Strategy Construction with Antichains

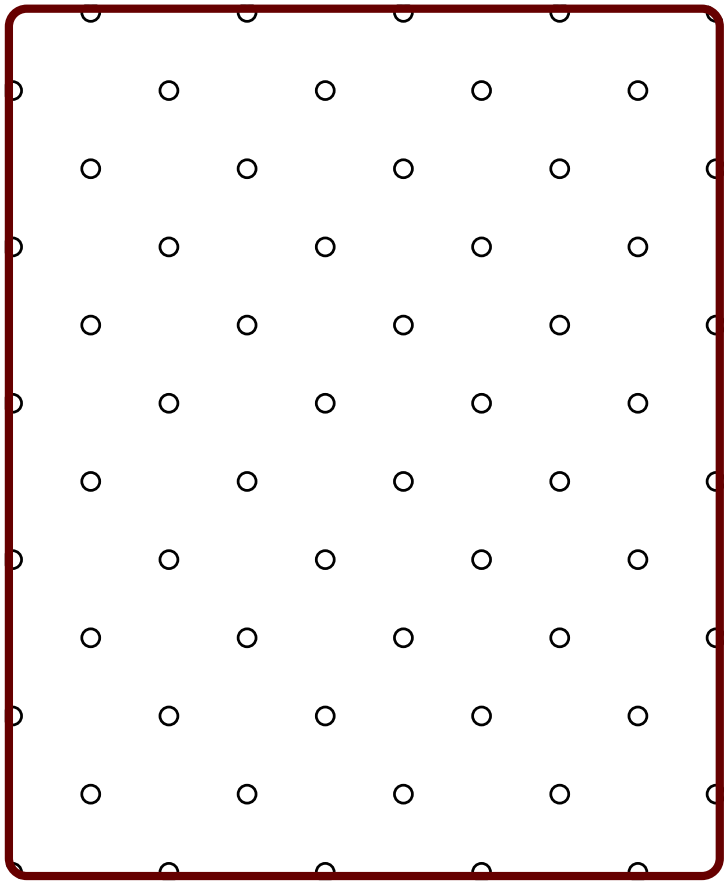
Dietmar Berwanger

LSV, CNRS & ENS Cachan

with Krishnendu Chatterjee, Laurent Doyen, Tom Henzinger

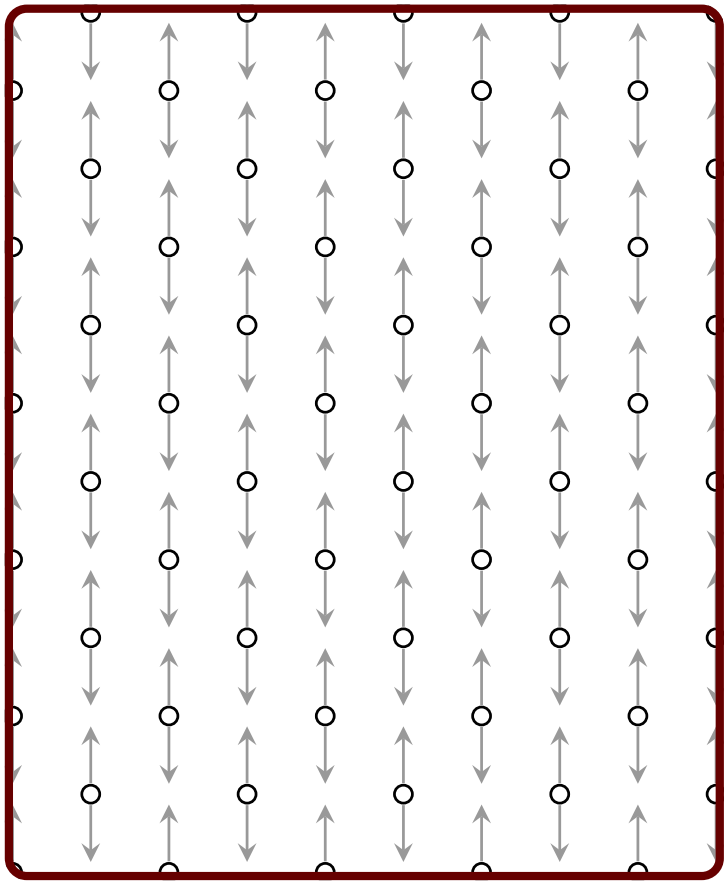
GASICS Aachen 2009

Transition structure with imperfect information



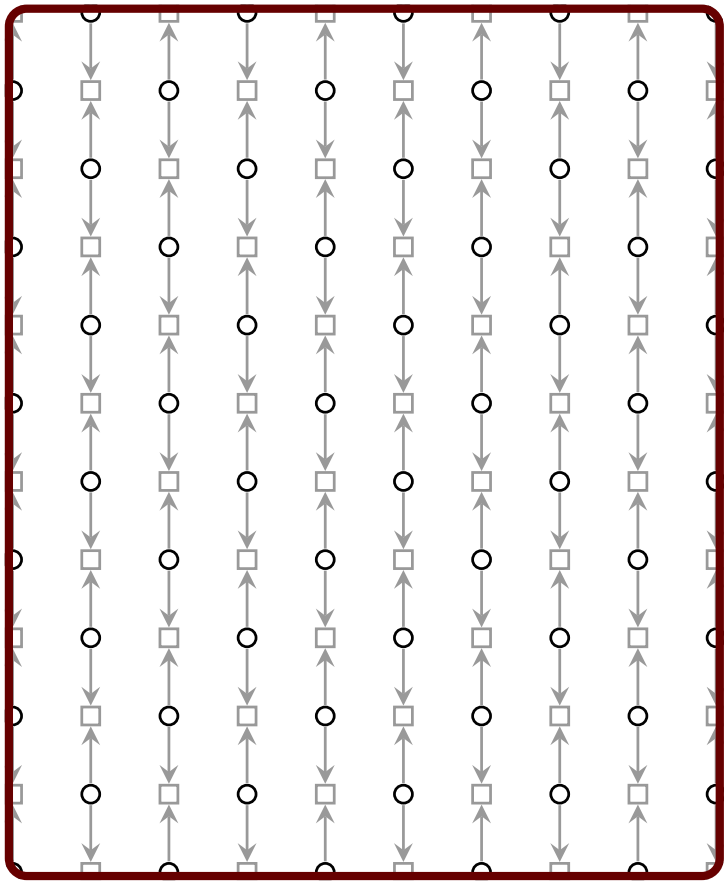
- States

Transition structure with imperfect information



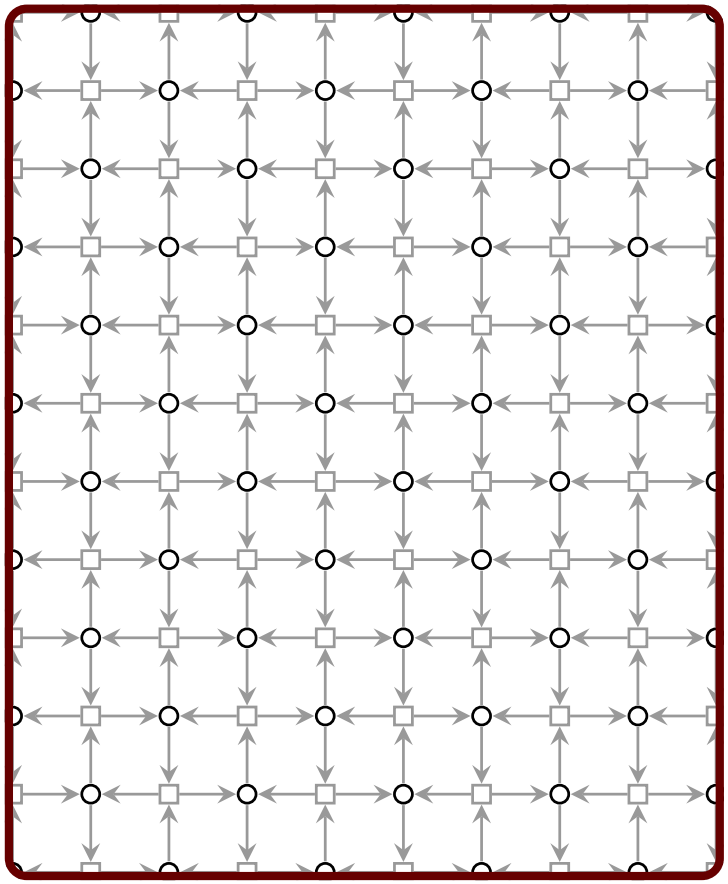
- States
- Player 1 – actions: $\uparrow, \downarrow$

Transition structure with imperfect information



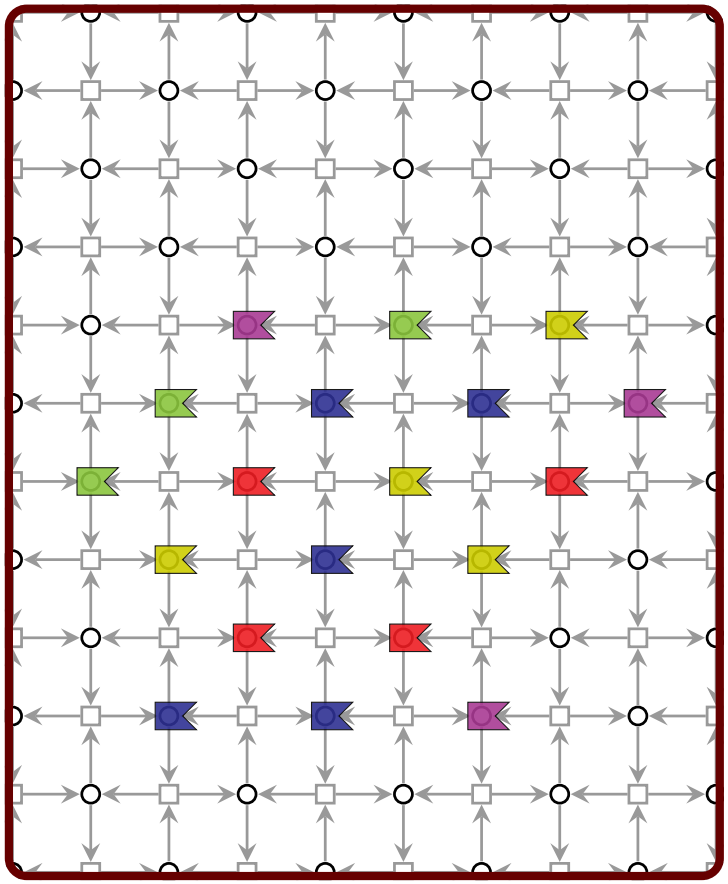
- States
- Player 1 – actions: $\uparrow, \downarrow$

Transition structure with imperfect information



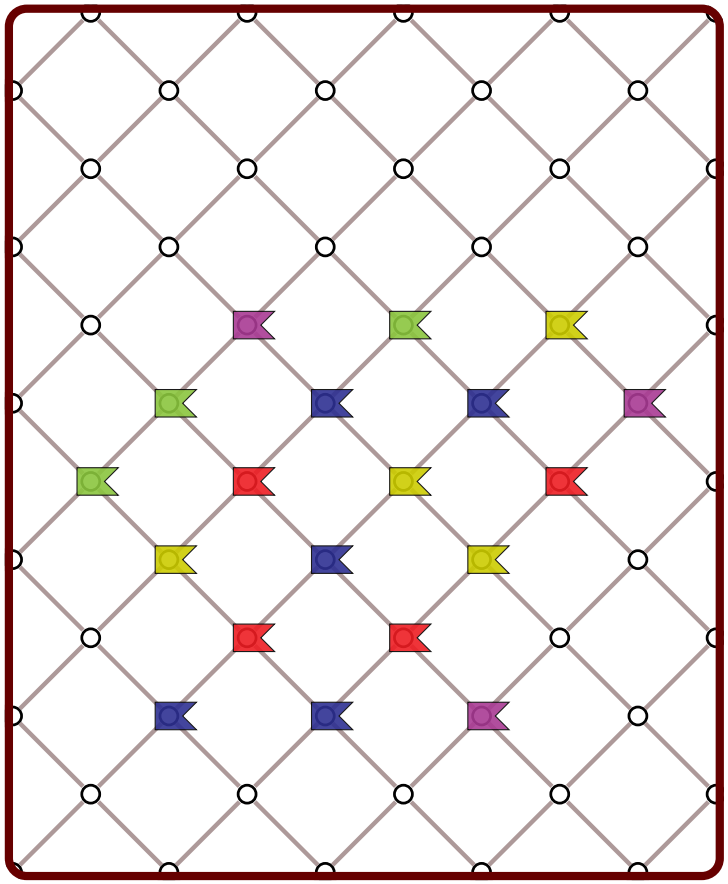
- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions

Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

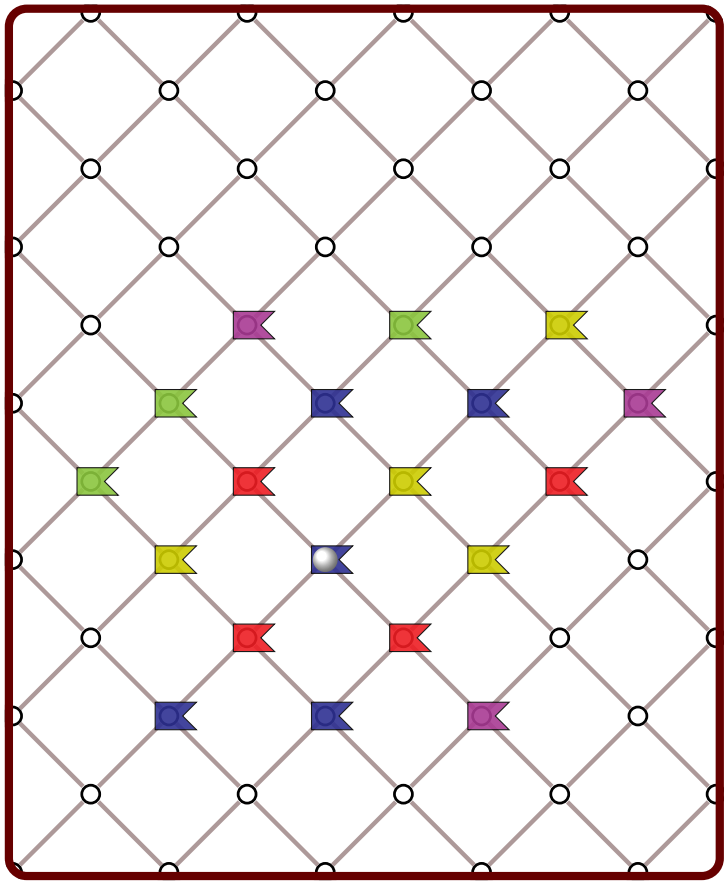
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play:

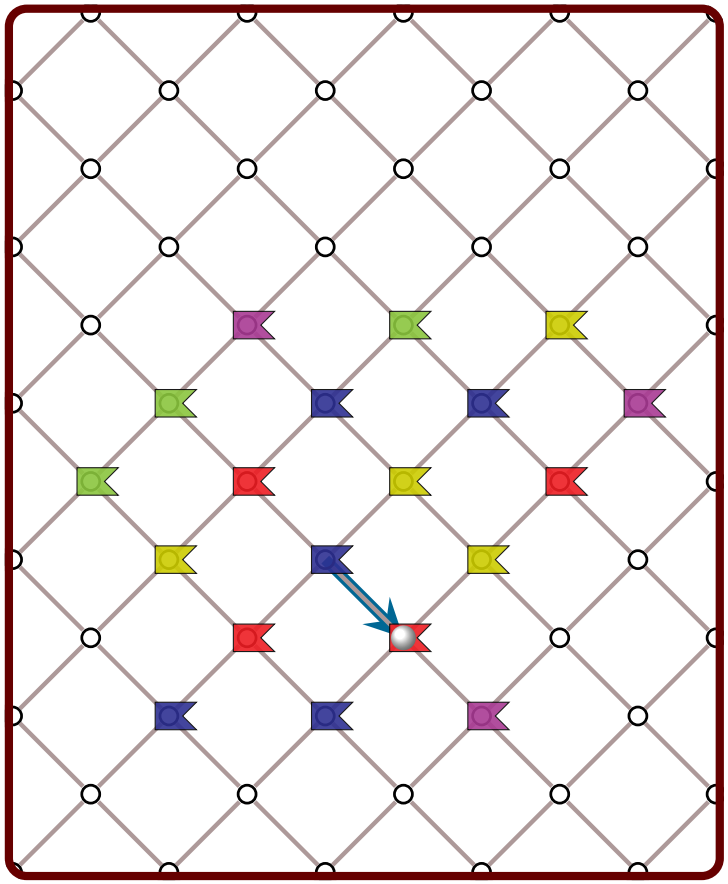
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: 

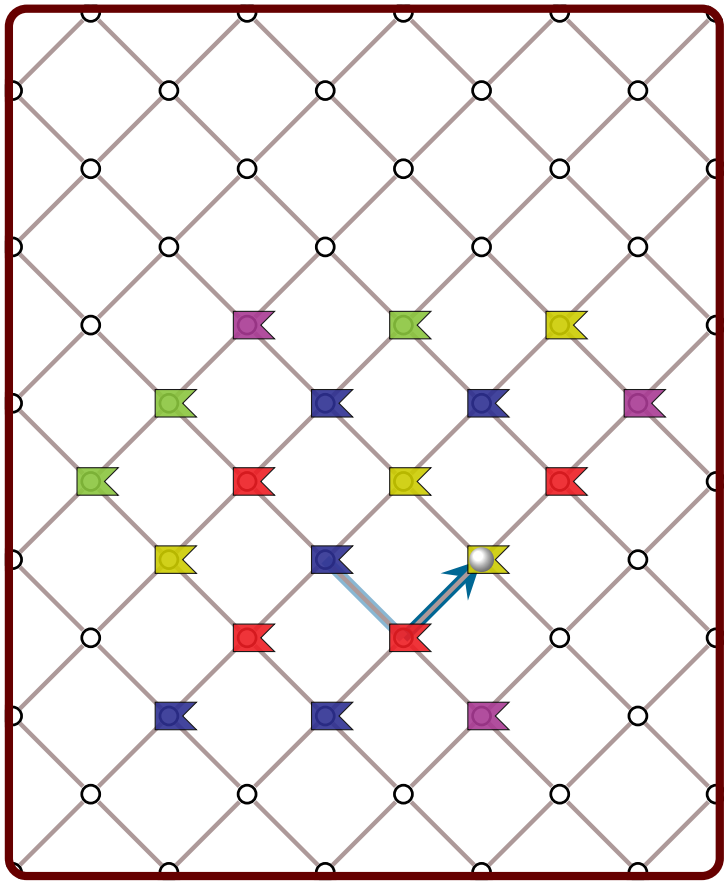
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: 

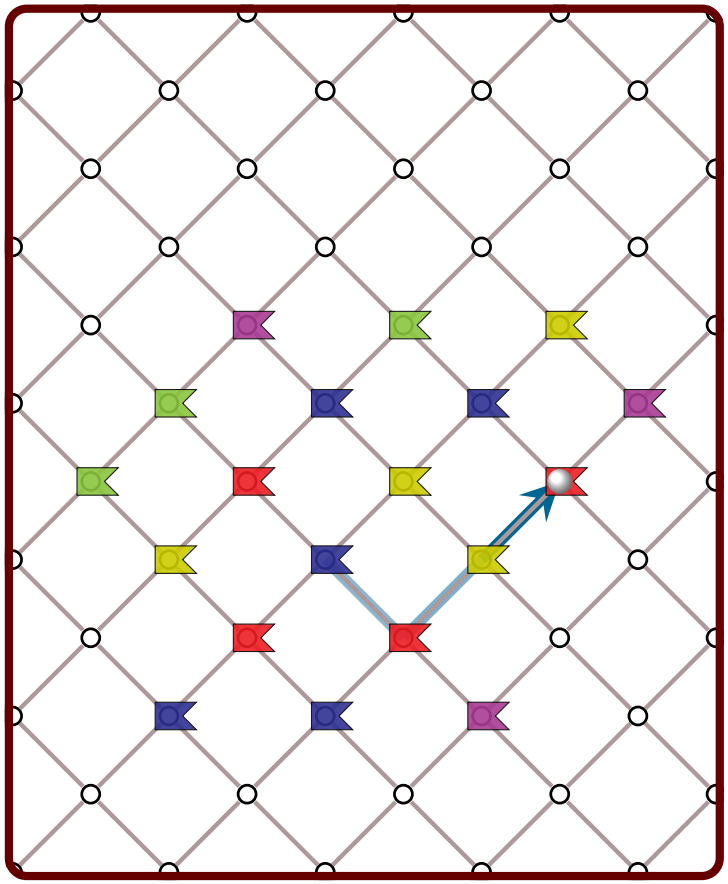
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations:

Play:

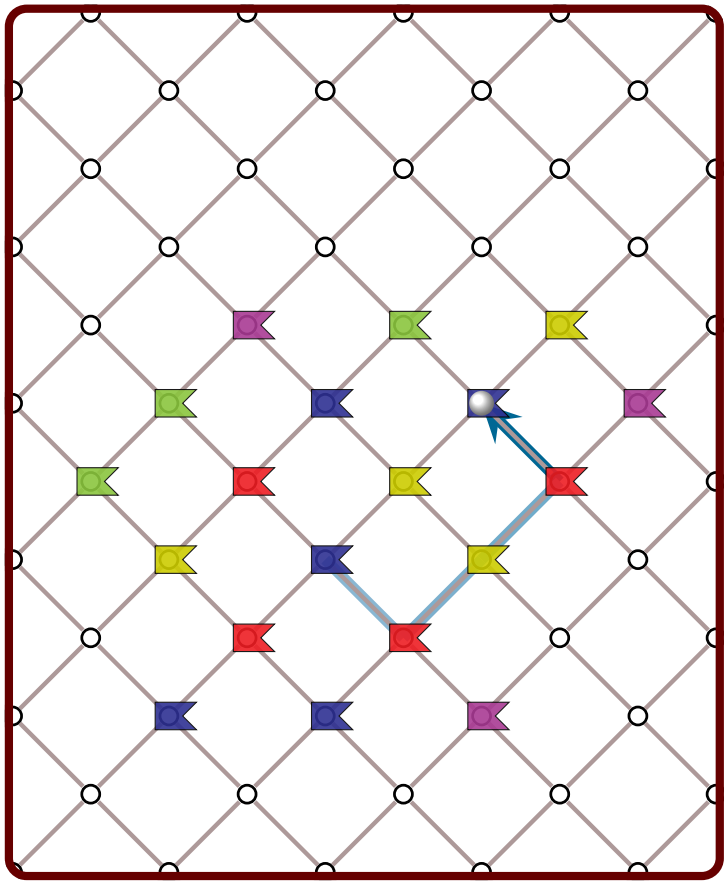
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: 

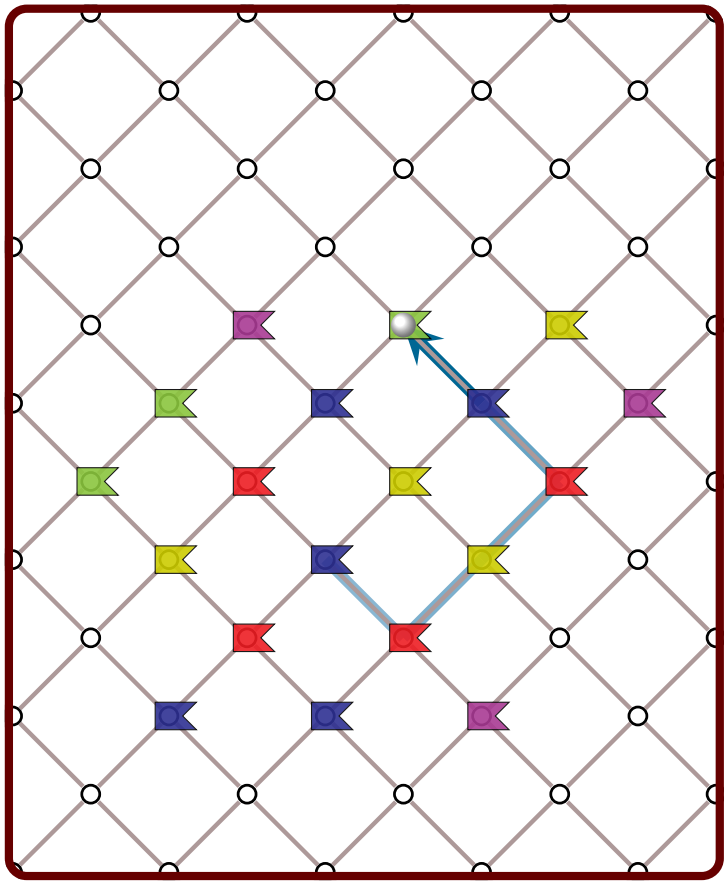
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: 

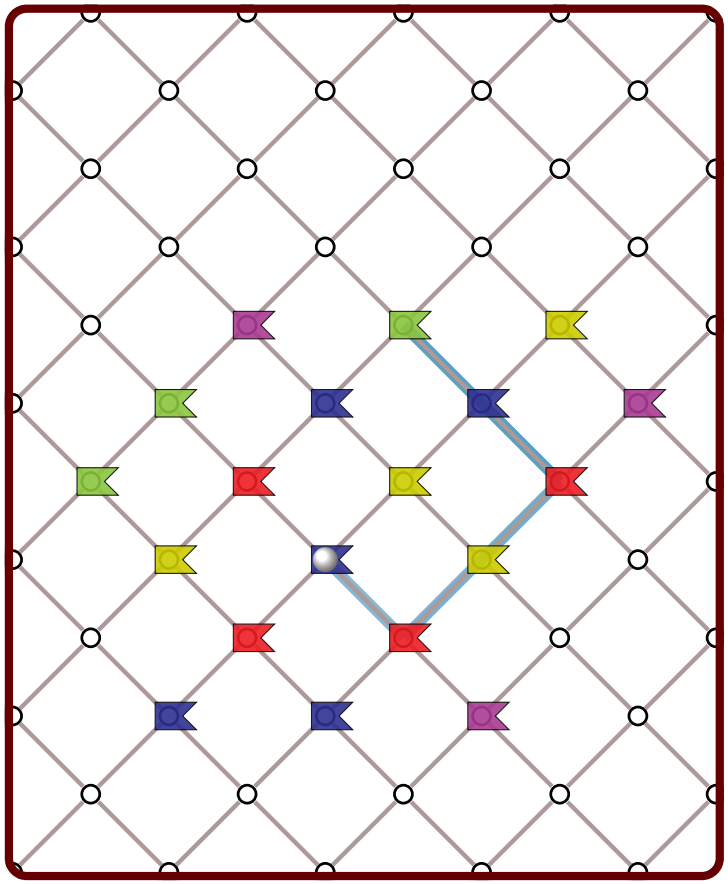
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: 

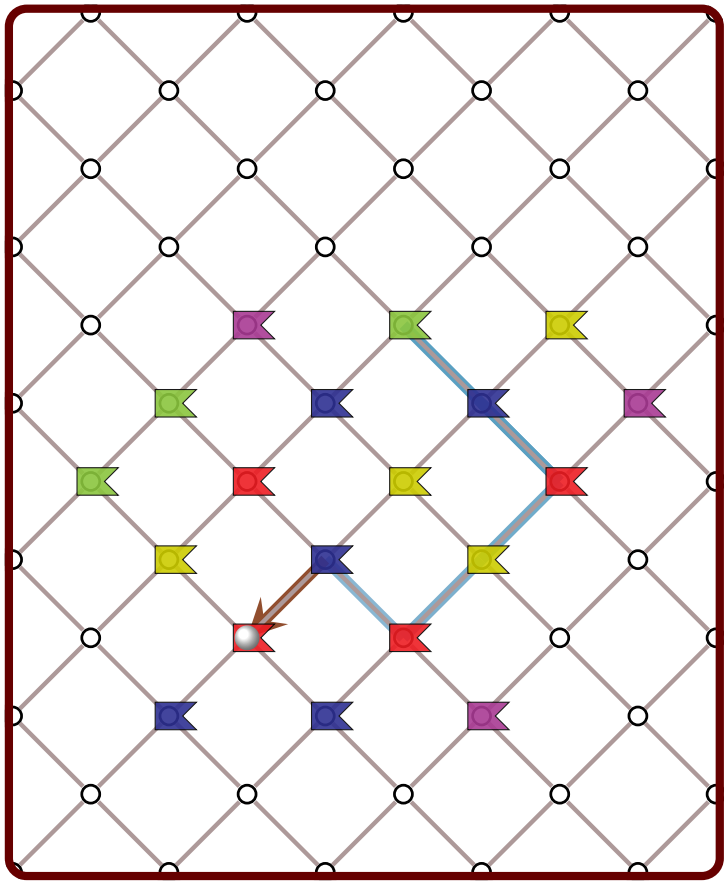
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: ...

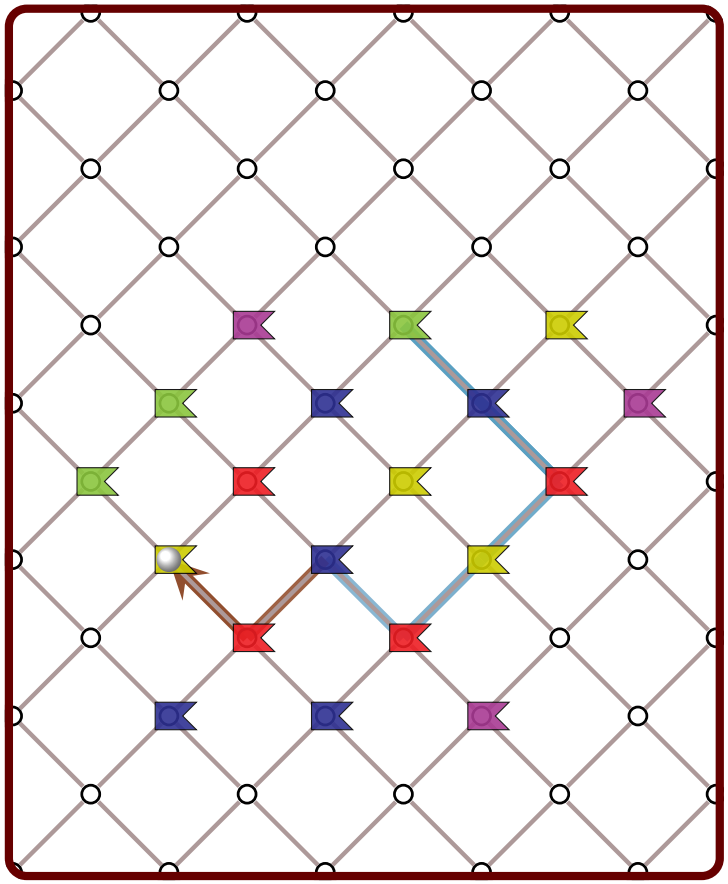
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: ...

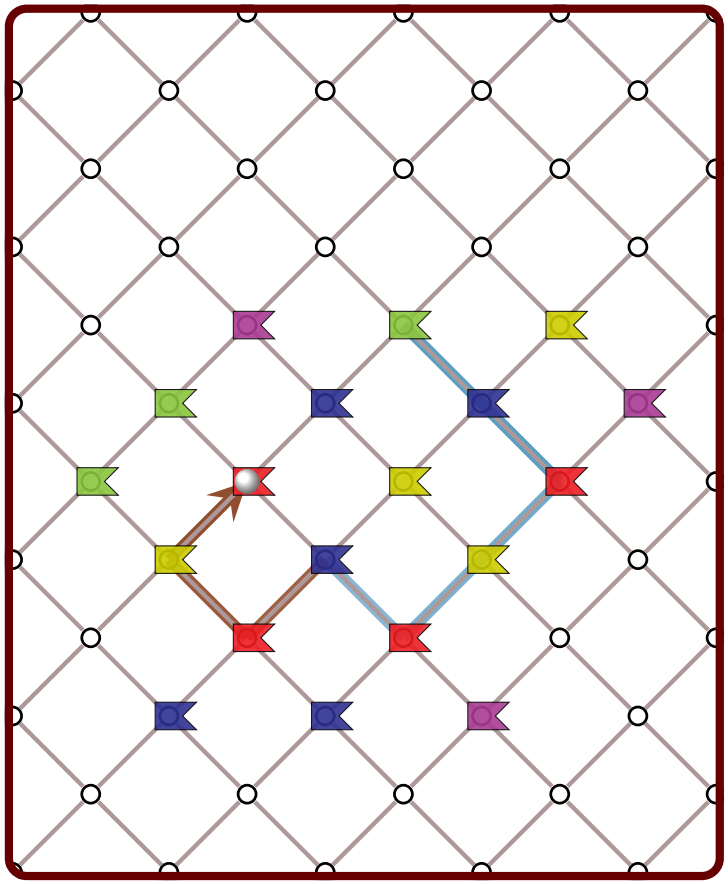
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: ...

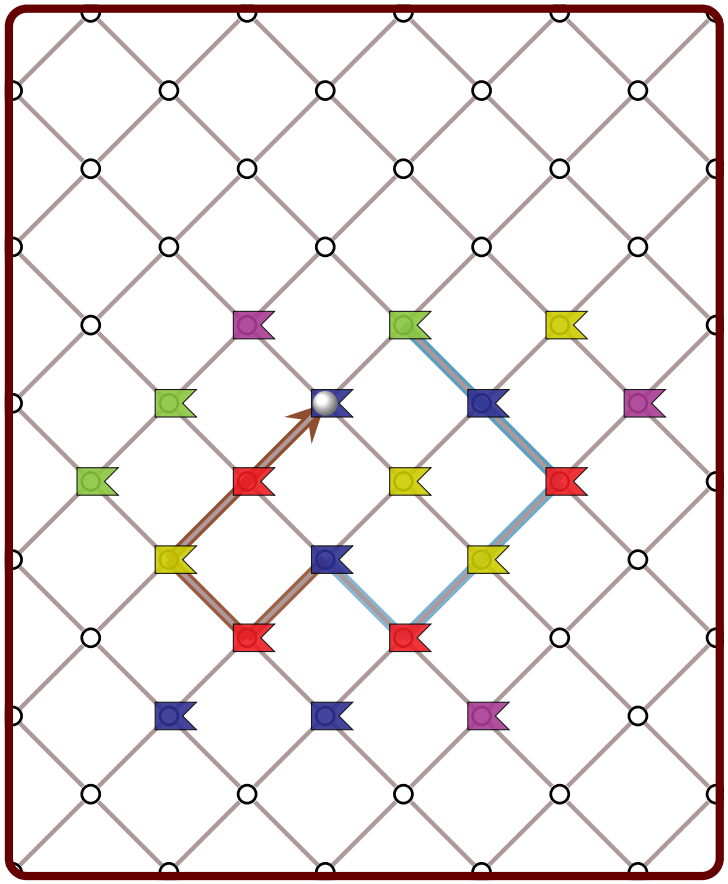
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: ...

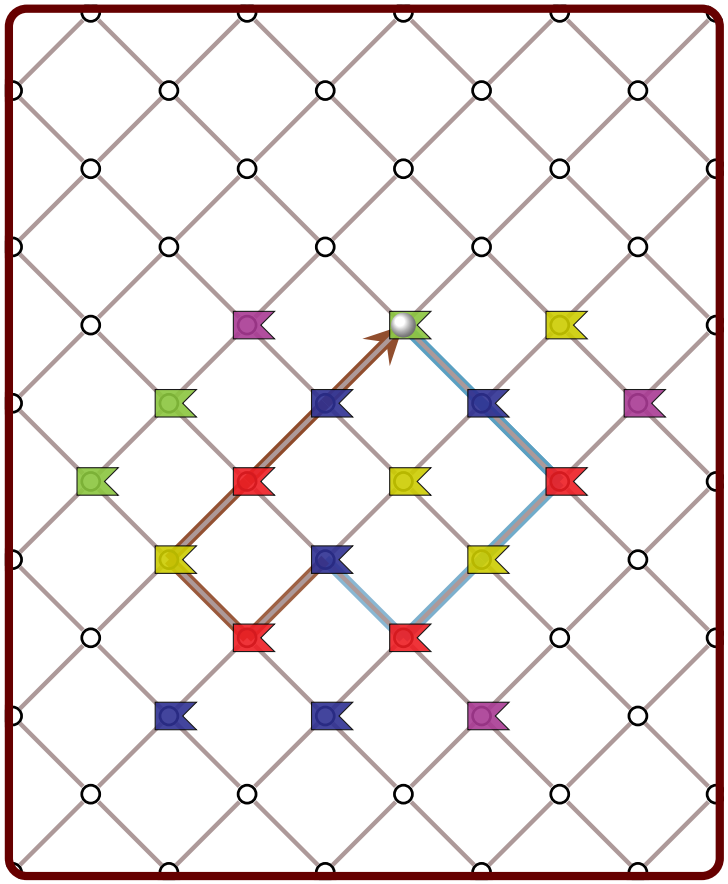
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: ...

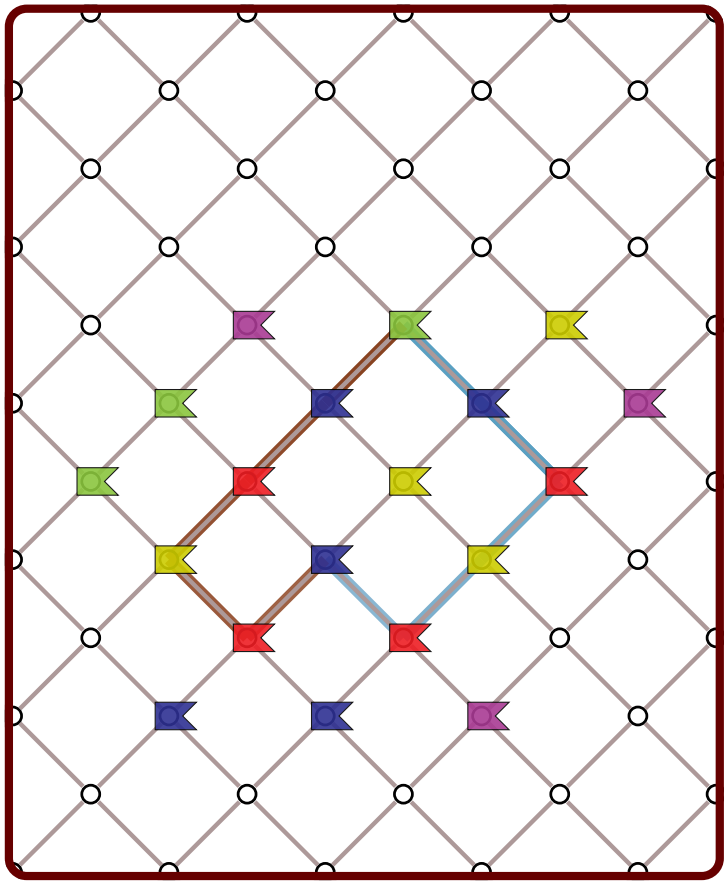
Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: ...

Transition structure with imperfect information



- States
- Player 1 – actions: $\uparrow, \downarrow$
- Transitions
- Player 2 – observations: 

Play: ...

Formal representation ▶

Game structure: $\mathcal{G} = (V, v_0, \Sigma, \Delta, \Gamma)$

V set of states/locations - finite

v_0 initial location

Σ action alphabet

$\Delta \subseteq V \times \Sigma \times V$ labelled transitions

Γ observation partition $\dot{\cup}\Gamma_o = V$;

Notation: $[v]$ – observational equivalence class of state $v \in V$

Formal representation ▶ ▶

strategy for Player 1 $\quad \alpha : \Gamma^+ \to \Sigma$

play $v_0, v_1, \ldots$ with $v_i \to^a v_{i+1} \in \Delta$, for some $a \in \Sigma$

follows strategy if $v_i \to^{\alpha([v_0],\ldots,[v_i])} v_{i+1} \in \Delta$, for all i.

outcome(α) set of all plays that follow α

Game: structure + objective

objective $\Phi \subseteq \Gamma^\omega$

winning strategy: $\quad [\text{outcome}(\alpha)] \subseteq \Phi$

Model particularities

- Synchroneous dynamics
  - asynchroneous: include actions that generate no observation/ticks
- Visible objectives
  - winning objective specified as sequences of observations
  - alternatively: specify sequences of states
- State-centered representation
  - alternatively: observation of actions, rather than of states
- Sure winning, asymmetric uncertainty

Solving a game – Basic technique

Game structure $\mathcal{G} = (V, v_0, \Sigma, \Delta, \Gamma)$

Winning condition: $\subseteq (\text{Observations})^\omega$

– reachability, safety, parity, ω-regular

Strategy: $(\text{Observations})^* \longrightarrow \text{Actions}$.

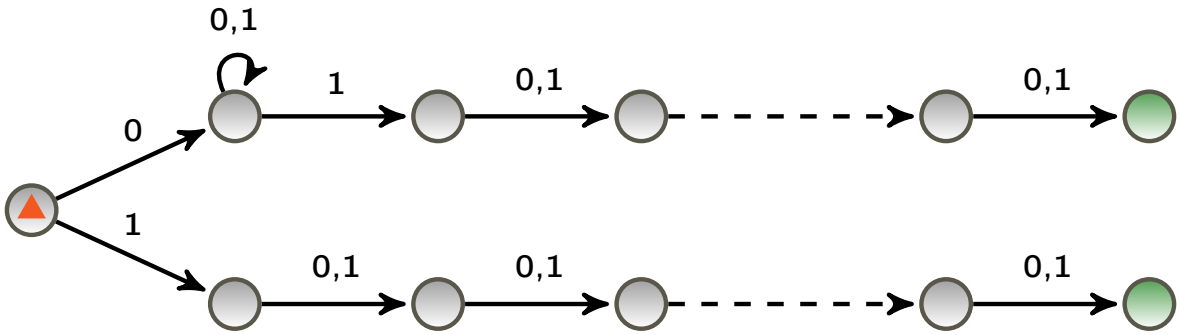
Questions:

- decide whether Player 1 has a winning strategy
- construct a winning strategy for Player 1

Parity games are generic for ω-regular objectives.

Key notion – knowledge set

What does Player 1 know about the current state of the system?



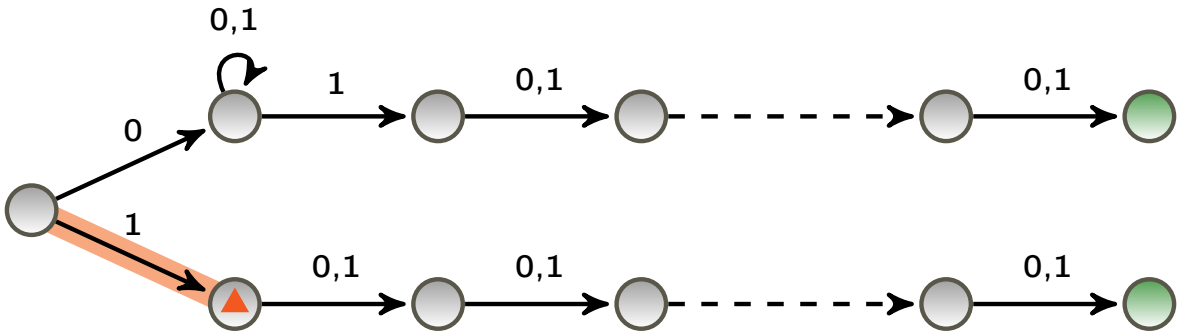
Set S of possible states (knowledge) – initially $S = \{v_0\}$;

If current state is known to be in S, and Player 1 performs action $a \in \Sigma$ receiving observation $o \in \Gamma$, then he knows that the new state will be in

$$S' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in S\} \cap \Gamma_o.$$

Key notion – knowledge set

What does Player 1 know about the current state of the system?



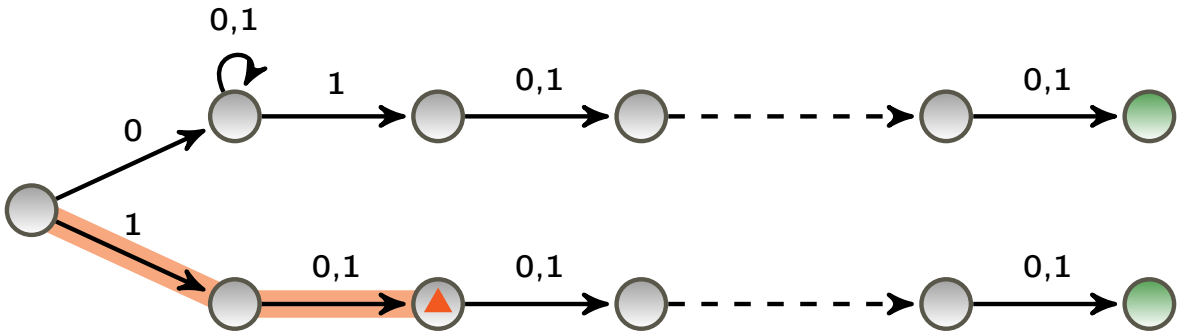
Set S of possible states (knowledge) – initially $S = \{v_0\}$;

If current state is known to be in S, and Player 1 performs action $a \in \Sigma$ receiving observation $o \in \Gamma$, then he knows that the new state will be in

$$S' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in S\} \cap \Gamma_o.$$

Key notion – knowledge set

What does Player 1 know about the current state of the system?



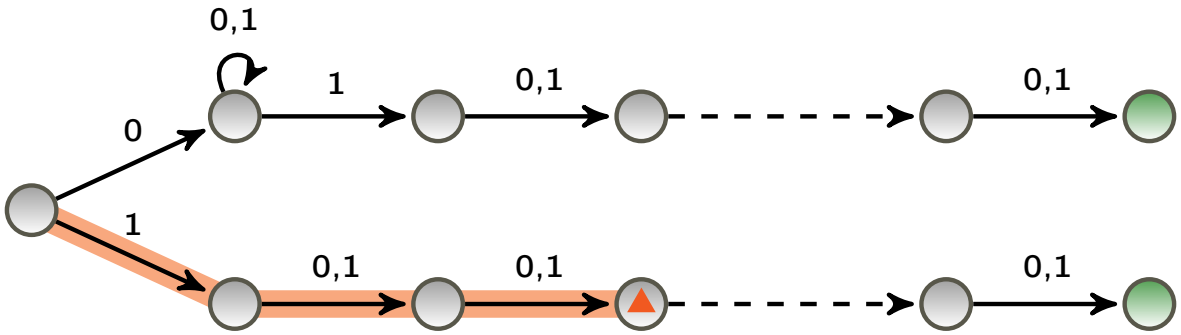
Set S of possible states (knowledge) – initially $S = \{v_0\}$;

If current state is known to be in S, and Player 1 performs action $a \in \Sigma$ receiving observation $o \in \Gamma$, then he knows that the new state will be in

$$S' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in S\} \cap \Gamma_o.$$

Key notion – knowledge set

What does Player 1 know about the current state of the system?



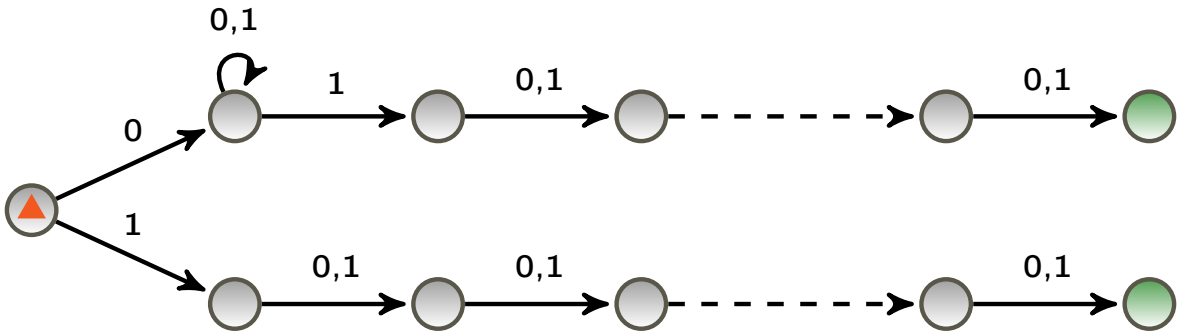
Set S of possible states (knowledge) – initially $S = \{v_0\}$;

If current state is known to be in S, and Player 1 performs action $a \in \Sigma$ receiving observation $o \in \Gamma$, then he knows that the new state will be in

$$S' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in S\} \cap \Gamma_o.$$

Key notion – knowledge set

What does Player 1 know about the current state of the system?



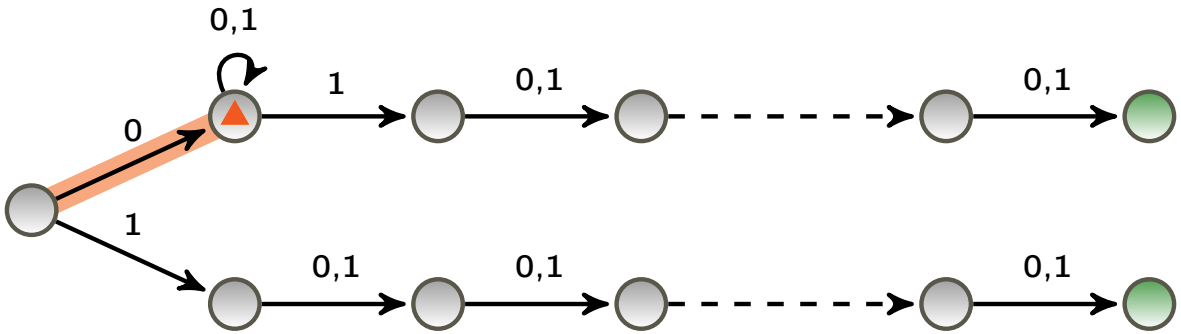
Set S of possible states (knowledge) – initially $S = \{v_0\}$;

If current state is known to be in S, and Player 1 performs action $a \in \Sigma$ receiving observation $o \in \Gamma$, then he knows that the new state will be in

$$S' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in S\} \cap \Gamma_o.$$

Key notion – knowledge set

What does Player 1 know about the current state of the system?



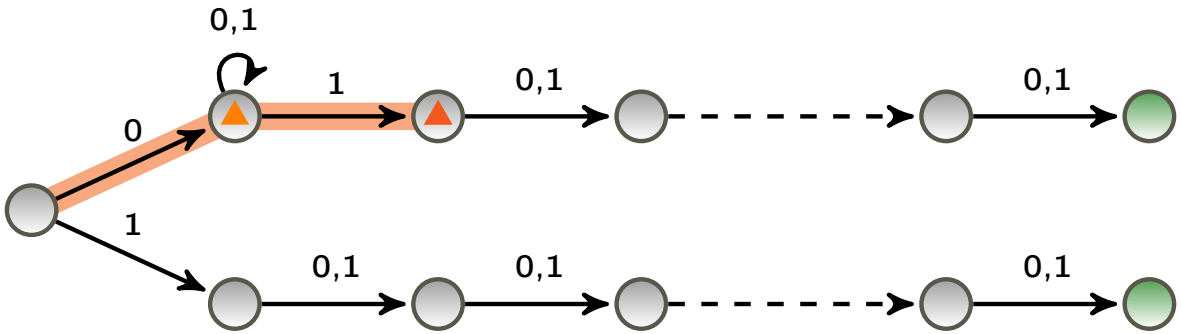
Set S of possible states (knowledge) – initially $S = \{v_0\}$;

If current state is known to be in S, and Player 1 performs action $a \in \Sigma$ receiving observation $o \in \Gamma$, then he knows that the new state will be in

$$S' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in S\} \cap \Gamma_o.$$

Key notion – knowledge set

What does Player 1 know about the current state of the system?



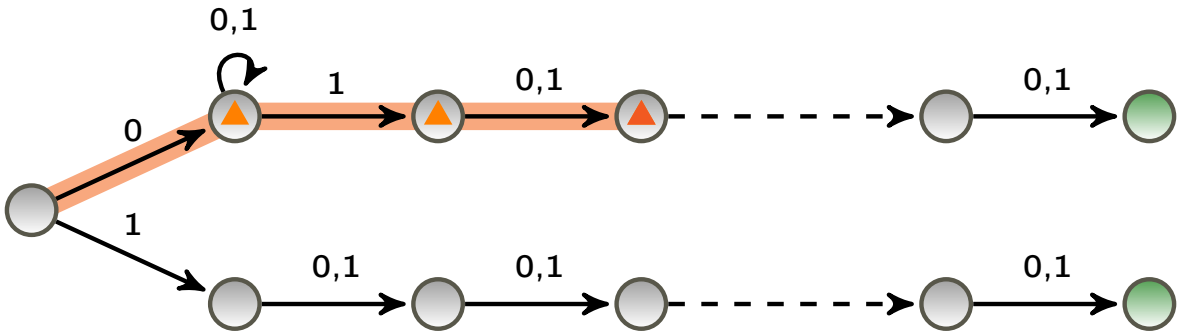
Set S of possible states (knowledge) – initially $S = \{v_0\}$;

If current state is known to be in S, and Player 1 performs action $a \in \Sigma$ receiving observation $o \in \Gamma$, then he knows that the new state will be in

$$S' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in S\} \cap \Gamma_o.$$

Key notion – knowledge set

What does Player 1 know about the current state of the system?



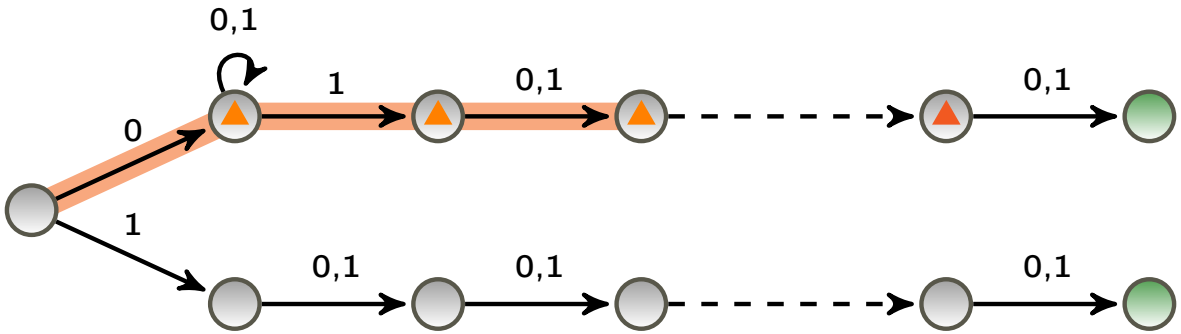
Set S of possible states (knowledge) – initially $S = \{v_0\}$;

If current state is known to be in S, and Player 1 performs action $a \in \Sigma$ receiving observation $o \in \Gamma$, then he knows that the new state will be in

$$S' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in S\} \cap \Gamma_o.$$

Key notion – knowledge set

What does Player 1 know about the current state of the system?



Set S of possible states (knowledge) – initially $S = \{v_0\}$;

If current state is known to be in S, and Player 1 performs action $a \in \Sigma$ receiving observation $o \in \Gamma$, then he knows that the new state will be in

$$S' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in S\} \cap \Gamma_o.$$

Reduction via powerset construction - Reif[84]

Transform game $\mathcal{G}$ with imperfect information

into a game $\mathcal{G}^K$ with perfect information

while preserving the power of Player 1 to win.

Powerset construction

- keeps track of what Player 1 can distinguish from memory
- treat set of indistinguishable states as an individual state

Powerset construction ▶

$$\mathcal{G} = (V, v_0, \Delta, \gamma') \quad \rightsquigarrow \quad \mathcal{G}^K = (S, s_0, \Delta^K, \Gamma^K)$$

states $S := \mathcal{P}(V) \setminus \{\emptyset\}, \quad s_0 := \{v_0\}$

transitions $s \rightarrow^a s' \in \Delta^K$ if

$$s' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in s\}$$

observations $s \in \Gamma_o^K$ if $v \in \Gamma_o$, for all $v \in s$.

$\mathcal{G}^{\mathcal{K}}$ is a game structure of perfect information.

Powerset construction ▶

$$\mathcal{G} = (V, v_0, \Delta, \gamma') \quad \rightsquigarrow \quad \mathcal{G}^K = (S, s_0, \Delta^K, \Gamma^K)$$

states $S := \mathcal{P}(V) \setminus \{\emptyset\}, \quad s_0 := \{v_0\}$

transitions $s \rightarrow^a s' \in \Delta^K$ if

$$s' = \{v' : v \rightarrow^a v' \in \Delta, \text{ for some } v \in s\} \cap \Gamma_o$$

for some observation o.

observations $s \in \Gamma_o^K$ if $v \in \Gamma_o$, for all $v \in s$.

$\mathcal{G}^{\mathcal{K}}$ is a game structure of perfect information.

Powerset construction ▶ ▶

Strategies translate back and forth.

Proposition

Every winning strategy for Player 1 in $(\mathcal{G}^{\mathcal{K}}, \Phi)$
is also a winning strategy in $\mathcal{G}$, and vice versa.

Corollary for parity games:

Memoryless determinacy / perfect-information

implies finite-memory determinacy / imperfect information

(memory automaton tracks set of possible positions)

State explosion

- Problem is Exptime-hard
- Exponential memory might be needed.

- However, the information-set construction
  - uses exponential space
  - has no on-the-fly solution
  - is independent of objective

Can we do any better?

Logical characterisation of winning regions

Set of states from which Player 1 can force the play into a set Z

$$\mathsf{CPre}(Z) = \{s \in S : \exists a \in \Sigma\ (\forall t\ .\ s \to^a t \in \Delta)\ t \in Z\}$$

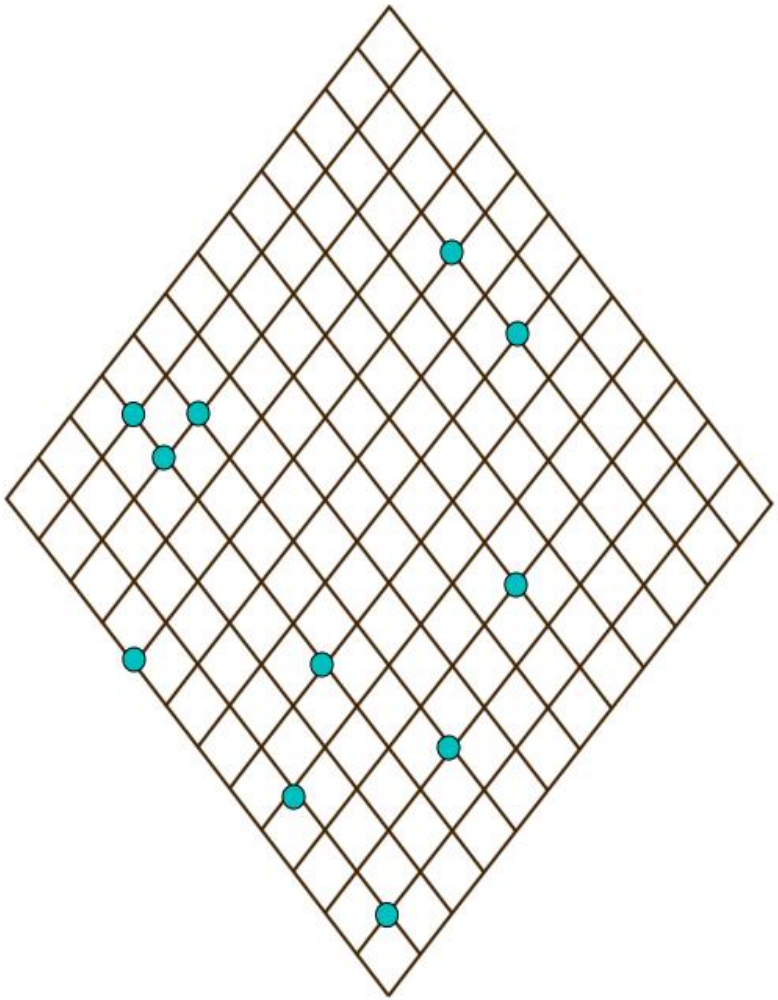
Example: Region in which Player 1 can maintain property P:

$$\mathsf{Safe} := \nu X.\ P \wedge \mathsf{CPre}(X)$$

Winning region in a parity game with priorities $0, \ldots, k$:

$$\mathsf{Win} := \nu X_0.\ \mu X_1.\ \ldots \nu X_k.\ \bigwedge_{i \leq k} (\ P_i \to \mathsf{CPre}(X_i)\ )$$

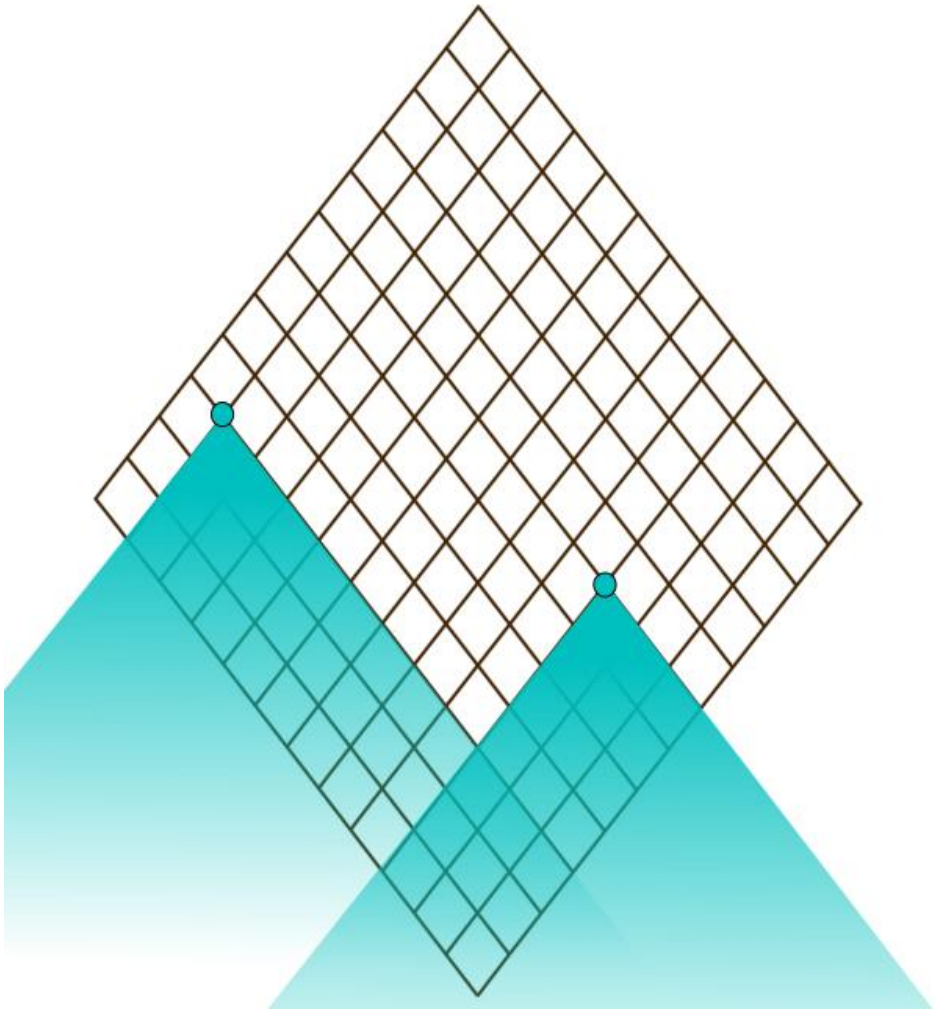
Antichains



- Interesting sets

  have a particular structure

Antichains

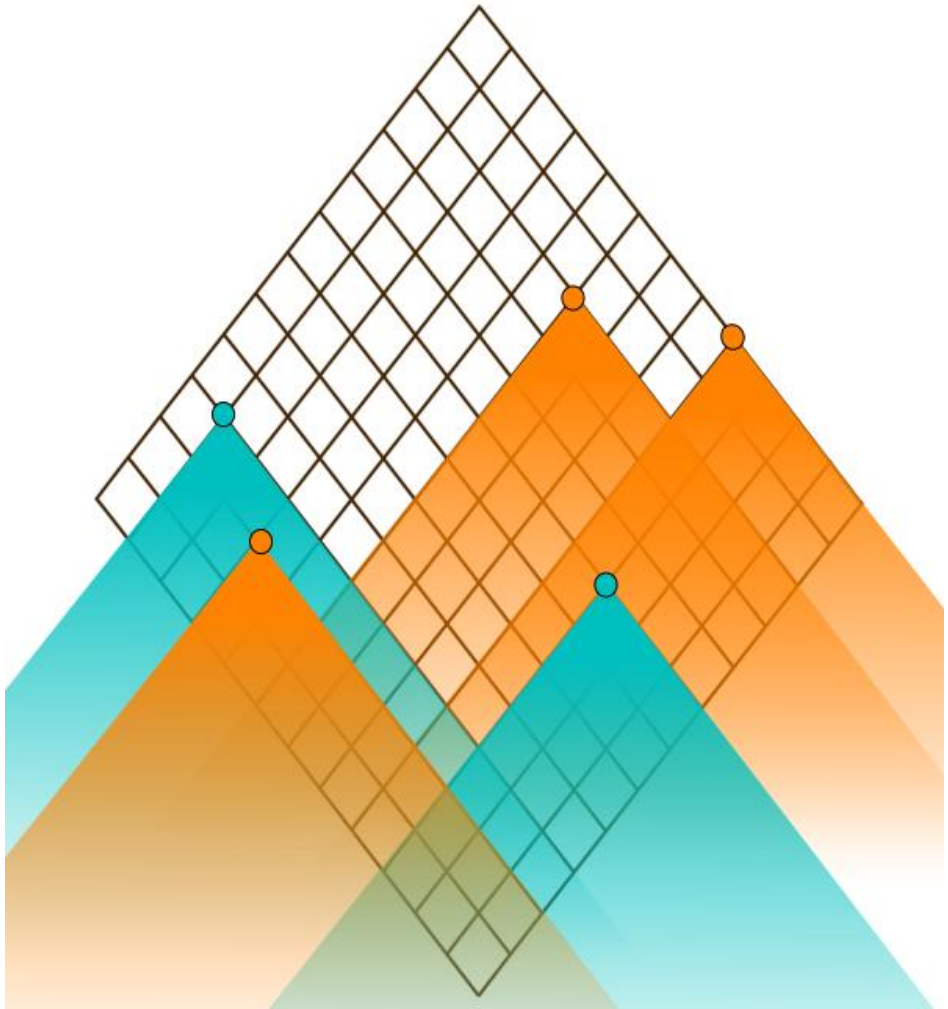


- Interesting sets

  have a particular structure

  - downwards-closed

Antichains



- Interesting sets

  have a particular structure

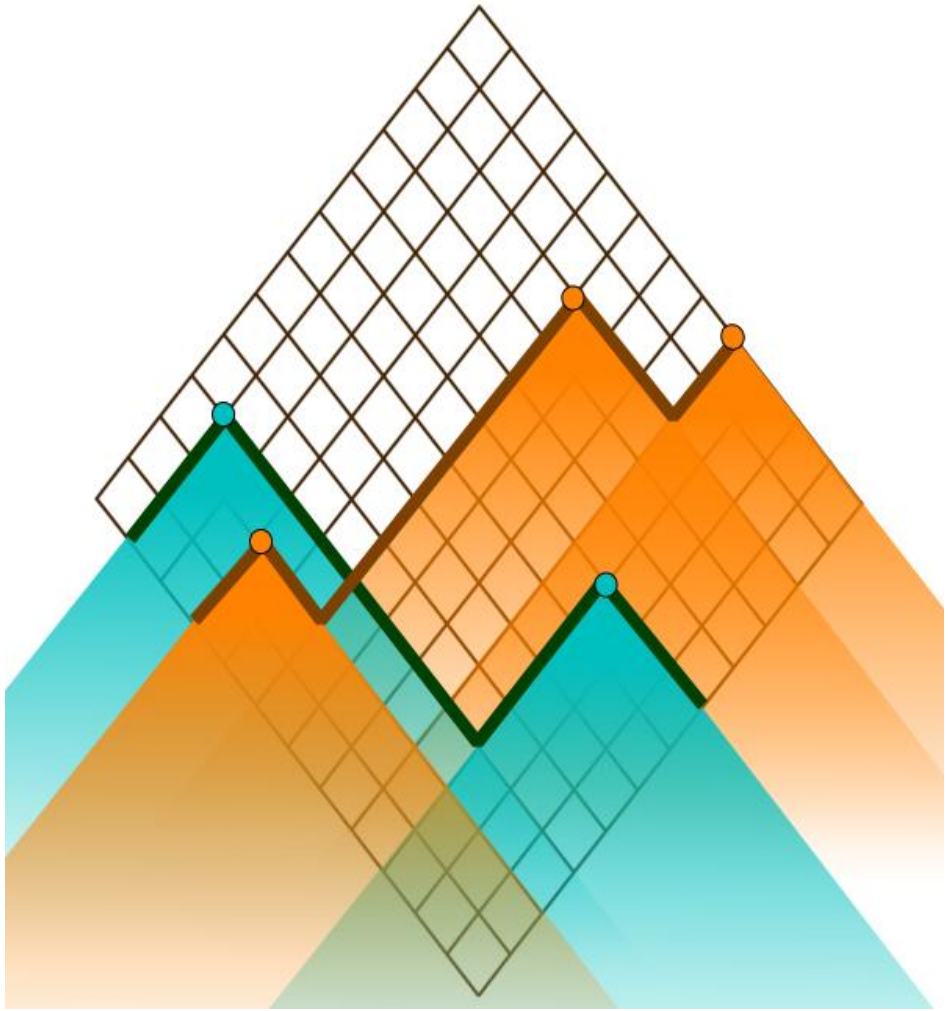
  - downwards-closed

- Interesting operations

  preserve it:

  - CPre, $\cup$, $\cap$, iteration

Antichains



- Interesting sets

  have a particular structure

  - downwards-closed

- Interesting operations

  preserve it:

  - CPre, $\cup$, $\cap$, iteration

Decision algorithm with antichains

[CDHR07]

- Evaluates characterisation of winning positions

  as a μ-calculus formula

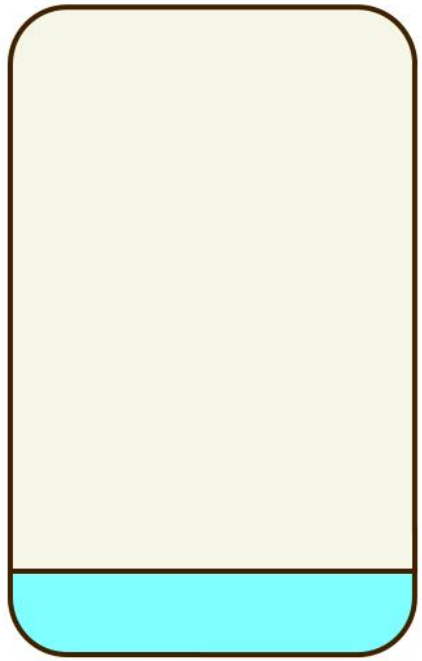
  over the lattice of antichains

- Performs well in experiments

- Does not construct strategies.

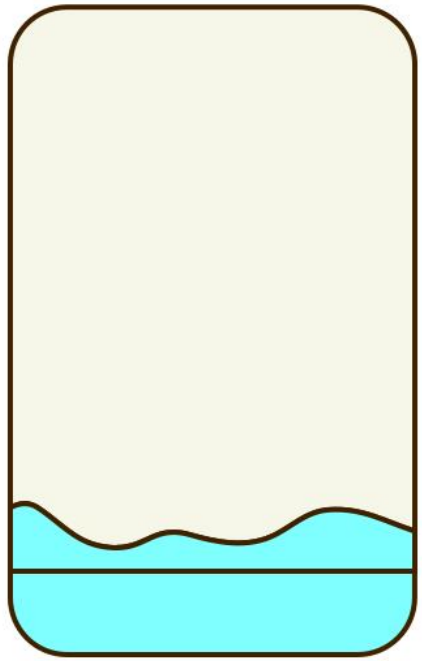
Strategy construction / perfect information

Start iteration: best priorities, clear win



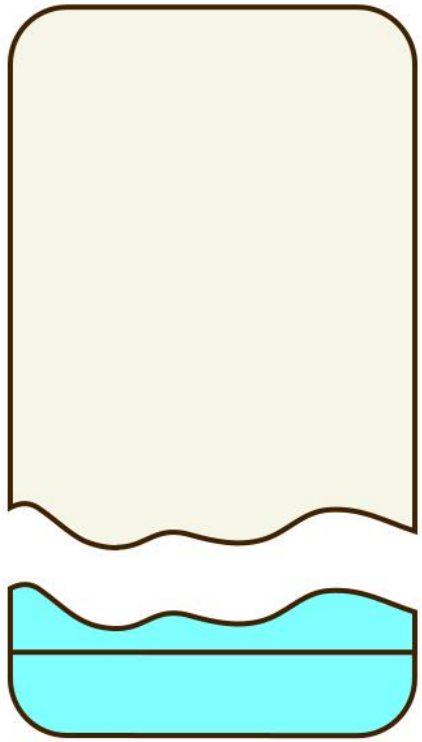
Strategy construction / perfect information

Compute attractor



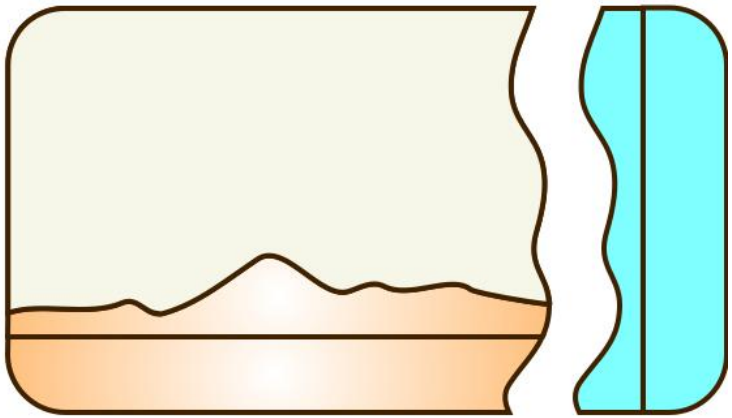
Strategy construction / perfect information

Decomposition: solved positions aside



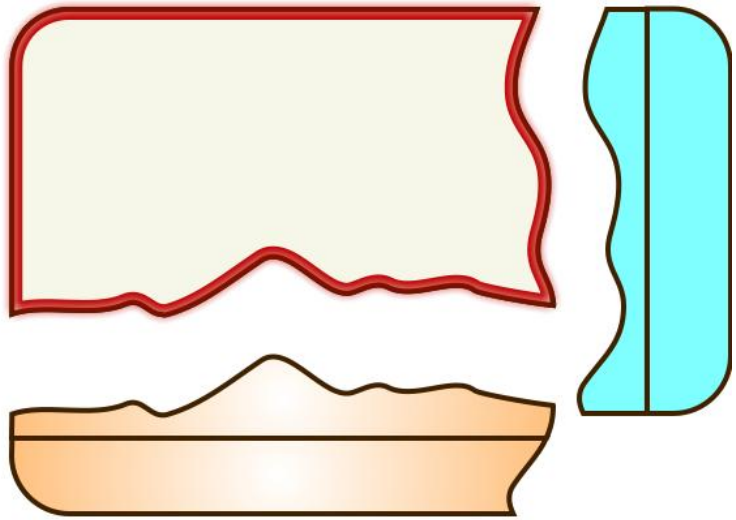
Strategy construction / perfect information

To avoid: worst priorites + co-attractor



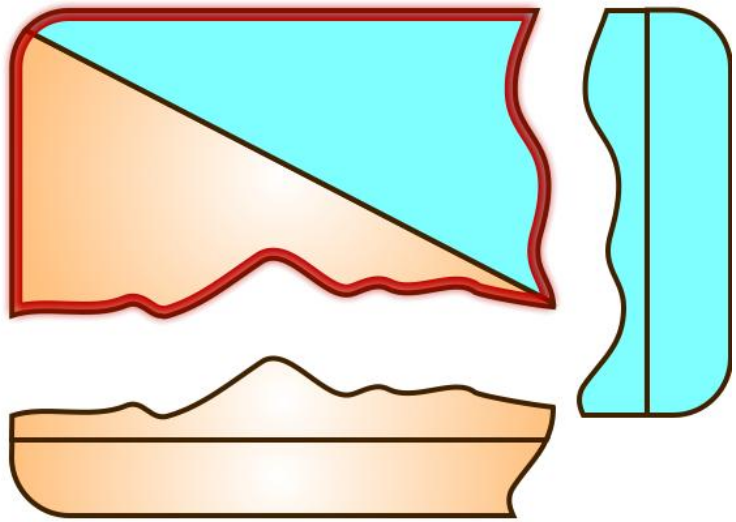
Strategy construction / perfect information

Decomposition: currently unsolvable positions aside



Strategy construction / perfect information

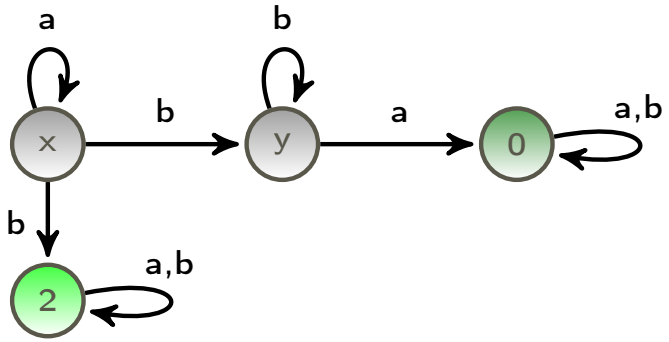
Decomposition: recursion



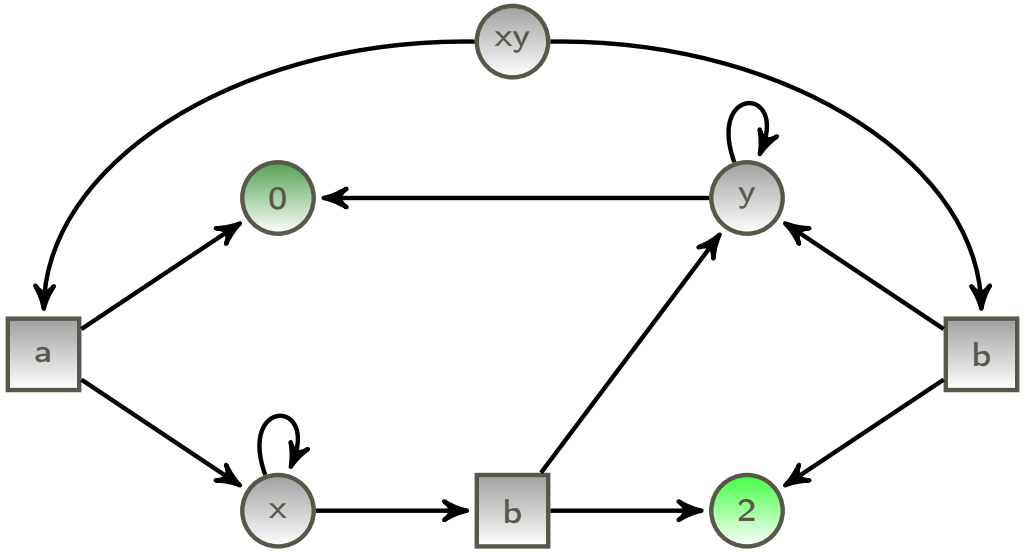
Complementation breaks antichains.

Broken chain

$\mathcal{G}$:



$\mathcal{G}^K$:



Algorithm - idea

Classical:

- Two operations that generate strategic data
  - attractor moves towards current winning region;
  - arbitrary moves, at best priorities.
- simple decomposition for recursion:
  - remove winning region and (attractor of) most significant priorities.

With antichains:

- Strategic data from one operation:
  - attractor moves towards best priorities
    that stay safe within current winning region
- recursion into subgame with fewer priorities and safety constraints.

New decomposition scheme

W: target set – current winning region (strategy fixed).

Merge the two orthogonal operations to reduce the problem:

- collection of best-priority attractor
- separation of worst-priority co-attractor

into one operation of:

- attraction to the solution of a simpler game with objective:

$$[\,\mathsf{parity}(p-2) \wedge \mathsf{safe}(p-2)\,] \vee \mathsf{reach}(W);$$

Decomposition now involves two fixed-point alternations,
but no complementation.

Composing a strategy

- Use antichain decision procedure as an oracle
  - work within set of safe positions
  - record attractor to current solution
- stack attractor moves as they arise
  - first-published move sticks
  - attraction stops when target set reached

Theorem: The construction follows the antichain decision algorithm, and has the same complexity.

Conclusion

- To deal with imperfect information

  we need to maintain information sets.

  - leads to state explosion.

- Antichains provide a succinct representation of information sets

  which occur when deciding the winner of parity games.

- The approach can be extended for constructing strategies.

- Test implementation Alpaga on antichains.be/alpaga/.