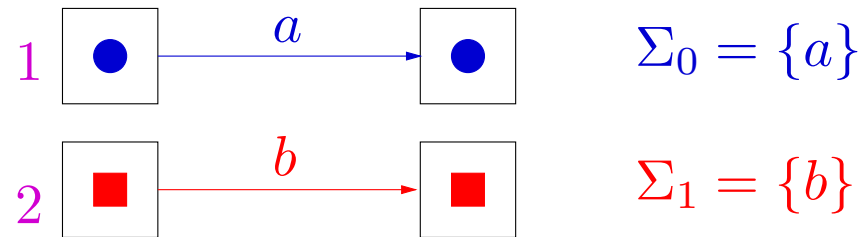
We want the WDS f for team 0 advising exactly 7 a 's to be winning.

Moves of Σ_1 cannot block Σ_0 since a and b are independent.

The play $a^6 b^\omega$ is not f -maximal.



Distributed strategies: maximality condition

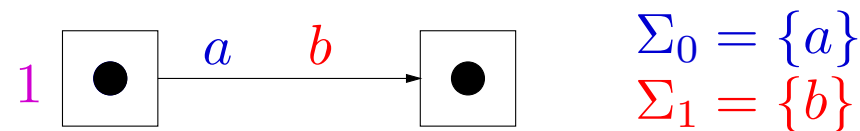


$$\mathcal{W} = \{t \in \mathbb{R}(\Sigma', D) \mid |t|_a = 7\}$$

We want the WDS f for team 0 advising exactly 7 a 's to be winning.

Moves of Σ_1 cannot block Σ_0 since a and b are independent.

The play $a^6 b^\omega$ is not f -maximal.



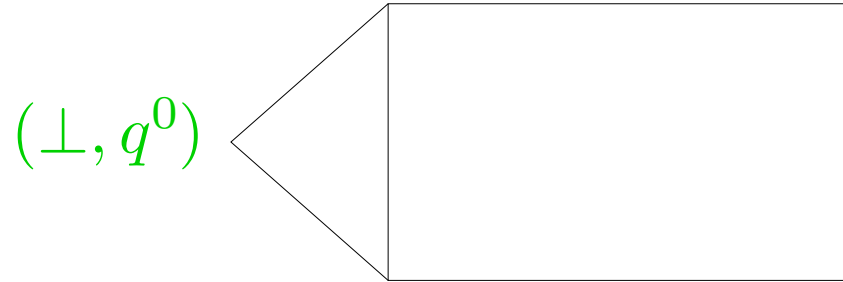
With this architecture (and same \mathcal{W}), Σ_1 can prevent Σ_0 to play.

The play b^ω is f -maximal.

In an f -maximal play, a move cannot be ultimately enabled by f .

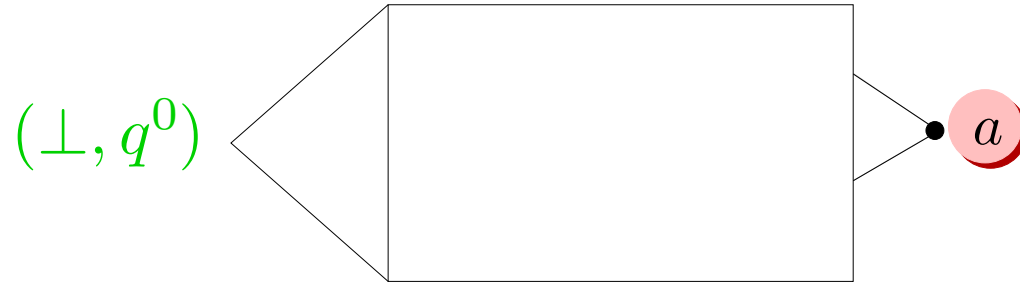
Memoryless distributed strategies

Coarse abstraction of the perfect memory:



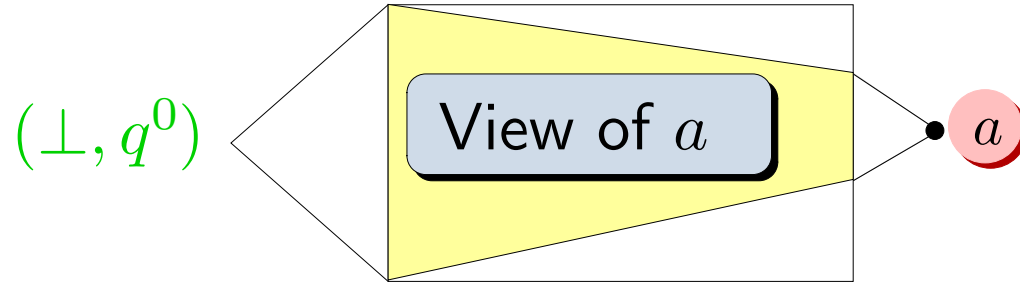
Memoryless distributed strategies

Coarse abstraction of the perfect memory:



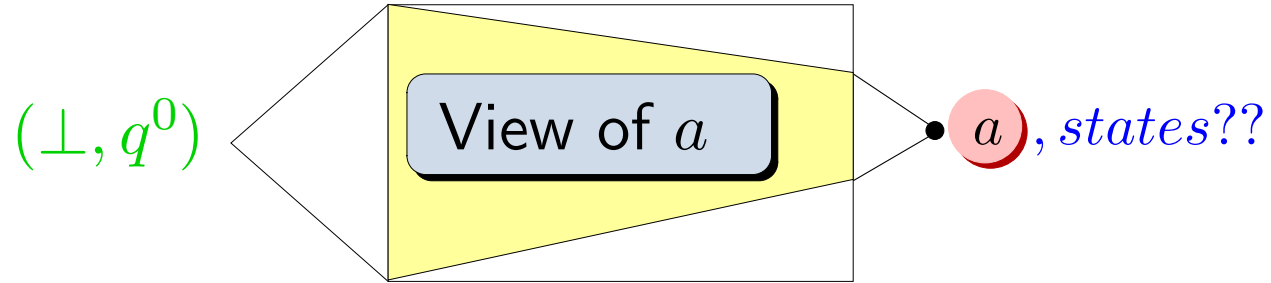
Memoryless distributed strategies

Coarse abstraction of the perfect memory:



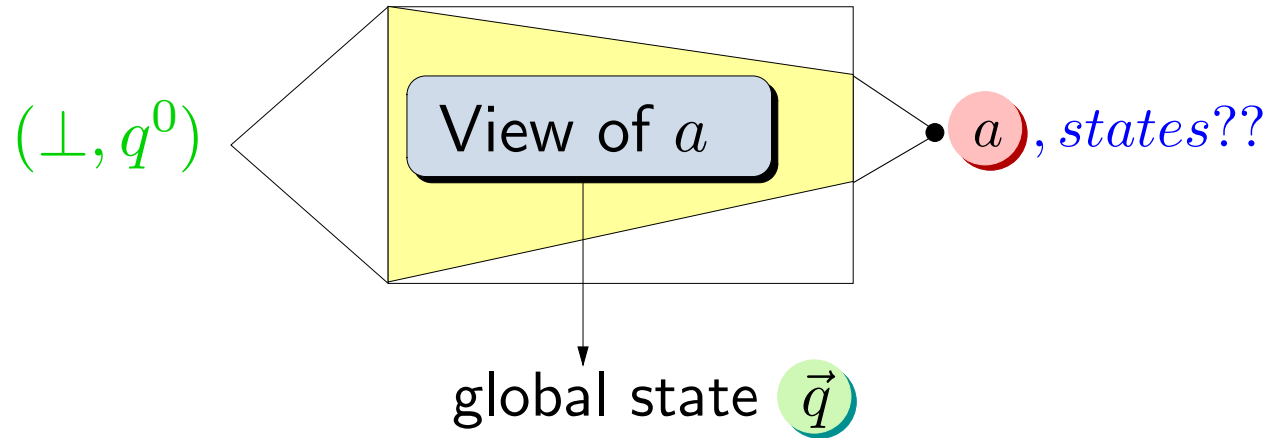
Memoryless distributed strategies

Coarse abstraction of the perfect memory:



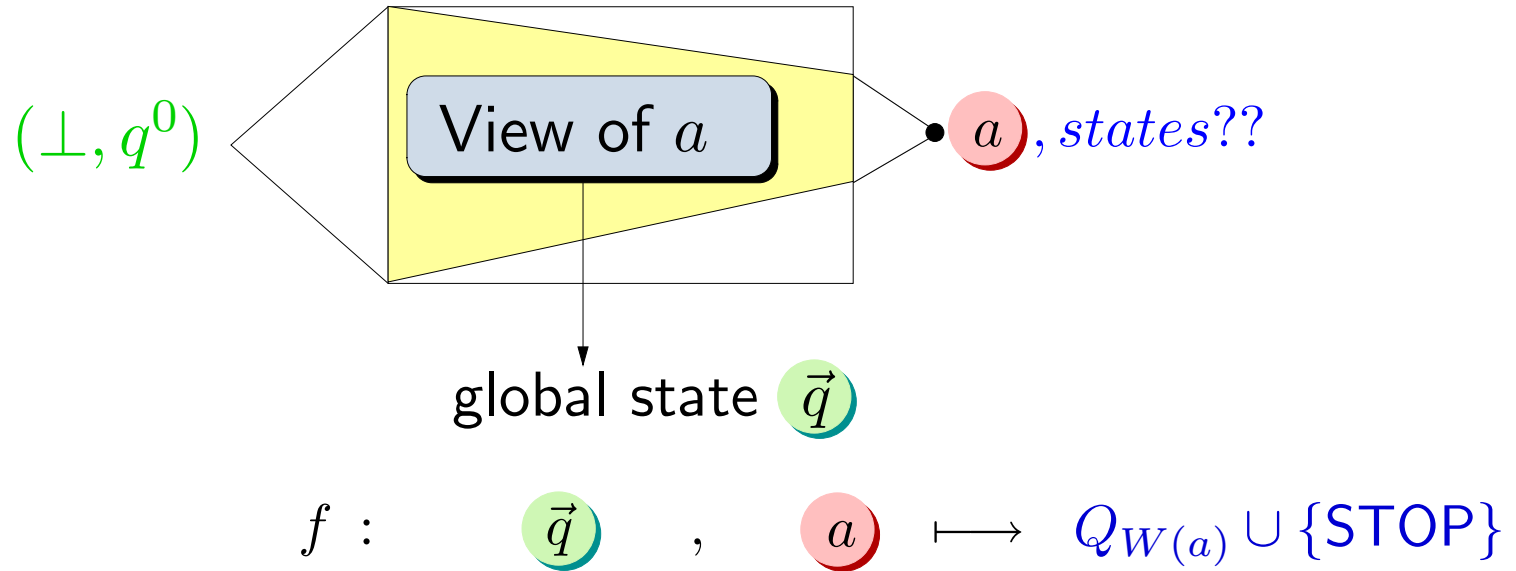
Memoryless distributed strategies

Coarse abstraction of the perfect memory:



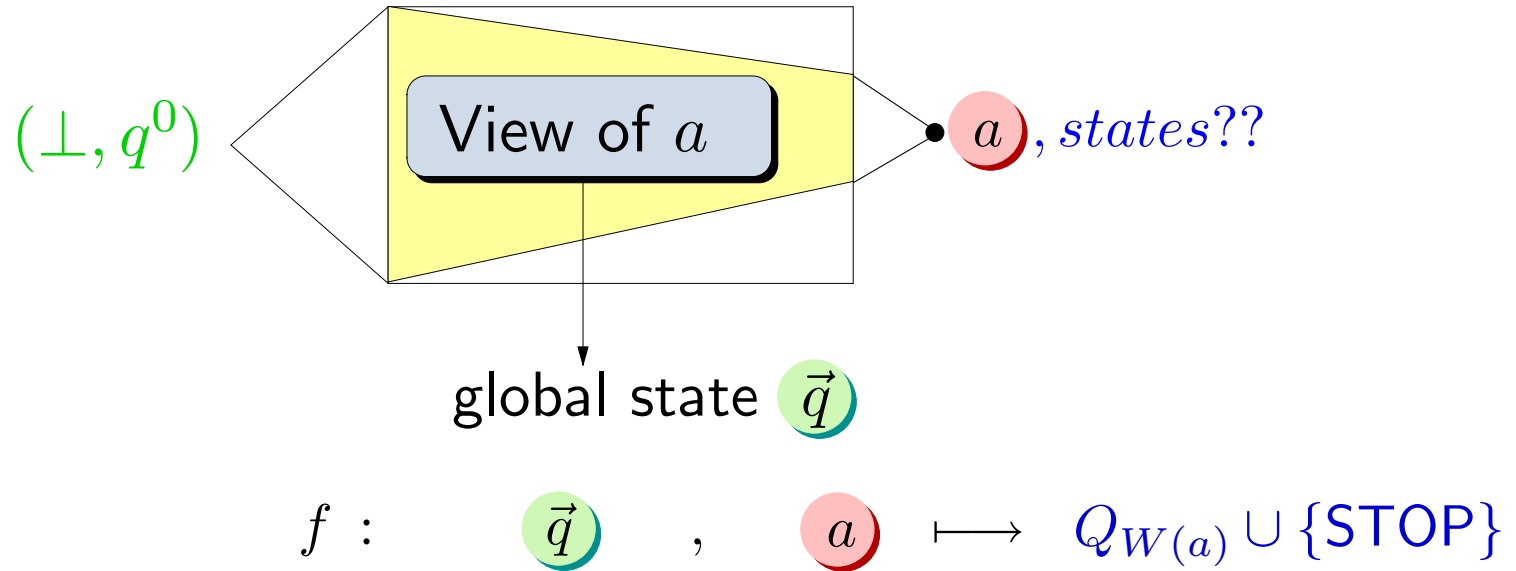
Memoryless distributed strategies

Coarse abstraction of the perfect memory:



Memoryless distributed strategies

Coarse abstraction of the perfect memory:



Finer abstractions of perfect memory \rightarrow distributed memories μ and strategies.

Proposition For a distributed game G and a distributed memory μ , one can build a game G^μ such that team 0 has a WDS in G with memory μ iff it has a memoryless WDS in G^μ .

Proof $G^\mu = G \times \mu$.

Existence of distributed strategies: undecidability

Proposition Deciding whether team 0 has a WDS is undecidable (even for rational winning conditions).

Proof. $\text{Rat}(\Sigma, D)$: trace languages of the form $[R]$, $R \in \text{Rat}(\Sigma^*)$.

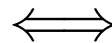
Undecidable: whether $\mathcal{L} \in \text{Rat}(A^* \times B^*)$ equals to $A^* \times B^*$.

$$(|A|, |B| \geq 2)$$

Let $G_{\mathcal{L}}$ be the following game:

- Team 0 has no players
- Team 1 has players $a_i \in A$ and $b_j \in B$ with $D = A^2 \uplus B^2$.
- $\mathcal{W} = \mathcal{L} \cup \{\text{infinite traces}\}$.

Team 0 has a (memoryless/perfect memory/arbitrary) WDS in $G_{\mathcal{L}}$



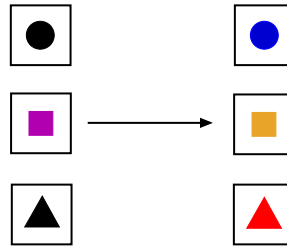
$$\mathcal{L} = A^* \times B^*$$

Asynchronous distributed games & classical 2-players sequential games

Deciding WMDS for simple winning conditions

- Finite number of winning memoryless distributed strategies.
- Deciding whether there is such a WMDS is decidable for recognizable \mathcal{W} .
- Is it possible to reduce the problem in the setting of sequential games?

Global game = (global states, E)

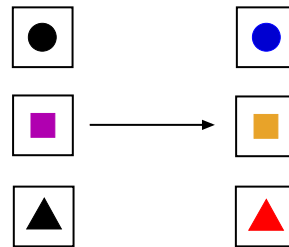


$$\vec{p} \xrightarrow{a} \vec{q} \in E \text{ if } (\vec{p}_{R(a)}, \vec{q}_{W(a)}) \in T_a \text{ and } \vec{q}_{\mathcal{P} \setminus W(a)} = \vec{p}_{\mathcal{P} \setminus W(a)}$$

Deciding WMDS for simple winning conditions

- Finite number of winning memoryless distributed strategies.
- Deciding whether there is such a WMDS is decidable for recognizable \mathcal{W} .
- Is it possible to reduce the problem in the setting of sequential games?

Global game = (global states, E)



$$\vec{p} \xrightarrow{a} \vec{q} \in E \text{ if } (\vec{p}_{R(a)}, \vec{q}_{W(a)}) \in T_a \text{ and } \vec{q}_{\mathcal{P} \setminus W(a)} = \vec{p}_{\mathcal{P} \setminus W(a)}$$

Fact Global game not adequate: may be determined while the ADG is not.
winning strategies might not be distributed

Aim for using efficient algorithms of the sequential case

From an ADG G , build a classical 2-players game \tilde{G} st.

$$\exists \text{ WMDS for team 0 in } G \iff \exists \text{ WS for player 0 in } \tilde{G}$$

Global game \tilde{G}

Player 1 wins if it shows that player 0

- does not reach \mathcal{W} , or
- does not follow a **distributed** strategy.

Player 1 can

- decide which actions are used and in which order.
- investigate all possible linearizations of distributed plays with **reset moves**

Positions: $Z = \underbrace{Q \times \Sigma_0}_{Z_0} \uplus \underbrace{Q \times (\Sigma_1 \cup \{0, 1, 2\})}_{Z_1}$, player i plays on Z_i .

- $(q, a) \in Z$
- q stores the global state;
 - $a \in \Sigma_i$: player i has to play a . Player 0 may **refuse** the move
 - $a = 0$: a reset move has just be performed by player 1.
 - $a = 1$: coming from (p, b) , the b -move has been performed
 - $a = 2$: player 0 refused the previous move.

Global game \tilde{G} , transitions

Initial position $(q^0, 0) \in Z_1$.

Set of moves $T \subseteq (Z_0 \times Z_1) \cup (Z_1 \times Z)$:

- $(p, b) \rightarrow (p, a)$ with $b \in \{0, 1, 2\}$ and $a \in \Sigma$. Player 1 requests an a -move.
- $(q, b) \rightarrow (q^0, 0)$ with $b \in \{0, 1, 2\}$. **Reset move** used by player 1 to show that player 0 is not following a **distributed** strategy.
- $(p, a) \rightarrow (q, 1)$ with $a \in \Sigma$, $(p_{R(a)}, q_{W(a)}) \in T_a$ and $q_{\mathcal{P} \setminus W(a)} = p_{\mathcal{P} \setminus W(a)}$. a -move executed by player 0 or 1 depending on whether $a \in \Sigma_0$ or $a \in \Sigma_1$.
- $(p, a) \rightarrow (p, 2)$ with $a \in \Sigma_0$. Player 0 **refuses** to make an a -move.

Global play: sequence $z = z_0 z_1 z_2 \cdots \in Z^\infty$ with $z_0 = (q^0, 0)$ and $z_n \rightarrow z_{n+1}$.

Consistency and fairness

$z = z_0 z_1 z_2 \cdots \in Z^\infty$ play in \tilde{G} with $z_n = (q^n, a_n) \in Z$.

- z is **consistent** if $\forall j, k$ ($a_j = a_k = a \in \Sigma_0$ and $q_{R(a)}^j = q_{R(a)}^k$)
 $\implies (a_{j+1} = a_{k+1}$ and $q_{W(a)}^{j+1} = q_{W(a)}^{k+1}$).

With the same (memoryless) view, player 0 plays the same.

- z is **fair** if $|\{n \geq 0 \mid a_n = 0\}| < \infty$ and $\forall a \in \Sigma_0, |\{n \geq 0 \mid a_n = a\}| = \infty$.

Finite numbers of resets and all letters of Σ_0 proposed infinitely many times

Consistency and fairness

$z = z_0 z_1 z_2 \cdots \in Z^\infty$ play in \tilde{G} with $z_n = (q^n, a_n) \in Z$.

- z is **consistent** if $\forall j, k$ ($a_j = a_k = a \in \Sigma_0$ and $q_{R(a)}^j = q_{R(a)}^k$)
 $\implies (a_{j+1} = a_{k+1}$ and $q_{W(a)}^{j+1} = q_{W(a)}^{k+1}$).

With the same (memoryless) view, player 0 plays the same.

- z is **fair** if $|\{n \geq 0 \mid a_n = 0\}| < \infty$ and $\forall a \in \Sigma_0, |\{n \geq 0 \mid a_n = a\}| = \infty$.

Finite numbers of resets and all letters of Σ_0 proposed infinitely many times

Distributed plays $(t_n)_{n \geq 0}$ in G associated with z :

- If $z_n = (q^0, 0)$ then $t_n = (\perp, q^0)$. Reset
- If $a_{n+1} \in \Sigma \cup \{2\}$ then $t_{n+1} = t_n$. Propose/refuse
- if $a_n = a \in \Sigma$ and $a_{n+1} = 1$ then $t_{n+1} = t_n \cdot (a, q_{W(a)}^{n+1})$. Accept

Lemma t_n is a distributed play reaching global state $\bar{\sigma}(t_n) = q^n$.

If z is consistent and fair, then $(t_n)_{n \geq \max\{n \mid a_n = 0\}}$ is increasing and its least upper bound $t(z)$ is a distributed play of G .

From a distributed game to its associated global game

Winning condition $\tilde{\mathcal{W}}$ of \tilde{G} only involves infinite plays $z \in Z^\omega$.

- If z is **not consistent** then **player 0 loses** the game.
He does not mimic a memoryless **distributed** strategy.
- Else if z is **not fair** then **player 1 loses** the game.
- If z is both **consistent and fair** then player 0 wins the game iff $t(z) \in \mathcal{W}$.

Proposition The following conditions are equivalent for a distributed game G :

1. There exists a WMDS for team 0 in the distributed game G .
2. There exists a WMS for player 0 in the global game \tilde{G} .
3. There exists a WS for player 0 in the global game \tilde{G} .

Global game: example

2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$

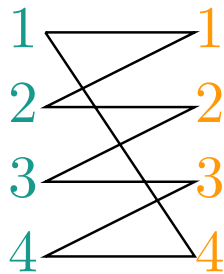
$\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$

Plays have the following shape: (\perp, q^0)

```

    graph LR
      A["(⊥, q0)"] --> B["a1"]
      A --> C["b1"]
      B --> D["a0"]
      C --> E["b0"]
      B --- F["Σ1"]
      C --- G["Σ0"]
  
```

$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$



Global game: example

2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$

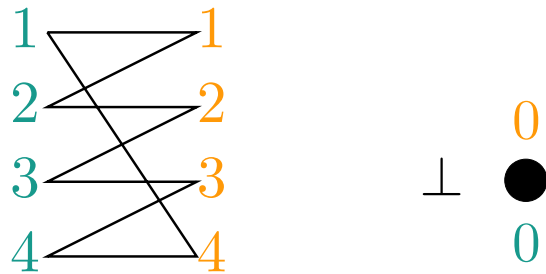
$\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$

Plays have the following shape: (\perp, q^0)

```

    graph LR
      A["(⊥, q0)"] -- "Σ1" --> B["a1"]
      A -- "Σ0" --> C["b1"]
      B --> D["a0"]
      C --> E["b0"]
  
```

$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$

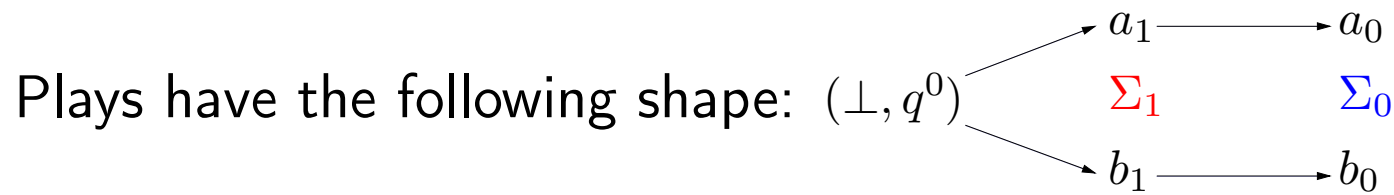


Global game: example

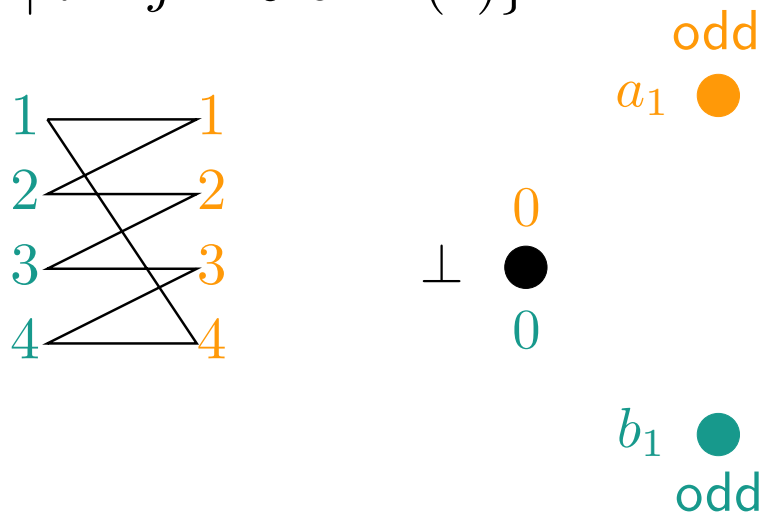
2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$
 $\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$



Global game: example

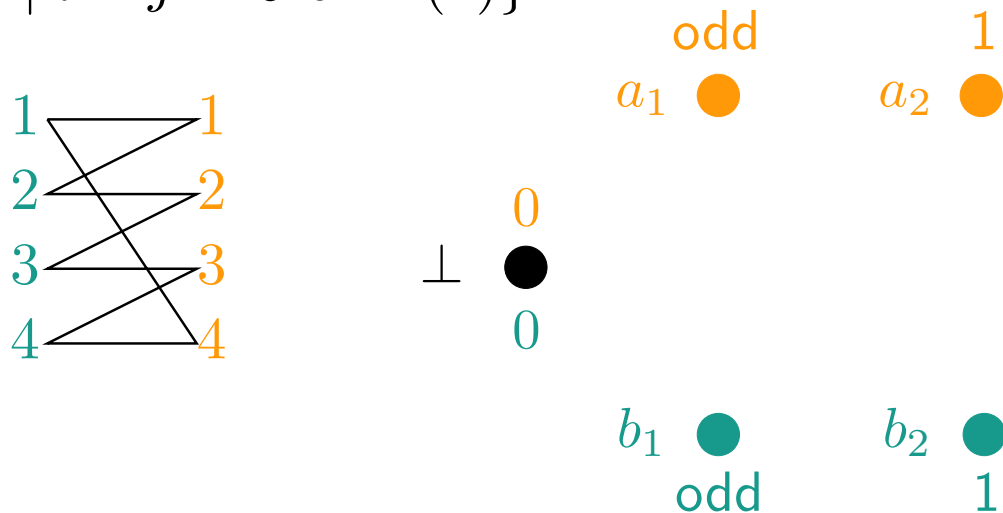
2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$
 $\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$

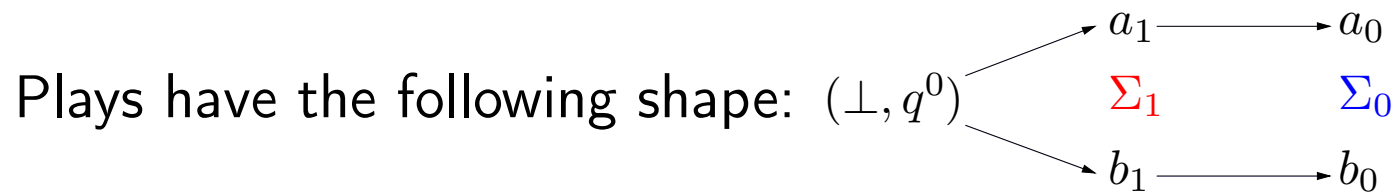


Global game: example

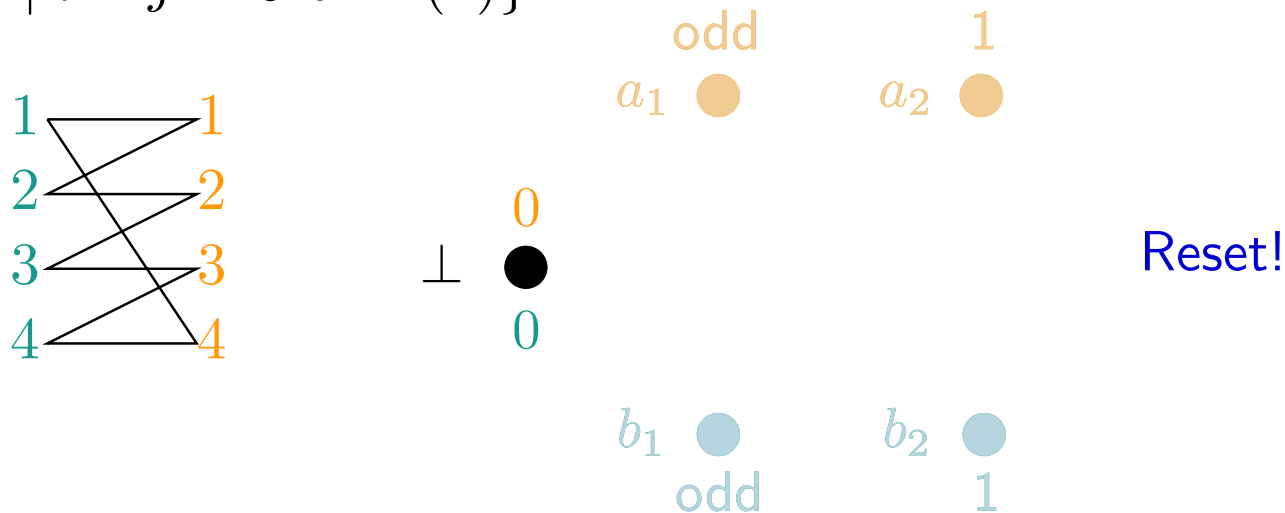
2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$
 $\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$

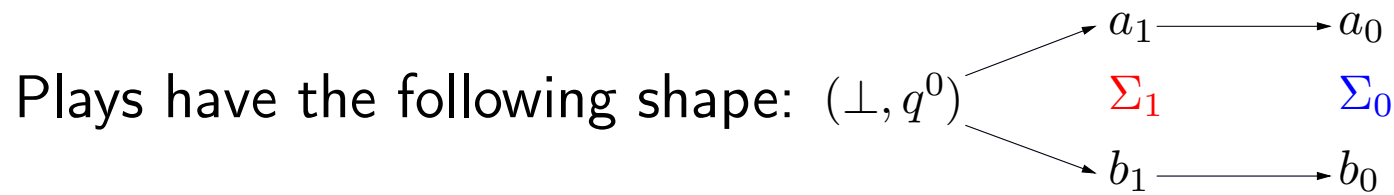


Global game: example

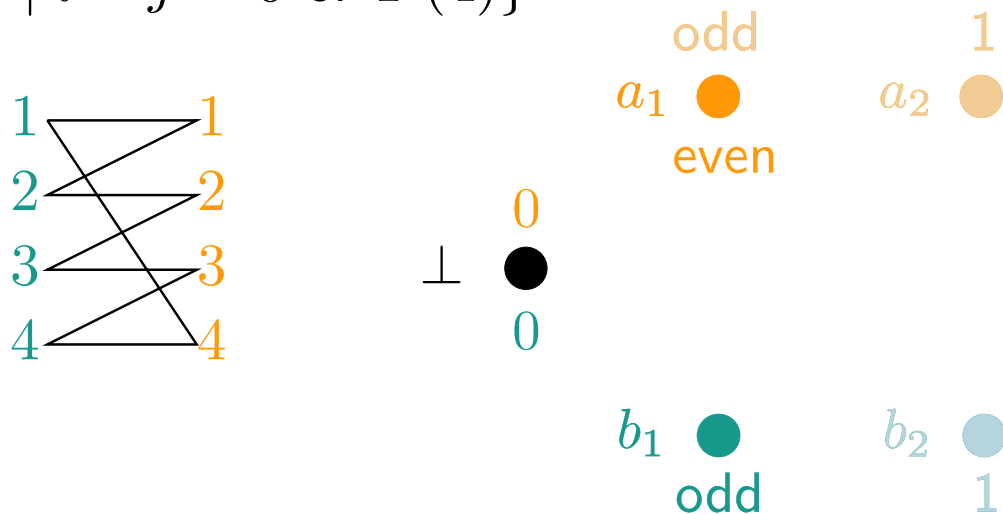
2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$
 $\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$

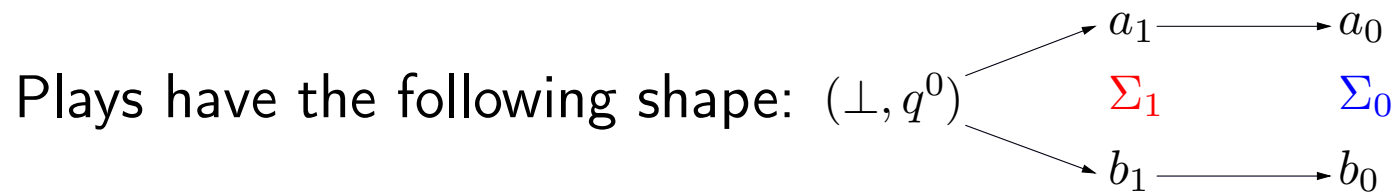


Global game: example

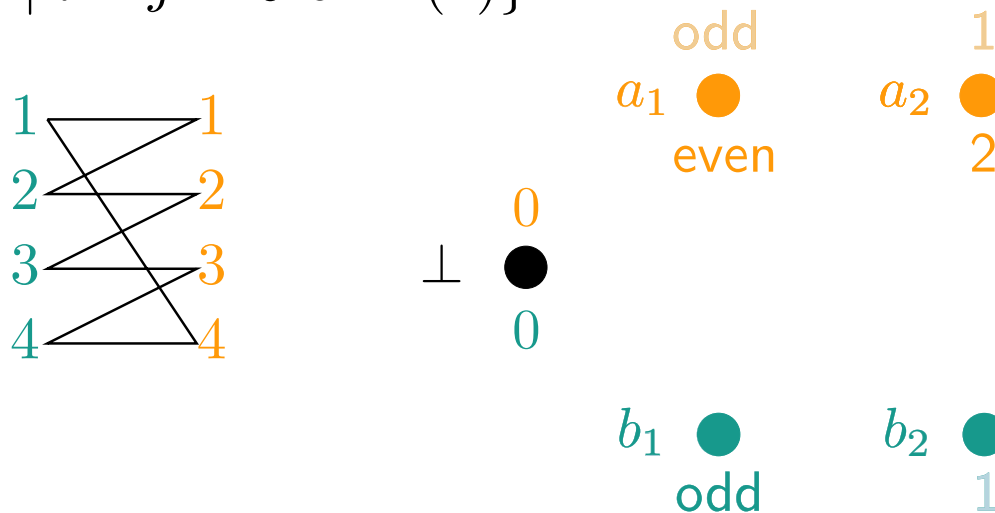
2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$
 $\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$

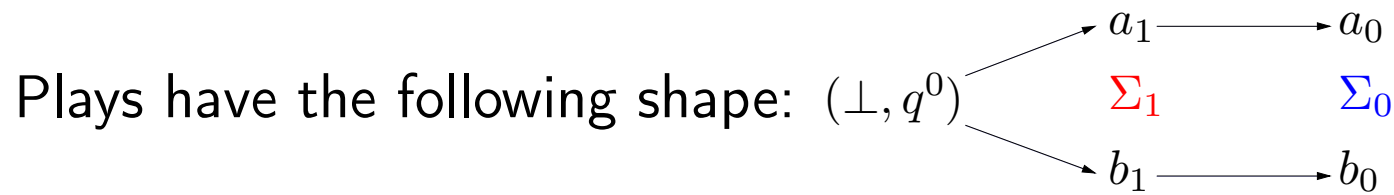


Global game: example

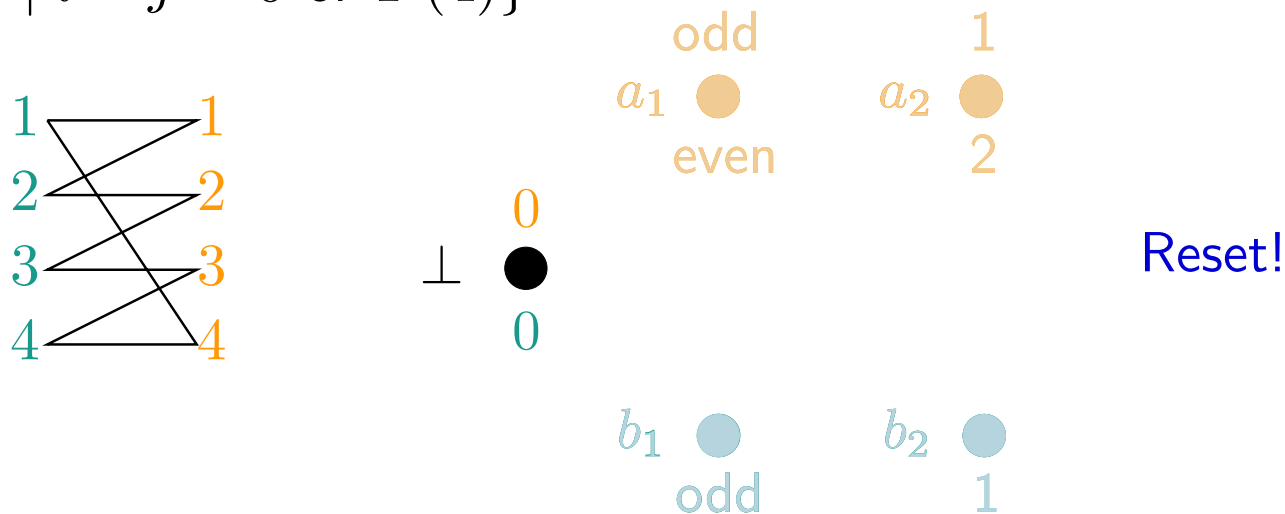
2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$
 $\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$



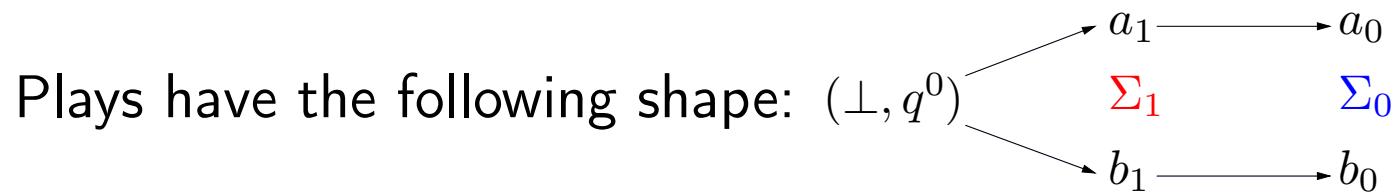
Global game: example

2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

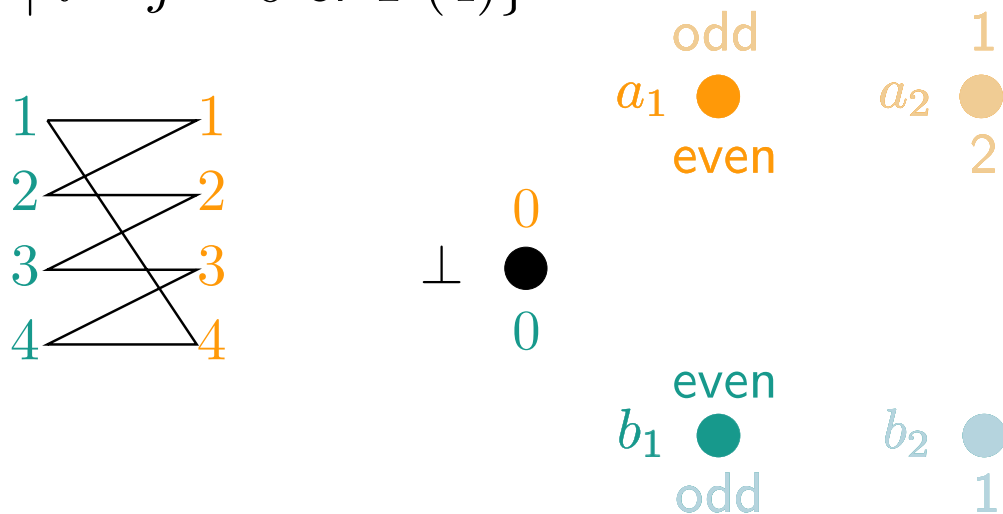
Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$

$\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$

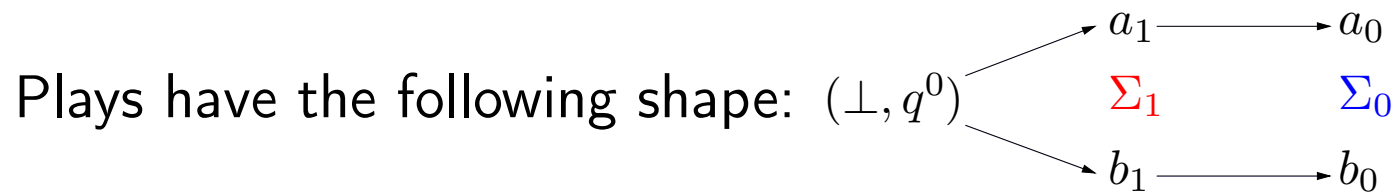


Global game: example

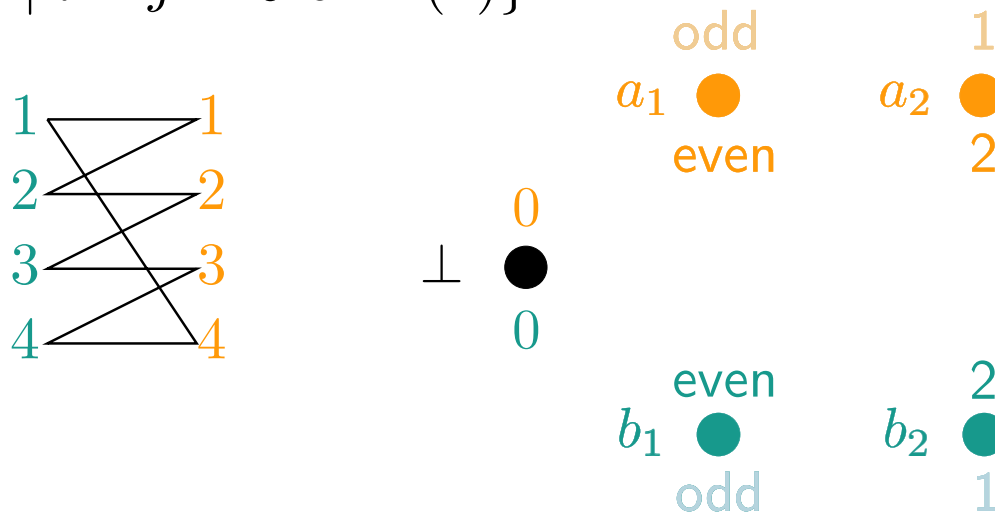
2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$
 $\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$

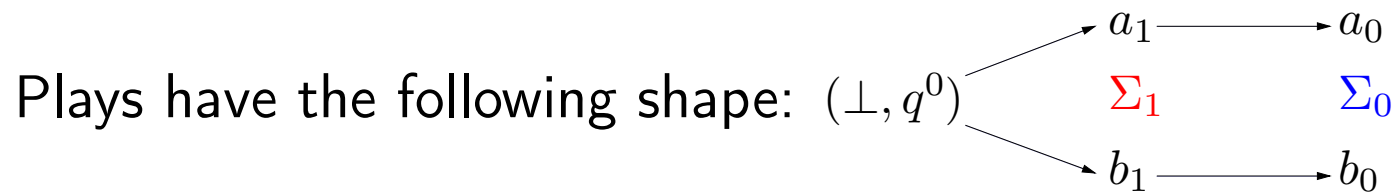


Global game: example

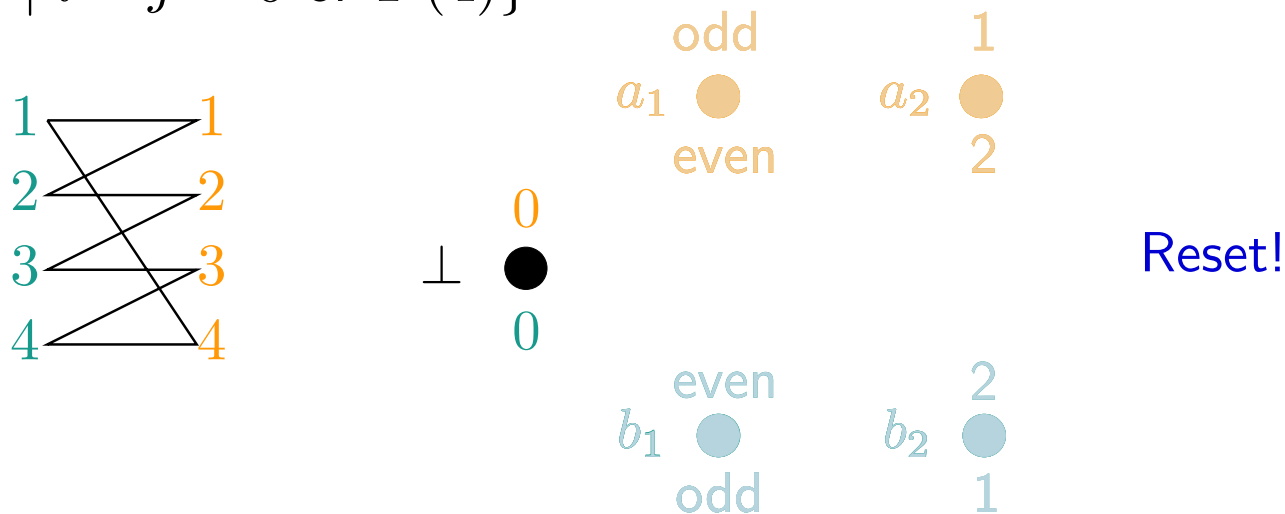
2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$
 $\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$



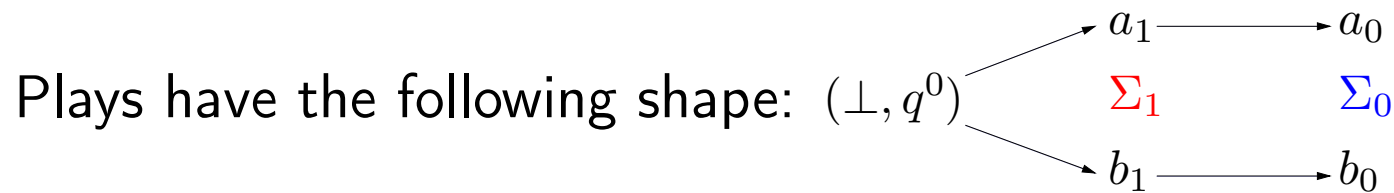
Global game: example

2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

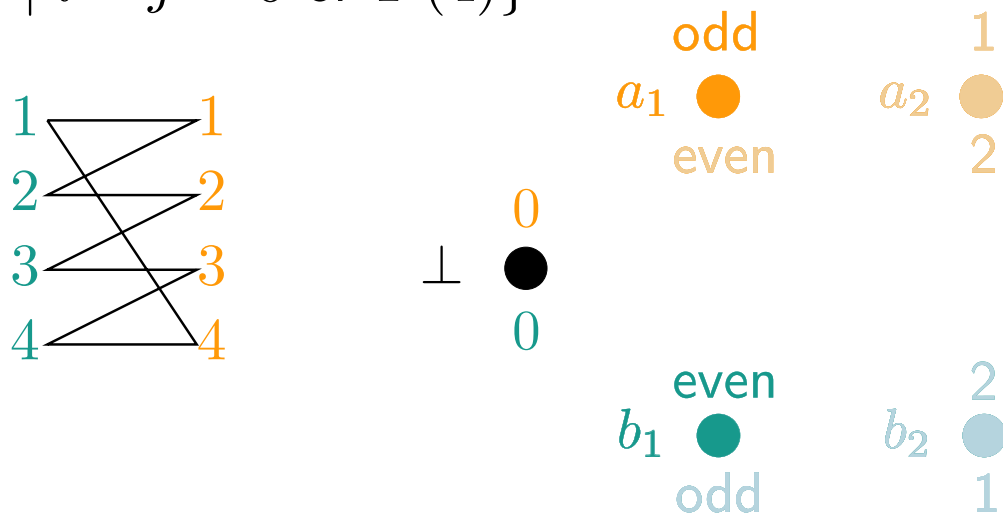
Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$

$\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$

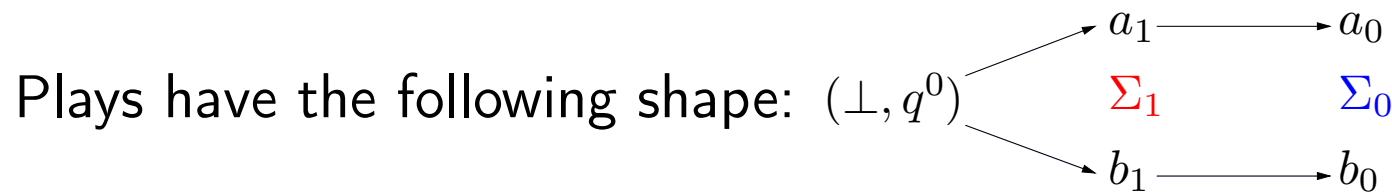


Global game: example

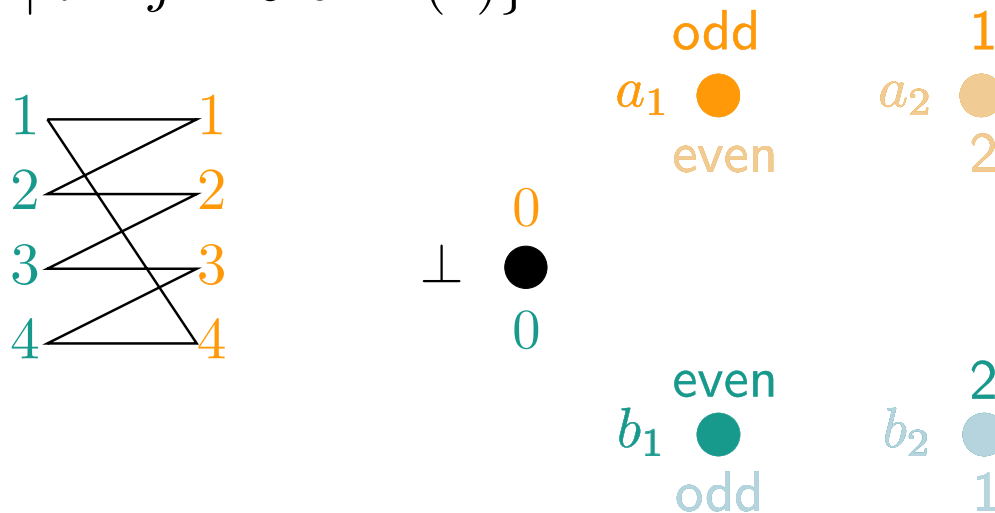
2 processes 1, 2. $\Sigma_i = \{a_i, b_i\}$, $R(a_i) = W(a_i) = 1$ and $R(b_i) = W(b_i) = 2$.

Σ_1 -transitions: $q^0 \xrightarrow{a_1} \text{odd} \vee \text{even}$ and $q^0 \xrightarrow{b_1} \text{odd} \vee \text{even}$

Σ_0 -transitions: $\text{odd} \xrightarrow{a_0} 1 \vee 3$, $\text{even} \xrightarrow{a_0} 2 \vee 4$
 $\text{odd} \xrightarrow{b_0} 1 \vee 3$, $\text{even} \xrightarrow{b_0} 2 \vee 4$



$$\mathcal{W} = \{(i, j) \mid i - j \equiv 0 \text{ or } 1 \pmod{4}\}$$



Player 0 loses.

Remarks on reachability games

Several kinds of reachability games

- Global: reach global states vs. local: reach states of some process.
- Strict: Reach and stop vs. relaxed: just reach.

$$\mathcal{W} \in \text{Rec}(\mathbb{M}(\Sigma', D)) \quad \mathcal{W} \in \text{Rec}(\mathbb{M}(\Sigma', D)) \cdot \mathbb{R}(\Sigma', D)$$

Proposition If team 0 has a WDS in a reachability game, then it also has a DS with bounded memory.

Proof Diagonalization argument.

Proposition For each $M > 0$, there exist strict/relaxed reachability games in which Σ_0 needs a memory of size (not less than) M . In particular, the following memories are not sufficient:

- ★ Global state of the view of a ;
- ★ Timestamping dags.

Memory in reachability games

In sequential games, no memory needed. Still the case here?

Game $G_{K,M}$. $\Sigma_0 = \{a, b\}$ $\Sigma_1 = \{e^+, e^-\}$. R & W

x_1	x_2	$x_3 = (k, m)$	x_4	x_5
-------	-------	----------------	-------	-------

e^+, e^- only write $x_3 \in \mathbb{N} \times \mathbb{Z}/M\mathbb{Z}$

$e^+ : k++ \ \& \ m++$

$e^- : k++ \ \& \ m--$

1st move of Σ_1 non-deterministic on k

a blocks itself and e^+ with $x_2 = 1$

b blocks itself and e^- with $x_4 = 1$

Memory in reachability games

In sequential games, no memory needed. Still the case here?

Game $G_{K,M}$. $\Sigma_0 = \{a, b\}$ $\Sigma_1 = \{e^+, e^-\}$. R & W

x_1	x_2	$x_3 = (k, m)$	x_4	x_5
-------	-------	----------------	-------	-------

e^+, e^- only write $x_3 \in \mathbb{N} \times \mathbb{Z}/M\mathbb{Z}$

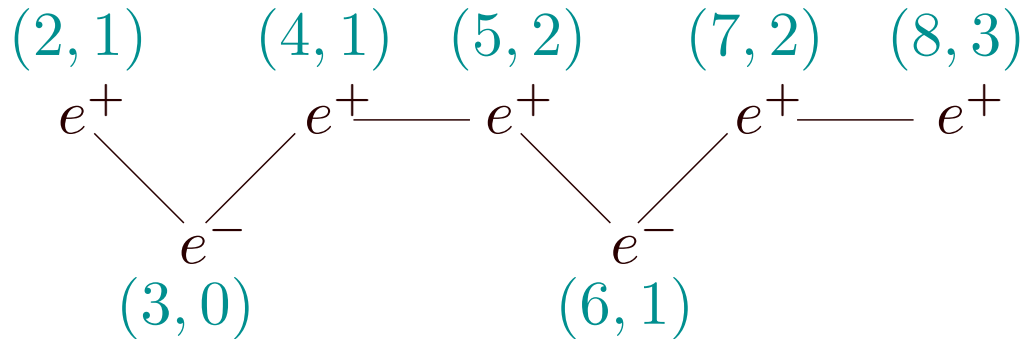
$e^+ : k++ \ \& \ m++$

$e^- : k++ \ \& \ m--$

1st move of Σ_1 non-deterministic on k

a blocks itself and e^+ with $x_2 = 1$

b blocks itself and e^- with $x_4 = 1$



Memory in reachability games

In sequential games, no memory needed. Still the case here?

Game $G_{K,M}$. $\Sigma_0 = \{a, b\}$ $\Sigma_1 = \{e^+, e^-\}$. R & W

x_1	x_2	$x_3 = (k, m)$	x_4	x_5
a		e^-		
e^+			b	

e^+, e^- only write $x_3 \in \mathbb{N} \times \mathbb{Z}/M\mathbb{Z}$

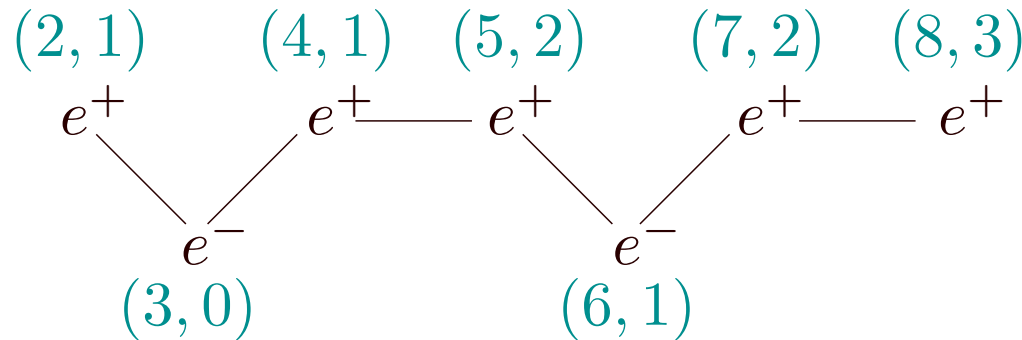
$e^+ : k++ \ \& \ m++$

$e^- : k++ \ \& \ m--$

1st move of Σ_1 non-deterministic on k

a blocks itself and e^+ with $x_2 = 1$

b blocks itself and e^- with $x_4 = 1$



$$\mathcal{W} = \{(x_i) \mid x_2 = x_4 = 1, k \leq K \text{ and } x_1 - x_5 = x_3 \pmod{M}\}$$

Memory in reachability games

In sequential games, no memory needed. Still the case here?

Game $G_{K,M}$. $\Sigma_0 = \{a, b\}$ $\Sigma_1 = \{e^+, e^-\}$. R & W

x_1	x_2	$x_3 = (k, m)$	x_4	x_5
a		e^-		
e^+			b	

e^+, e^- only write $x_3 \in \mathbb{N} \times \mathbb{Z}/M\mathbb{Z}$

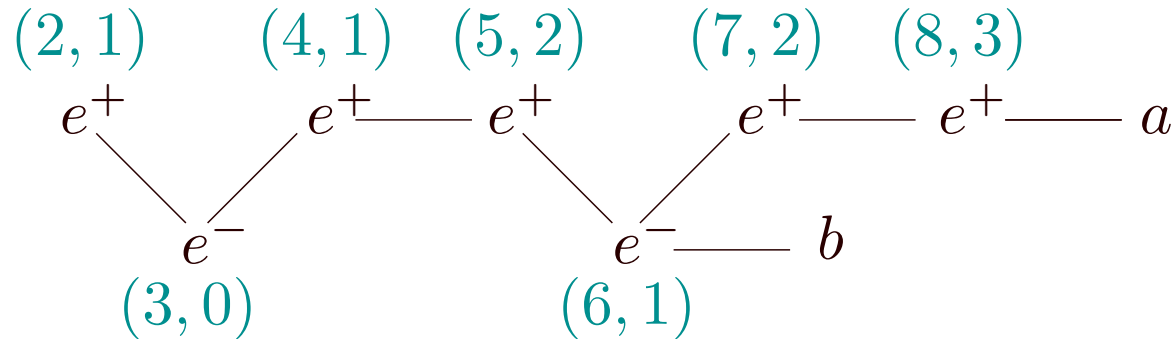
$e^+ : k++ \ \& \ m++$

$e^- : k++ \ \& \ m--$

1st move of Σ_1 non-deterministic on k

a blocks itself and e^+ with $x_2 = 1$

b blocks itself and e^- with $x_4 = 1$



$$\mathcal{W} = \{(x_i) \mid x_2 = x_4 = 1, k \leq K \text{ and } x_1 - x_5 = x_3 \pmod{M}\}$$

Memory in reachability games

In sequential games, no memory needed. Still the case here?

Game $G_{K,M}$. $\Sigma_0 = \{a, b\}$ $\Sigma_1 = \{e^+, e^-\}$. R & W

x_1	x_2	$x_3 = (k, m)$	x_4	x_5
-------	-------	----------------	-------	-------

e^+, e^- only write $x_3 \in \mathbb{N} \times \mathbb{Z}/M\mathbb{Z}$

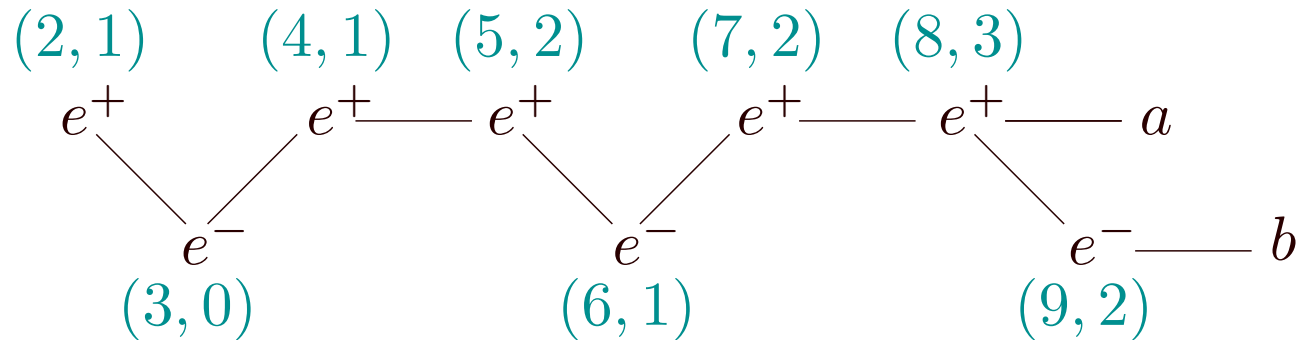
$e^+ : k++ \ \& \ m++$

$e^- : k++ \ \& \ m--$

1st move of Σ_1 non-deterministic on k

a blocks itself and e^+ with $x_2 = 1$

b blocks itself and e^- with $x_4 = 1$



$$\mathcal{W} = \{(x_i) \mid x_2 = x_4 = 1, k \leq K \text{ and } x_1 - x_5 = x_3 \pmod{M}\}$$

Memory in reachability games

In sequential games, no memory needed. Still the case here?

Game $G_{K,M}$. $\Sigma_0 = \{a, b\}$ $\Sigma_1 = \{e^+, e^-\}$. R & W

x_1	x_2	$x_3 = (k, m)$	x_4	x_5
a		e^-		
e^+			b	

e^+, e^- only write $x_3 \in \mathbb{N} \times \mathbb{Z}/M\mathbb{Z}$

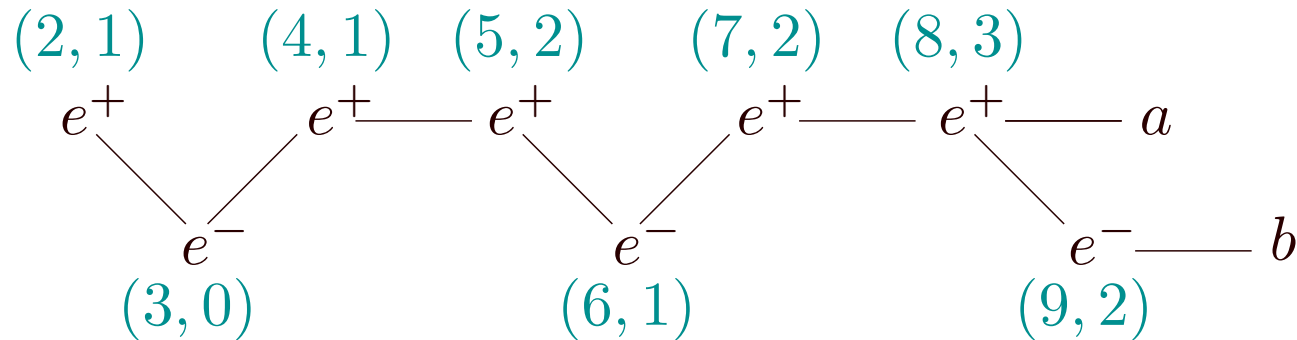
$e^+ : k++ \ \& \ m++$

$e^- : k++ \ \& \ m--$

1st move of Σ_1 non-deterministic on k

a blocks itself and e^+ with $x_2 = 1$

b blocks itself and e^- with $x_4 = 1$



$$\mathcal{W} = \{(x_i) \mid x_2 = x_4 = 1, k \leq K \text{ and } x_1 - x_5 = x_3 \pmod{M}\}$$

“Only” way for Σ_0 to win: count mod M .

Memory in reachability games

Strict & global reachability games



recognizable winning conditions

- In reachability games, memory can be **necessary** to win.
- Can we turn this into an undecidability result?
- **Conjecture** For series parallel alphabets, strict reachability games are **decidable**

Asynchronous alternating automata

Asynchronous automata. Motivation

- Strong relationships between games and automata.

- **Asynchronous** automata used for specification.

- **Alternating** automata can also be used for specification.

Compilation of LTL formulas into alternating automata.

- Some branching properties of distributed systems cannot be captured when working on linearizations.

One should be able to distinguish between

- non-determinism
- concurrency.

Alternating asynchronous automata (AAA)

Alternating automata: $\delta_a : Q_{R(a)} \rightarrow 2^{Q_{W(a)}}$

Alternating asynchronous automata: $\delta_a : Q_{R(a)} \rightarrow \mathcal{B}^+(Q_{W(a)})$

Runs: (prime) **well-formed** event structures.

$$R(a) = W(a) = \{1\}$$

$$\delta_a(0) = 1 \wedge 2$$

$$\delta_b(0) = 1 \wedge 2$$

$$R(b) = W(b) = \{2\}$$

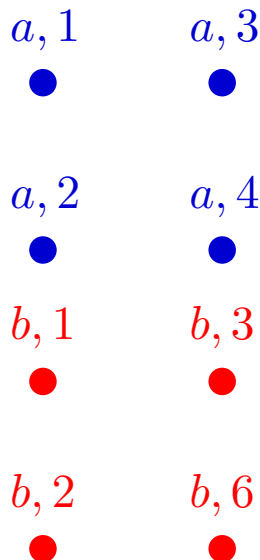
$$\delta_a(1) = 3 \vee 5$$

$$\delta_b(1) = 3 \vee 5$$

$$\delta_a(2) = 4 \vee 6$$

$$\delta_b(2) = 4 \vee 6$$

Run on *aabb*



Alternating asynchronous automata (AAA)

Alternating automata: $\delta_a : Q_{R(a)} \rightarrow 2^{Q_{W(a)}}$

Alternating asynchronous automata: $\delta_a : Q_{R(a)} \rightarrow \mathcal{B}^+(Q_{W(a)})$

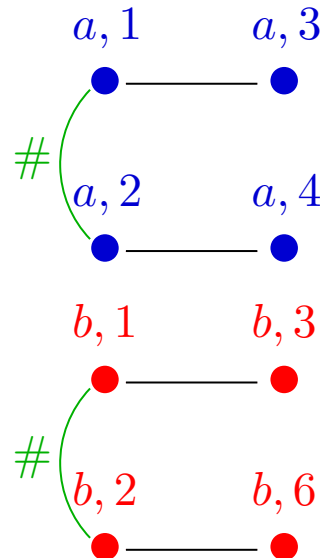
Runs: (prime) **well-formed** event structures.

$$R(a) = W(a) = \{1\} \quad \delta_a(0) = 1 \wedge 2 \quad \delta_b(0) = 1 \wedge 2$$

$$R(b) = W(b) = \{2\} \quad \delta_a(1) = 3 \vee 5 \quad \delta_b(1) = 3 \vee 5$$

$$\delta_a(2) = 4 \vee 6 \quad \delta_b(2) = 4 \vee 6$$

Run on *aabb*



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent $\delta_a(0) = 1 \wedge 2$ $\delta_b(0) = 1 \wedge 2$

$\delta_a(1) = 1 \vee 3$ $\delta_b(1) = 1 \vee 3$

$\delta_a(2) = 2 \vee 4$ $\delta_b(2) = 2 \vee 4$

Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

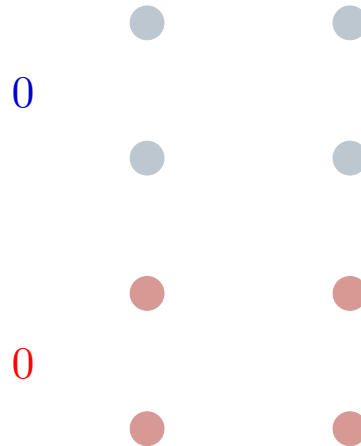
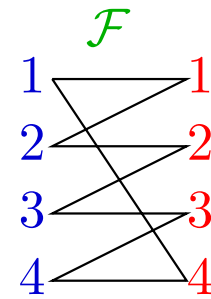
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

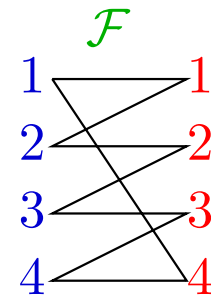
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

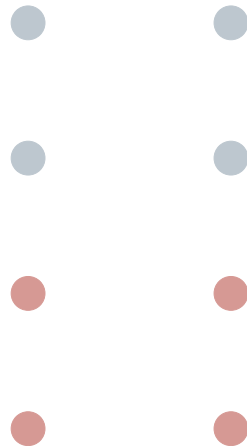
$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



0

0



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

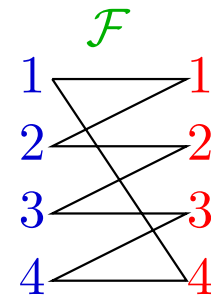
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



0



1



Player 0 loses



2



0



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

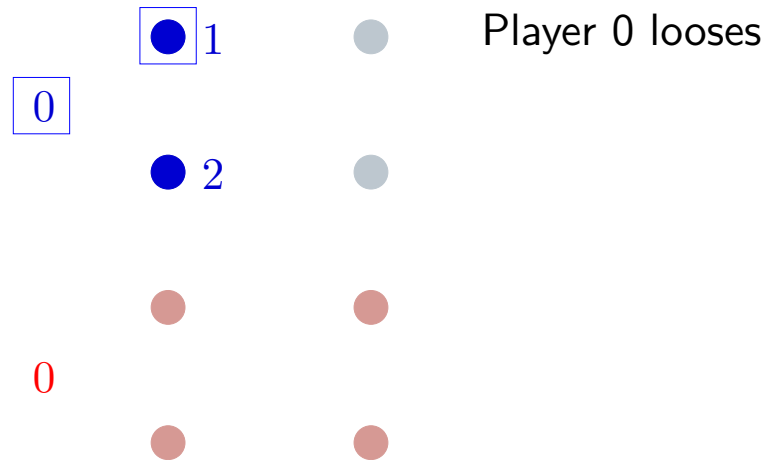
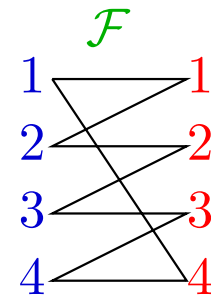
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

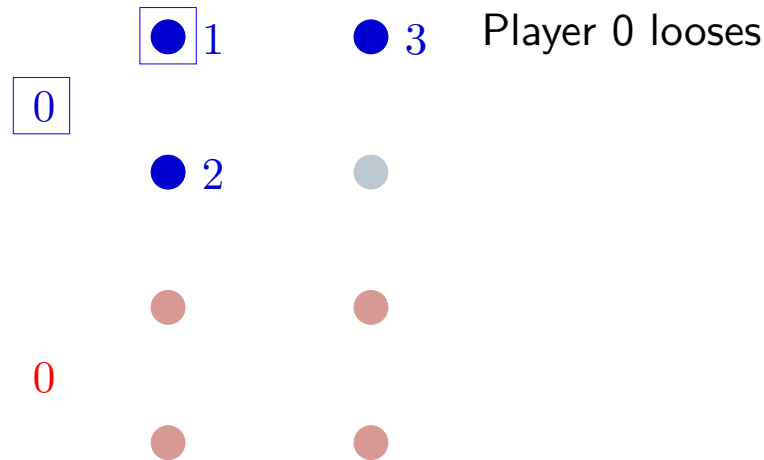
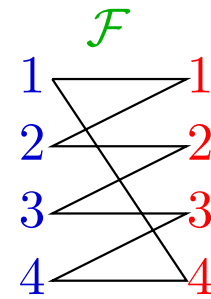
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

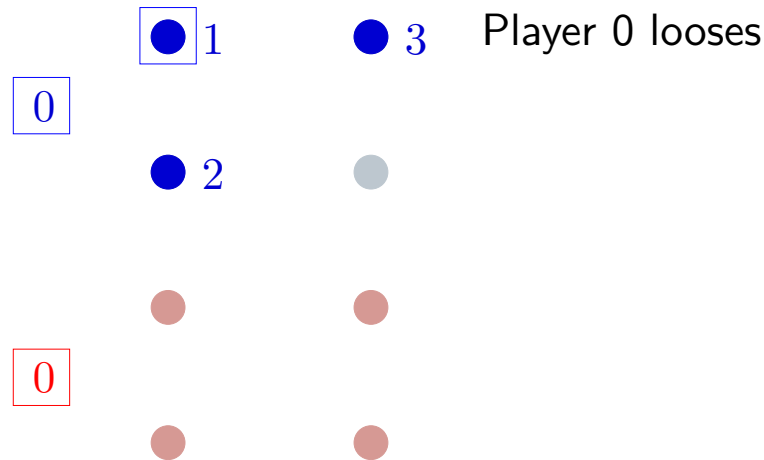
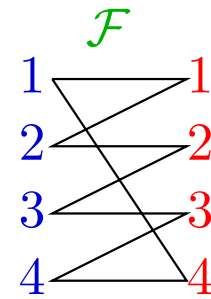
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

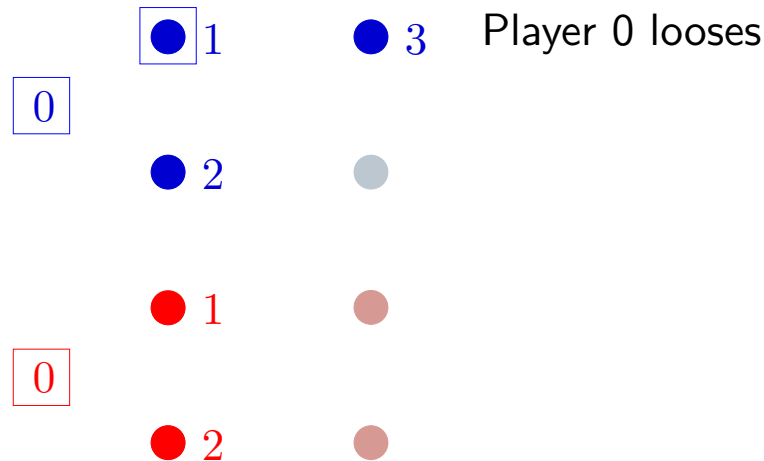
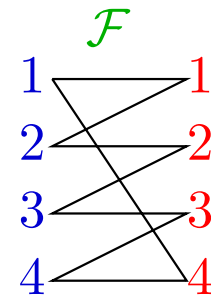
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

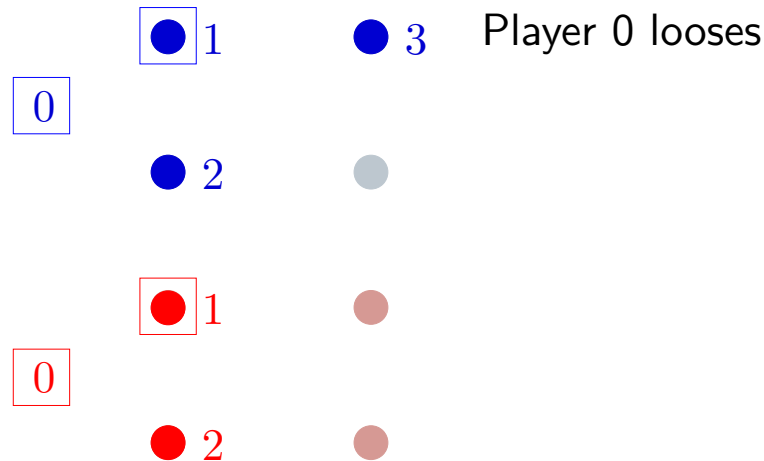
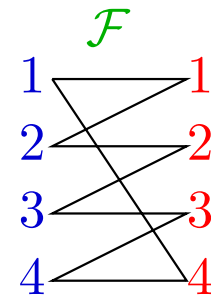
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

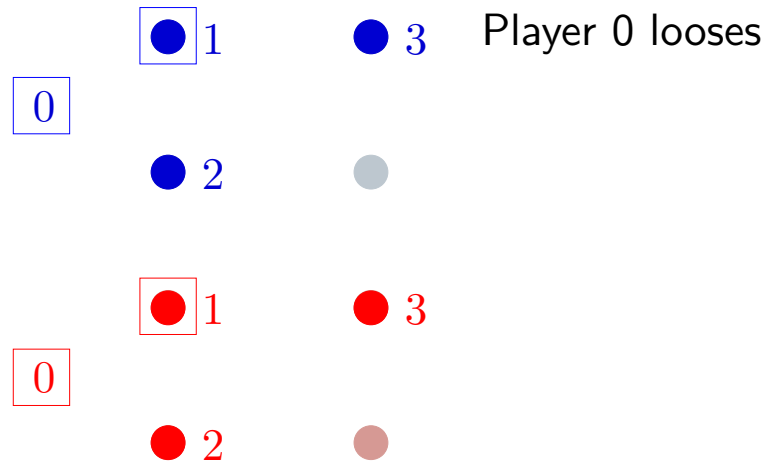
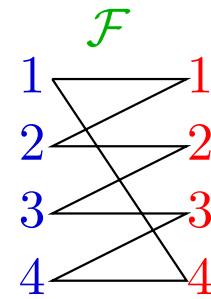
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

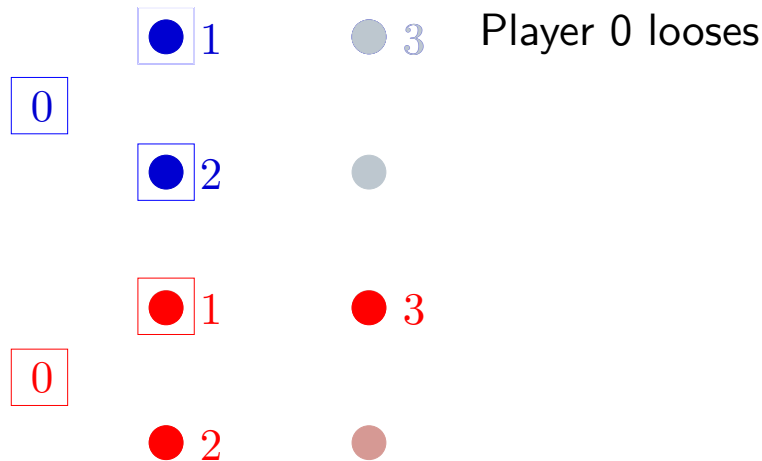
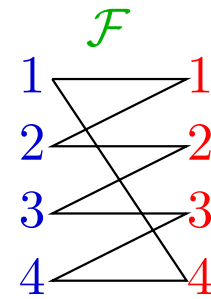
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

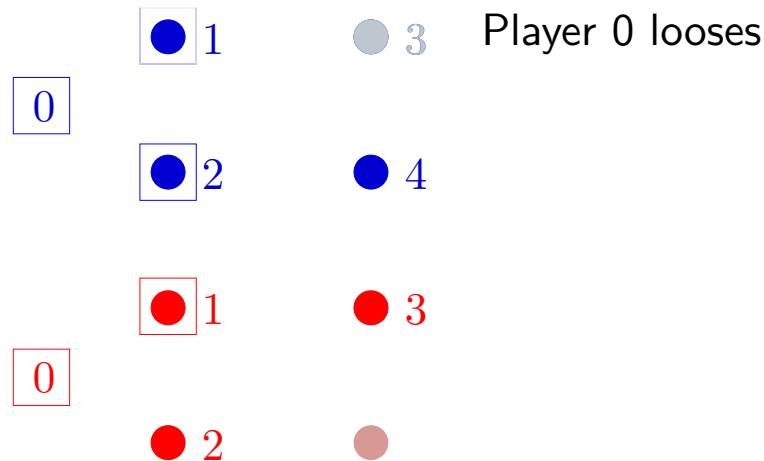
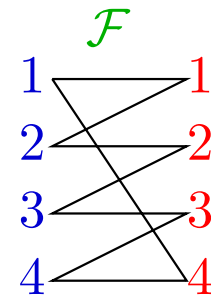
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

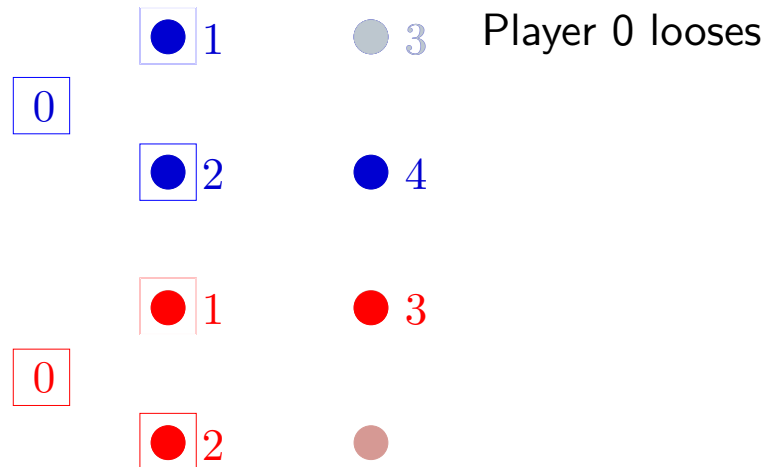
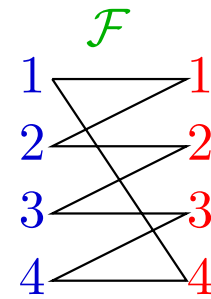
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

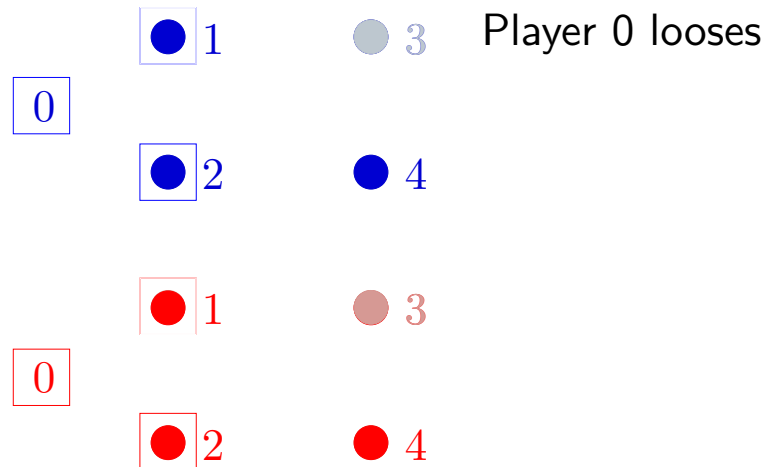
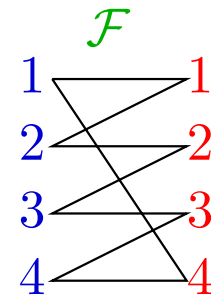
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Distributed game associated to an AAA

Prop Given an AAA \mathcal{A} and $t \in \mathbb{M}(\Sigma, D)$, \exists a **reachability** game $G(\mathcal{A}, t)$:

$t \in \mathcal{L}(\mathcal{A})$ iff team 0 has a WDS in $G(\mathcal{A}, t)$

a,b independent

$$\delta_a(0) = 1 \wedge 2$$

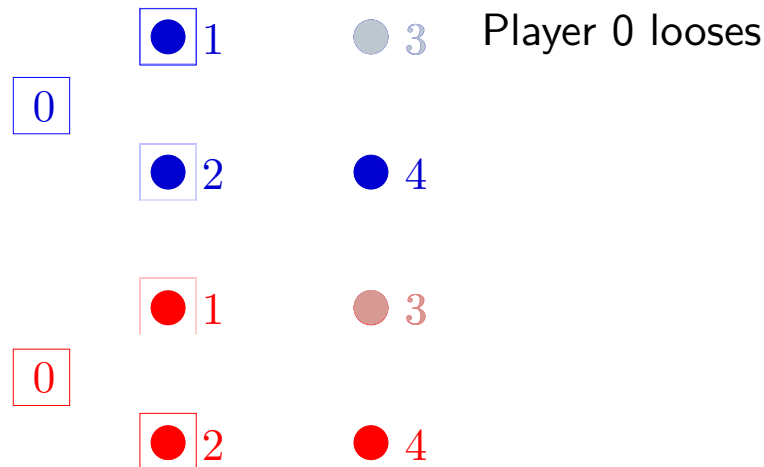
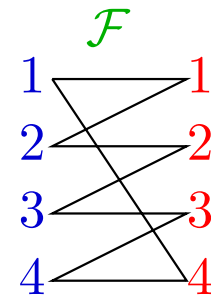
$$\delta_b(0) = 1 \wedge 2$$

$$\delta_a(1) = 1 \vee 3$$

$$\delta_b(1) = 1 \vee 3$$

$$\delta_a(2) = 2 \vee 4$$

$$\delta_b(2) = 2 \vee 4$$



Further work

- Are reachability games decidable for recognizable winning conditions?
- Same for infinite games.
- Look for winning conditions of low complexity (local conditions?)
- Optimal controller: permissive strategies (compare with Ramadge-Wonham)
- Relationship between other types of distributed games ([MT02] and [MW03]).
- Recognizability of languages accepted by an AAA?
- AAA for infinite traces.

Related work

References

- [1] Zielonka. Notes on finite asynchronous automata. [RAIRO ITA](#), 1987.
- [2] Zielonka. Asynchronous automata. [Book of Traces](#), 1995.
- [3] Pnueli & Rosner. Distributed reactive systems are hard to synthesize. [FOCS'90](#)
- [4] Madhusudan & Thiagarajan. Distributed controller synthesis for local specifications. [ICALP'01](#)
- [5] Madhusudan & Thiagarajan. Effective strategies for asynchronous distributed control. [CONCUR'02](#)
- [6] Mohalik & Walukiewicz. Distributed Games. [FSTTCS'03](#).
- [7] Bednarczyk. Hereditary history preserving bisimulations. TR, Gdansk, 1991.
- [8] Nielsen & Clausen. Games and logics for a noninterleaving bisimulation. *Nordic J. Comput.* 1995.
- [9] Jurdziński & Nielsen. Hereditary history preserving bisimilarity is undecidable. [STACS'00](#).
- [10] Mukund. Hereditary history preserving bisimulation is decidable for trace-labeled systems. [FSTTCS'02](#).
- [11] Ramadge & Wonham. The control of discrete event systems. [Proc IEEE](#), 1989.

References

- [1] Zielonka. Notes on finite asynchronous automata. [RAIRO ITA](#), 1987.
- [2] Zielonka. Asynchronous automata. [Book of Traces](#), 1995.
- [3] Pnueli & Rosner. Distributed reactive systems are hard to synthesize. [FOCS'90](#)
- [4] Madhusudan & Thiagarajan. Distributed controller synthesis for local specifications. [ICALP'01](#)
- [5] Madhusudan & Thiagarajan. Effective strategies for asynchronous distributed control. [CONCUR'02](#)
- [6] Mohalik & Walukiewicz. Distributed Games. [FSTTCS'03](#).
- [7] Bednarczyk. Hereditary history preserving bisimulations. TR, Gdansk, 1991.
- [8] Nielsen & Clausen. Games and logics for a noninterleaving bisimulation. *Nordic J. Comput.* 1995.
- [9] Jurdziński & Nielsen. Hereditary history preserving bisimilarity is undecidable. [STACS'00](#).
- [10] Mukund. Hereditary history preserving bisimulation is decidable for trace-labeled systems. [FSTTCS'02](#).
- [11] Ramadge & Wonham. The control of discrete event systems. [Proc IEEE](#), 1989.

References

- [1] Zielonka. Notes on finite asynchronous automata. [RAIRO ITA](#), 1987.
- [2] Zielonka. Asynchronous automata. [Book of Traces](#), 1995.
- [3] Pnueli & Rosner. Distributed reactive systems are hard to synthesize. [FOCS'90](#)
- [4] Madhusudan & Thiagarajan. Distributed controller synthesis for local specifications. [ICALP'01](#)
- [5] Madhusudan & Thiagarajan. Effective strategies for asynchronous distributed control. [CONCUR'02](#)
- [6] Mohalik & Walukiewicz. Distributed Games. [FSTTCS'03](#).
- [7] Bednarczyk. Hereditary history preserving bisimulations. TR, Gdansk, 1991.
- [8] Nielsen & Clausen. Games and logics for a noninterleaving bisimulation. *Nordic J. Comput.* 1995.
- [9] Jurdziński & Nielsen. Hereditary history preserving bisimilarity is undecidable. [STACS'00](#).
- [10] Mukund. Hereditary history preserving bisimulation is decidable for trace-labeled systems. [FSTTCS'02](#).
- [11] Ramadge & Wonham. The control of discrete event systems. [Proc IEEE](#), 1989.

References

- [1] Zielonka. Notes on finite asynchronous automata. [RAIRO ITA](#), 1987.
- [2] Zielonka. Asynchronous automata. [Book of Traces](#), 1995.
- [3] Pnueli & Rosner. Distributed reactive systems are hard to synthesize. [FOCS'90](#)
- [4] Madhusudan & Thiagarajan. Distributed controller synthesis for local specifications. [ICALP'01](#)
- [5] Madhusudan & Thiagarajan. Effective strategies for asynchronous distributed control. [CONCUR'02](#)
- [6] Mohalik & Walukiewicz. Distributed Games. [FSTTCS'03](#).
- [7] Bednarczyk. Hereditary history preserving bisimulations. TR, Gdansk, 1991.
- [8] Nielsen & Clausen. Games and logics for a noninterleaving bisimulation. *Nordic J. Comput.* 1995.
- [9] Jurdziński & Nielsen. Hereditary history preserving bisimilarity is undecidable. [STACS'00](#).
- [10] Mukund. Hereditary history preserving bisimulation is decidable for trace-labeled systems. [FSTTCS'02](#).
- [11] Ramadge & Wonham. The control of discrete event systems. [Proc IEEE](#), 1989.