Games for Controller Synthesis

Laurent Doyen

EPFL

MoVeP’08
The Synthesis Question

Given a plant $P$ ...
Given a plant $P$ and a specification $\varphi$, ...

Maintain the temperature in the range $[T_{\text{min}}, T_{\text{max}}]$. 

![Diagram of a tank with a thermometer and a gas burner.](image)
Given a plant $P$ and a specification $\varphi$, is there a controller $C$ such that the closed-loop system $C \parallel P$ satisfies $\varphi$?

Maintain the temperature in the range $[T_{\text{min}}, T_{\text{max}}]$. 

Tank

Thermometer

Gas burner

Digital controller
Synthesis as a game

Given a plant $P$ and a specification $\varphi$, is there a controller $C$ such that the closed-loop system $C \parallel P$ satisfies $\varphi$?

Plant: 2-players game arena

Input (Player 1, System, Controller)

vs.

Output (Player 2, Environment, Plant)

Specification: game objective for Player 1

$\varphi \equiv \varphi_1(a_1, u_1) \land \varphi_2(a_2, u_2)$
Given a plant $P$ and a specification $\varphi$, is there a controller $C$ such that the closed-loop system $C \parallel P$ satisfies $\varphi$?

If a controller $C$ exists, then construct such a controller.
Synthesis as a game

Plant: 2-players game arena
Specification: game objective for Player 1
Controller: winning strategy for Player 1

We are often interested in simple controllers: finite-state, or even stateless (memoryless).

We are also often interested in “least restrictive” controllers.
Example

Objective: avoid Bad

Uncontrollable actions
Objective: avoid Bad

Uncontrollable actions
Objective: avoid Bad

Uncontrollable actions
Objective: avoid Bad

Winning strategy = Controller
Games for Synthesis

Several types of games:

• Turn-based vs. Concurrent
• Perfect-information vs. Partial information
• Sure vs. Almost-sure winning
• Objective: graph labelling vs. monitor
• Timed vs. untimed
• Stochastic vs. deterministic
• etc. …

This tutorial: Games played on graphs, 2 players, turn-based, $\omega$-regular objectives.
This tutorial: Games played on graphs, 2 players, turn-based, $\omega$-regular objectives.

Outline

Part #1: perfect-information

Part #2: partial-information
Two-player game structures
Rounded states belong to Player 1
Rounded states belong to Player 1

Square states belong to Player 2
Playing the game: the players move a **token** along the edges of the graph

- The token is initially in $v_0$.
- In rounded states, Player 1 chooses the next state.
- In square states, Player 2 chooses the next state.
Play: $v_0$
Play: $v_0$ $v_1$
Play: $v_0 \ v_1 \ v_3$
belongs to Player 1

belongs to Player 2

Play: $v_0 \ v_1 \ v_3 \ v_0$
Play: $v_0 \ v_1 \ v_3 \ v_0 \ v_2 \ ...$
Two-player game graphs

A 2-player game graph $G = \langle V_1, V_2, \hat{v}, \text{Succ} \rangle$ consists of:

- $V_1$ the set of Player 1 states,
- $V_2$ the set of Player 2 states,
- $V_1 \cap V_2 = \emptyset$ and $V := V_1 \cup V_2$;
- $\hat{v} \in V$ the initial state,
- $\text{Succ} : V \to 2^V \setminus \emptyset$ the transition relation.
A play in $G = \langle V_1, V_2, \hat{v}, \text{Succ}\rangle$ is an infinite sequence $w = v_0v_1v_2\cdots \in V^\omega$ such that:

1. $v_0 = \hat{v}$,
2. $v_{i+1} \in \text{Succ}(v_i)$ for all $i \geq 0$.

$V = V_1 \cup V_2$

$\text{Succ} : V \to 2^V \setminus \emptyset$
Who is winning?

Play: $v_0 \ v_1 \ v_3 \ v_0 \ v_2 \ \ldots$
Who is winning?

A winning condition for Player k is a set $W_k \subseteq V^\omega$ of plays.

Play: $v_0 \, v_1 \, v_3 \, v_0 \, v_2 \, \ldots$
Who is winning?

A winning condition for Player k is a set $W_k \subseteq V^\omega$ of plays.

A 2-player game is zero-sum if $W_2 = V^\omega \setminus W_1$. 
A winning condition for Player k is a set $W_k \subseteq V^\omega$ of plays.

Given $\mathcal{T} \subseteq V$, let

- $\text{Reach}(\mathcal{T}) = \{v_0 v_1 \cdots \in V^\omega \mid \exists i : v_i \in \mathcal{T}\}$  

  Touch $\mathcal{T}$ eventually
A **winning condition** for Player $k$ is a set $W_k \subseteq V^\omega$ of plays.

Given $\mathcal{T} \subseteq V$, let

- $\text{Reach}(\mathcal{T}) = \{v_0v_1\cdots \in V^\omega \mid \exists i : v_i \in \mathcal{T}\}$  
  Touch $\mathcal{T}$ eventually

- $\text{Safe}(\mathcal{T}) = \{v_0v_1\cdots \in V^\omega \mid \forall i : v_i \in \mathcal{T}\}$  
  Avoid $V \setminus \mathcal{T}$ forever

**Diagram:**

- Reachability
- Safety
Winning condition

A winning condition for Player $k$ is a set $W_k \subseteq V^\omega$ of plays.

Given $T \subseteq V$, let

- $\text{Reach}(T) = \{v_0v_1 \cdots \in V^\omega | \exists i : v_i \in T\}$
  - Touch $T$ eventually
- $\text{Safe}(T) = \{v_0v_1 \cdots \in V^\omega | \forall i : v_i \in T\}$
  - Avoid $V \setminus T$ forever
- $\text{Büchi}(T) = \{v_0v_1 \cdots \in V^\omega | \forall j \cdot \exists i \geq j : v_i \in T\}$
  - Visit $T$ $\infty$-often
- $\text{coBüchi}(T) = \{v_0v_1 \cdots \in V^\omega | \exists j \cdot \forall i \geq j : v_i \in T\}$
  - Visit $V \setminus T$ finitely often
Remark

A **winning condition** for Player $k$ is a set $W_k \subseteq V^\omega$ of plays.

Reach($\mathcal{T}$), Safe($\mathcal{T}$), Büchi($\mathcal{T}$) and coBüchi($\mathcal{T}$) are subsumed by the **parity** condition:

- Given a priority function $p : V \to \mathbb{N}$, define
  
  $\text{Parity}(p) = \{v_0v_1 \cdots | \min\{d \mid \forall i \cdot \exists j \geq i : p(v_i) = d\} \text{ is even}\}$

  “Minimal priority seen $\infty$-often is even”

---

\[ p=4 \rightarrow p=1 \rightarrow p=3 \rightarrow p=1 \rightarrow p=0 \]
\[ \in \text{Parity}(p) \]

\[ p=2 \rightarrow p=0 \rightarrow p=3 \rightarrow p=1 \rightarrow p=2 \]
\[ \not\in \text{Parity}(p) \]
Players use strategies to play the game, *i.e.* to choose the successor of the current state.

A **strategy for Player** $k$ is a function:

$$\lambda : V^*V_k \rightarrow V$$

such that

$$\lambda(v_1v_2\ldots v_n) \in \text{Succ}(v_n) \text{ for all } v_1, \ldots, v_{n-1} \in V \text{ and } v_n \in V_k$$
Strategies outcome

Graph: nondeterministic generator of behaviors.

Strategy: deterministic selector of behavior.

Graph + Strategies for both players $\rightarrow$ Behavior
Given strategies $\lambda_k$ for Player $k$ ($k = 1, 2$), the **outcome** of $\langle \lambda_1, \lambda_2 \rangle$ is the play $w = v_0v_1 \ldots$ such that:

$$v_i \in V_k \rightarrow v_{i+1} = \lambda_k(v_0 \ldots v_i)$$

for all $i \geq 0$ and $k \in \{1, 2\}$

This play is denoted $\text{Outcome}(G, \lambda_1, \lambda_2)$
Winning strategies

• Given a game $G$ and winning conditions $W_1$ and $W_2$, a strategy $\lambda_k$ is **winning** for Player $k$ in $(G,W_k)$ if for all strategies $\lambda_{3-k}$ of Player 3-$k$, the outcome of $\{\lambda_k, \lambda_{3-k}\}$ in $G$ is a winning play of $W_k$.

• Player 1 is winning if $\exists \lambda_1 \cdot \forall \lambda_2 : \text{Outcome}(G, \lambda_1, \lambda_2) \in W_1$

• Player 2 is winning if $\exists \lambda_2 \cdot \forall \lambda_1 : \text{Outcome}(G, \lambda_1, \lambda_2) \in W_2$
Winning strategies

= Controllers that enforce winning plays
Symbolic algorithms to solve games
Given $\mathcal{T} \subseteq V$, let

- $\exists \text{CPre}(\mathcal{T}) = \{v \in V \mid \exists v' \in \text{Succ}(v) : v' \in \mathcal{T}\}$
Controllable predecessors

Given $\mathcal{T} \subseteq V$, let

- $\exists\text{Pre}(\mathcal{T}) = \{v \in V \mid \exists v' \in \text{Succ}(v) : v' \in \mathcal{T}\}$
- $\forall\text{Pre}(\mathcal{T}) = \{v \in V \mid \forall v' \in \text{Succ}(v) : v' \in \mathcal{T}\}$
Controllable predecessors

Given $\mathcal{T} \subseteq V$, let

- $\exists \text{CPre}(\mathcal{T}) = \{v \in V \mid \exists v' \in \text{Succ}(v) : v' \in \mathcal{T}\}$
- $\forall \text{CPre}(\mathcal{T}) = \{v \in V \mid \forall v' \in \text{Succ}(v) : v' \in \mathcal{T}\}$

From a state $v$, Player 1 can **force** the next position of the game to be in $\mathcal{T}$ if:

$$v \in \left( \exists \text{CPre}(\mathcal{T}) \cap V_1 \right) \cup \left( \forall \text{CPre}(\mathcal{T}) \cap V_2 \right)$$

$$\underbrace{1 \text{CPre}(\mathcal{T})}$$
Controllable predecessors

\[ 1\text{CPre}(\mathcal{T}) ::= (\exists\text{CPre}(\mathcal{T}) \cap V_1) \cup (\forall\text{CPre}(\mathcal{T}) \cap V_2) \]

and symmetrically

\[ 2\text{CPre}(\mathcal{T}) ::= (\forall\text{CPre}(\mathcal{T}) \cap V_1) \cup (\exists\text{CPre}(\mathcal{T}) \cap V_2) \]

Note: \( \mathcal{T}' \subseteq \mathcal{T} \) implies

\[
\left\{ \begin{array}{l}
1\text{CPre}(\mathcal{T}') \subseteq 1\text{CPre}(\mathcal{T}) \\
2\text{CPre}(\mathcal{T}') \subseteq 2\text{CPre}(\mathcal{T})
\end{array} \right.
\]

1CPre(\cdot) and 2CPre(\cdot) are **monotone** functions.
Symbolic algorithm to solve safety games
Solving safety games

\[ \text{Safe}(\mathcal{T}) = \{v_0 v_1 \cdots \in V^\omega \mid \forall i : v_i \in \mathcal{T}\} \]

Avoid \( V \setminus \mathcal{T} \) forever

To win a safety game, Player 1 should be able to force the game to be in \( \mathcal{T} \) at every step.
Solving safety games

To win a safety game, Player 1 should be able to force the game to be in $\mathcal{T}$ at every step.

States in which Player 1 can force the game to stay in $\mathcal{T}$ for the next:

0 step: $X_0 = \mathcal{T}$
To win a safety game, Player 1 should be able to force the game to be in $\mathcal{T}$ at every step.

States in which Player 1 can force the game to stay in $\mathcal{T}$ for the next:

0 step: $X_0 = \mathcal{T}$
1 step: $X_1 = \mathcal{T} \cap \text{Pre}(\mathcal{T})$
Solving safety games

To win a safety game, Player 1 should be able to force the game to be in $\mathcal{T}$ at every step.

States in which Player 1 can force the game to stay in $\mathcal{T}$ for the next:

0 step: $X_0 = \mathcal{T}$

1 step: $X_1 = \mathcal{T} \cap 1\text{CPre}(\mathcal{T})$

2 steps: $X_2 = \mathcal{T} \cap 1\text{CPre}(\mathcal{T}) \cap 1\text{CPre}(\mathcal{T} \cap 1\text{CPre}(\mathcal{T}))$
To win a safety game, Player 1 should be able to force the game to be in $\mathcal{T}$ at every step.

States in which Player 1 can force the game to stay in $\mathcal{T}$ for the next:

0 step: $X_0 = \mathcal{T}$
1 step: $X_1 = \mathcal{T} \cap 1\text{CPre}(\mathcal{T})$
2 steps: $X_2 = \mathcal{T} \cap 1\text{CPre}(\mathcal{T}) \cap 1\text{CPre}(\mathcal{T} \cap 1\text{CPre}(\mathcal{T}))$ subset of $\mathcal{T}$
To win a safety game, Player 1 should be able to force the game to be in $\mathcal{T}$ at every step.

States in which Player 1 can force the game to stay in $\mathcal{T}$ for the next:

0 step: $X_0 = \mathcal{T}$

1 step: $X_1 = \mathcal{T} \cap 1\text{CPre}(\mathcal{T})$

2 steps: $X_2 = \mathcal{T} \cap 1\text{CPre}(\mathcal{T} \cap 1\text{CPre}(\mathcal{T}))$
Solving safety games

To win a safety game, Player 1 should be able to force the game to be in $\mathcal{T}$ at every step.

States in which Player 1 can force the game to stay in $\mathcal{T}$ for the next:

0 step: $X_0 = \mathcal{T}$

1 step: $X_1 = \mathcal{T} \cap 1\text{CPre}(X_0)$

2 steps: $X_2 = \mathcal{T} \cap 1\text{CPre}(X_1)$
To win a safety game, Player 1 should be able to force the game to be in $\mathcal{T}$ at every step.

States in which Player 1 can force the game to stay in $\mathcal{T}$ for the next:

0 step: $X_0 = \mathcal{T}$
1 step: $X_1 = \mathcal{T} \cap 1\text{CPre}(X_0)$
2 steps: $X_2 = \mathcal{T} \cap 1\text{CPre}(X_1)$

\[ \vdots \]

n steps: $X_n = \mathcal{T} \cap 1\text{CPre}(X_{n-1})$
\[ T = V \setminus \{v_7\} \]

Objective: \( \text{Safe}(T) \)
Solving safety games

\[ \mathcal{T} = V \setminus \{v_7\} \]

Objective: Safe(\(\mathcal{T}\))

\[ X_0 = \mathcal{T} \]
\[ \mathcal{T} = V \setminus \{v_7\} \]

Objective: Safe(\(\mathcal{T}\))

\[ X_0 = \mathcal{T} \]
\[ X_1 = \mathcal{T} \cap \text{1CPre}(X_0) \]
\[ \mathcal{T} = V \setminus \{v_7\} \]

Objective: Safe(\(\mathcal{T}\))

\[ X_0 = \mathcal{T} \]
\[ X_1 = \mathcal{T} \cap 1\text{CPre}(X_0) \]
Solving safety games

\[ T = V \setminus \{v_7\} \]

Objective: \( \text{Safe}(T) \)

\[ X_0 = T \]
\[ X_1 = T \cap 1\text{CPre}(X_0) \]
Solving safety games

\[ T = V \setminus \{v_7\} \]

**Objective:** Safe(\( T \))

\[ X_0 = T \]

\[ X_1 = T \cap \text{1CPre}(X_0) \]

\[ X_2 = T \cap \text{1CPre}(X_1) \]
Solving safety games

\[ T = V \setminus \{v_7\} \]

Objective: Safe(\( T \))

\[
\begin{align*}
X_0 &= T \\
X_1 &= T \cap 1\text{CPre}(X_0) \\
X_2 &= T \cap 1\text{CPre}(X_1)
\end{align*}
\]
Solving safety games

\( T = V \setminus \{v_7\} \)

Objective: \( \text{Safe}(T) \)

\[ X_0 = T \]
\[ X_1 = T \cap 1\text{CPre}(X_0) \]
\[ X_2 = T \cap 1\text{CPre}(X_1) \]
This is the set of states from which Player 1 can confine the game in $\mathcal{T}$ forever no matter how Player 2 behaves.
$X_2$ is a solution of the set-equation $X = \mathcal{T} \cap 1\text{CPre}(X)$ and it is the greatest solution.
$X_2$ is a solution of the set-equation $X = \mathcal{T} \cap 1\text{CPre}(X)$

and it is the greatest solution.

We say that $X_2$ is the greatest fixpoint of the function $\mathcal{T} \cap 1\text{CPre}(\cdot)$, written:

$$X_2 = \nu X \cdot \mathcal{T} \cap 1\text{CPre}(X)$$

greatest fixpoint operator
On fixpoint computations
A partially ordered set $\langle S, \sqsubseteq \rangle$ is a set $S$ equipped with a **partial order** $\sqsubseteq$, i.e. a relation such that:

<table>
<thead>
<tr>
<th>Condition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\forall x \quad x \sqsubseteq x$</td>
<td>(reflexivity)</td>
</tr>
<tr>
<td>$\forall x, y, z \quad \text{if } x \sqsubseteq y \text{ and } y \sqsubseteq z \text{ then } x \sqsubseteq z$</td>
<td>(transitivity)</td>
</tr>
<tr>
<td>$\forall x, y \quad \text{if } x \sqsubseteq y \text{ and } y \sqsubseteq x \text{ then } x = y$</td>
<td>(anti-symmetry)</td>
</tr>
</tbody>
</table>

$\sqsubseteq$ is not necessarily total, i.e. there can be $x, y$ such that $x \not\sqsubseteq y$ and $y \not\sqsubseteq x$. 
Let $X \subseteq S$.

**y** is an **upper bound** of $X$ if $x \sqsubseteq y$ for all $x \in X$.

**y** is a **least upper bound** of $X$ if

1. $y$ is an upper bound of $X$, and
2. $y \sqsubseteq y'$ for all upper bounds $y'$ of $X$.

Note: if $X$ has a least upper bound, then it is unique (by anti-symmetry), and we write $y = \text{lub}(X)$. 
Partial order

Examples: $(\mathbb{N}, \leq)$

\[ X = \{3, 5, 7, 8\} \quad \text{lub}(X) = 8 \]
\[ X = \{1, 3, 5, 7, 9, \ldots\} \quad X \text{ has no lub} \]
Partial order

Examples: \( \langle \mathbb{N}, \leq \rangle \)

\[ X = \{3, 5, 7, 8\} \quad \text{lub}(X) = 8 \]
\[ X = \{1, 3, 5, 7, 9, \ldots \} \quad X \text{ has no lub} \]

\( \langle \mathcal{P}(\{0, 1, 2\}), \subseteq \rangle \)

\[ X = \{\{0\}, \{2\}\} \quad \text{lub}(X) = \{0, 2\} \]
A set \( X = \{x_0, x_1, x_2, \ldots \} \) is a chain if \( x_0 \sqsubseteq x_1 \sqsubseteq x_2 \sqsubseteq \ldots \)

The partially ordered set \( \langle S, \sqsubseteq \rangle \) is complete if

1. \( \emptyset \) has a lub, written \( \text{lub}(\emptyset) = \bot \), and
2. every chain \( X \subseteq S \) has a lub.
Fixpoints

Let $f : S \rightarrow S$ be a function.

$f$ is **monotonic** if $x \sqsubseteq y$ implies $f(x) \sqsubseteq f(y)$.

is **continuous** if (1) $f$ is monotonic, and

(2) $f(\text{lub}(X)) = \text{lub}(f(X))$ for every chain $X$.

where $f(X) = \{f(x_0), f(x_1), f(x_2), \ldots\}$

Note: $f(X)$ is a chain (i.e. $f(x_0) \sqsubseteq f(x_1) \sqsubseteq f(x_2) \sqsubseteq \ldots$) by monotonicity, and therefore $\text{lub}(f(X))$ exists.
Let $f : S \rightarrow S$ be a function.

$x$ is a **fixpoint** of $f$ if $x = f(x)$

$x$ is a **least fixpoint** of $f$ if

1. $x$ is a fixpoint of $f$, and
2. $x \subseteq x'$ for all fixpoints $x'$ of $f$. 
Let \( \langle S, \sqsubseteq \rangle \) be a partially ordered set.

If \( \sqsubseteq \) is a complete partial order, and \( f : S \to S \) is a continuous function, then

\[
f \text{ has a least fixpoint, denoted } \text{Lfp}(f)
\]

and \( \text{Lfp}(f) = \text{lub}(\{\bot, f(\bot), f^2(\bot), f^3(\bot), \ldots \}) \)

Proof: exercise.
Kleene-Tarski Theorem

Let \( \langle S, \sqsubseteq \rangle \) be a partially ordered set.

If \( \sqsubseteq \) is a complete partial order, and \( f : S \to S \) is a continuous function, then

\[
f \text{ has a least fixpoint, denoted } \operatorname{Lfp}(f)
\]

and \( \operatorname{Lfp}(f) = \operatorname{lub}(\{ \bot, f(\bot), f^2(\bot), f^3(\bot), \ldots \}) \)

Proof: exercise.

Over finite sets \( S \), all monotonic functions are continuous.
Kleene-Tarski Theorem

The greatest fixpoint of $f$ can be defined dually by

$$gfp(f) = glb\left(\{\top, f(\top), f^2(\top), f^3(\top), \ldots \}\right)$$

where $glb(\cdot)$ is the greatest lower bound operator (dual of $lub(\cdot)$) and $glb(\emptyset) = \top$

and $lfp(f) = lub\left(\{\bot, f(\bot), f^2(\bot), f^3(\bot), \ldots \}\right)$

Proof: exercise.

Over finite sets $S$, all monotonic functions are continuous.
Safety game

Winning states of a safety game: \( \nu X \cdot \mathcal{T} \cap 1\text{CPre}(X) \)
\[ \text{gfp}(\mathcal{T} \cap 1\text{CPre}(X)) \]

Limit of the iterations:
- \( X_0 = \mathcal{T} \cap 1\text{CPre}(V) \)
- \( X_1 = \mathcal{T} \cap 1\text{CPre}(X_0) \)
- \( X_2 = \mathcal{T} \cap 1\text{CPre}(X_1) \)
- \( \vdots \)

Partial order: \( \langle 2^V, \subseteq \rangle \) with \( \top = V, \bot = \emptyset \).
Symbolic algorithm to solve reachability games
Solving reachability games

\[ \text{Reach}(\mathcal{T}) = \{v_0v_1 \cdots \in V^\omega \mid \exists i : v_i \in \mathcal{T} \} \]

Visit \( \mathcal{T} \) eventually

To win a reachability game, Player 1 should be able to force the game be in \( \mathcal{T} \) after finitely many steps.
Solving reachability games

Reach(\mathcal{T}) = \{v_0v_1\cdots \in V^\omega \mid \exists i : v_i \in \mathcal{T}\} \quad \text{Visit } \mathcal{T} \text{ eventually}

To win a reachability game, Player 1 should be able to force the game be in \mathcal{T} after finitely many steps.

Let \(X_i\) be the set of states from which Player 1 can force the game to be in \(\mathcal{T}\) within at most \(i\) steps:

\[
X_0 = \mathcal{T} \\
X_{i+1} = X_i \cup 1\text{Pre}(X_i) \quad \text{for all } i \geq 0
\]
The limit of this iteration is the **least fixpoint** of the function $\mathcal{T} \cup 1\text{CPre}(\cdot)$, written:

$$\mu X \cdot \mathcal{T} \cup 1\text{CPre}(X)$$
Symbolic algorithms

Let $G = \langle V_1, V_2, \hat{v}, \text{Succ} \rangle$ be a 2-player game graph.

**Theorem**

Player 1 has a winning strategy

in $\langle G, \text{Reach}(T) \rangle$ iff \( \hat{v} \in \mu X \cdot T \cup 1\text{CPre}(X) \)

in $\langle G, \text{Safe}(T) \rangle$ iff \( \hat{v} \in \nu X \cdot T \cap 1\text{CPre}(X) \)

in $\langle G, \text{Büchi}(T) \rangle$ iff \( \hat{v} \in \nu Y \cdot \mu X \cdot 1\text{CPre}(X) \cup (T \cap 1\text{CPre}(Y)) \)

in $\langle G, \text{coBüchi}(T) \rangle$ iff \( \hat{v} \in \mu Y \cdot \nu X \cdot 1\text{CPre}(X) \cap (T \cup 1\text{CPre}(Y)) \)
Memoryless strategies are always sufficient to win parity games, and therefore also for safety, reachability, Büchi and coBüchi objectives.
Remarks (I)

\[ \mathcal{T} = V \setminus \{v_7\} \]

Objective: Safe(\(\mathcal{T}\))

A memoryless winning strategy
Parity games are determined: in every state, either Player 1 or Player 2 has a winning strategy.
Remarks (II)

Parity games are **determined**: in every state, either Player 1 or Player 2 has a winning strategy.

\[
\phi_1 \equiv \exists \lambda_1 \cdot \forall \lambda_2 : \text{Outcome}(G, \lambda_1, \lambda_2) \in \text{Parity}(p)
\]

\[
\phi_2 \equiv \exists \lambda_2 \cdot \forall \lambda_1 : \text{Outcome}(G, \lambda_1, \lambda_2) \not\in \text{Parity}(p)
\]

**Determinacy says**: \( \phi_1 \lor \phi_2 \)

More generally, zero-sum games with Borel objectives are determined [Martin75].
Remarks (II)

For instance, since  \( V^\omega \setminus \text{Safe}(T) = \text{Reach}(V \setminus T) \),

Player 1 does not win \( \langle G, \text{Safe}(T) \rangle \)

iff Player 2 wins \( \langle G, \text{Reach}(V \setminus T) \rangle \).

\[
X_* = \nu X \cdot T \cap 1\text{CPre}(X) \\
X'_* = \mu X' \cdot T' \cup 2\text{CPre}(X')
\]

Claim: if \( T' = V \setminus T \), then \( X'_* = V \setminus X_* \)

Proof: exercise

Hint: show that \( V \setminus 1\text{CPre}(X) = 2\text{CPre}(V \setminus X) \)
Remarks (II)

\[ \mathcal{T} = V \setminus \{v_7\} \]

Objective for Player 1: Safe(\(\mathcal{T}\))

for Player 2: Reach(\(\{v_7\}\))

\[
\begin{align*}
X_0 &= \{v_0, v_1, v_2, v_3, v_4, v_5, v_6\} \\
X_1 &= \{v_0, v_1, v_2, v_3, v_4\} \\
X_2 &= \{v_0, v_1, v_2, v_3, v_4\} \\
X'_0 &= \{v_7\} \\
X'_1 &= \{v_5, v_6, v_7\} \\
X'_2 &= \{v_5, v_6, v_7\}
\end{align*}
\]
$T = V \setminus \{v_7\}$

States in which Player 1 wins for $\text{Safe}(T)$.

States in which Player 2 wins for $\text{Reach}(V \setminus T)$. 
Games of imperfect information
The Synthesis Question

The controller knows the state of the plant ("perfect information"). This, however, is often unrealistic.

- Sensors provide partial information (imprecision),
- Sensors have internal delays,
- Some variables of the plant are invisible,
- etc....
Imperfect information $\Rightarrow$ Observations
Imperfect information → Observations

Diagram:
- $v_0$ to $v_1$ with edge labeled "hot, off, delay"
- $v_1$ to $v_2$ with edge labeled "cold, delay"
- $v_1$ to $v_3$ with edge labeled "delay"
- $v_2$ to $v_4$ with edge labeled "on, off, delay"
- $v_4$ to Bad with edge labeled "off, delay"
- Bad to Obs 0
- Obs 0 to Obs 1
Imperfect information ➔ Observations

\[ v_0 \rightarrow v_1 \rightarrow v_3 \rightarrow \text{Bad} \]

- $v_0$: on, cold
- $v_1$: off, delay
- $v_2$: off, delay
- $v_3$: on, delay
- $v_4$: on, off

Observations:
- Obs 0
- Obs 1
- Obs 2
Imperfect information ➔ Observations

When observing Obs 2, there is no unique good choice: memory is necessary
Playing the game: Player 2 moves a **token** along the edges of the graph, Player 1 does not see the position of the token.

- Player 1 chooses an action (on, off, delay), and then
- Player 2 resolves the nondeterminism and announces the color of the state.
Player 2:

Player 1:
Player 2: \( v_1 \) chooses \( v_1 \), announces \( \text{Obs 0} \)

Player 1: \[ \text{on, off} \]

Player 2: \[ \text{on, delay} \]

Player 1: \[ \text{off, delay} \]
Player 2: \( v_1 \) delay

Player 1: delay plays action \textit{delay}
Player 2: \( v_1 \), delay \( v_3 \) chooses \( v_3 \), announces Obs 2

Player 1: 

![Diagram](image-url)
Player 2: $v_1$ delay $v_3$ off

Player 1: 
- $v_1$ delay
- $v_3$ off
Player 2: \( v_1 \) delay \( v_3 \) off \( v_2 \) ...

Player 1: [Green] delay [Blue] off [Red] ...

Diagram:

- Node \( v_1 \) with transition on to \( v_2 \), on, delay to \( v_3 \), delay to \( v_4 \), on, off to Bad, off, delay to Bad
- Node \( v_2 \) with transition delay to \( v_1 \), delay to \( v_4 \), off, delay to Bad
- Node \( v_3 \) with transition delay to \( v_1 \), on, delay to \( v_4 \), off, delay to Bad
- Node \( v_4 \) with transition off, delay to \( v_3 \), off, delay to Bad

Bad state connected to all other states with off, delay transitions.
Imperfect information

A game graph + Observation structure

\[ G = \langle V, \delta, \text{Succ} \rangle \quad \langle \Sigma, \text{Obs} \rangle \]

- \( \Sigma \) is a finite alphabet,
- \( \text{Obs} \) is a partition of \( V \),
- \( \text{Succ} : V \times \Sigma \rightarrow 2^V \setminus \emptyset \).

\[ \text{Post}_\sigma(s) = \{ v' \in \text{Succ}(v, \sigma) \mid v \in s \} \]

\[ \Sigma = \{ \text{delay}, \text{on}, \text{off} \} \]

\[ \text{Obs} = \{ \{ v_1 \}, \{ v_2 \}, \{ v_3, v_4 \} \} \]

Indistinguishable states belong to the same observation.
Let \( \text{obs}(v) \in \text{Obs} \) be the (unique) observation containing \( v \).
Strategies

Player 1 chooses a letter in \( \Sigma \),
Player 2 resolves nondeterminism.

**An observation-based strategy for Player 1** is a function:

\[
\lambda_1 : \text{Obs}^+ \to \Sigma
\]

A strategy for Player 2 is a function:

\[
\lambda_2 : V^+ \times \Sigma \to V
\]

such that

\[
\lambda_2(v_1 \ldots v_n, \sigma) \in \text{Succ}(v_n, \sigma) \text{ for all } v_1, \ldots, v_n \in V \text{ and } \sigma \in \Sigma
\]
The **outcome** of \( \langle \lambda_1, \lambda_2 \rangle \) is the play

\[ w = v_0v_1\ldots \text{ such that:} \]

\[ v_{i+1} = \lambda_2(v_0\ldots v_i, \sigma) \text{ where } \sigma = \lambda_1(\text{obs}(v_0)\ldots\text{obs}(v_i)) \]

for all \( i \geq 0 \).

This play is denoted \( \text{Outcome}(G, \lambda_1, \lambda_2) \).
A **winning condition** for Player 1 is a set \( U_1 \subseteq \text{Obs}^\omega \) of sequences of observations. The set \( U_1 \) defines the set of winning plays:

\[
W_1 = \{ v_0v_1\cdots | \text{obs}(v_0)\text{obs}(v_1)\cdots \in U_1 \}
\]

Player 1 is winning if

\[
\exists \lambda_1 \cdot \forall \lambda_2 : \text{Outcome}(G, \lambda_1, \lambda_2) \in W_1
\]
Solving games of imperfect information
Games of imperfect information can be solved by a reduction to games of perfect information.

- \( G, \text{Obs} \) → \( G' \) → Winning region

- Imperfect information → Perfect information
  - Subset construction
  - Classical techniques
Subset construction

After a finite prefix of a play, Player 1 has a partial knowledge of the current state of the game: a set of states, called a cell.
After a finite prefix of a play, Player 1 has a partial knowledge of the current state of the game: \textbf{a set of states}, called a \textbf{cell}.
After a finite prefix of a play, Player 1 has a partial knowledge of the current state of the game: a set of states, called a cell.

Initial knowledge: cell \{ \hat{v} \}

Player 1 plays \( \sigma \),

Player 2 chooses \( v_2 \).

Current knowledge: cell \( \{ v_2, v_3 \} \)

\[ \text{Post}_\sigma(\{ \hat{v} \}) \cap o_2 \]
Subset construction

**Imperfect information**

\[ G = \langle V, \hat{v}, \text{Succ} \rangle \]

\[ \langle \Sigma, \text{Obs} \rangle \]

State space

\[ V \]

Initial state

\[ \hat{v} \]

**Perfect information**

\[ G' = \langle V_1', V_2', \hat{v}', \text{Succ}' \rangle \]

\[ V_1' = 2^V \]

\[ V_2' = 2^V \times \Sigma \]

\[ \hat{v}' = \{ \hat{v} \} \]
Subset construction

\[ G = \langle V, \hat{v}, \text{Succ} \rangle \]

\[ \langle \Sigma, \text{Obs} \rangle \]

Transitions

\[ G' = \langle V'_1, V'_2, \hat{v}', \text{Succ}' \rangle \]

\[ \text{Succ}'(s) = \{(s, \sigma) \mid \sigma \in \Sigma\} \]
Subset construction

\[ G = \langle V, \widehat{v}, \text{Succ} \rangle \]
\[ \langle \Sigma, \text{Obs} \rangle \]

Transitions

\[ G' = \langle V'_1, V'_2, \widehat{v}', \text{Succ}' \rangle \]

\[ \text{Succ}'(s) = \{(s, \sigma) \mid \sigma \in \Sigma\} \]
\[ \text{Succ}'(s, \sigma) = \{\text{Post}_\sigma(s) \cap o \mid o \in \text{Obs}\} \]
Subset construction

\[ G = \langle V, \hat{v}, \text{Succ} \rangle \]
\[ \langle \Sigma, \text{Obs} \rangle \]

Parity condition

\[ p : \text{Obs} \rightarrow \mathbb{N} \]

\[ G' = \langle V'_1, V'_2, \hat{v}', \text{Succ}' \rangle \]

\[ p' : V'_1 \cup V'_2 \rightarrow \mathbb{N} \]

\[ p'(s) = p'(s, \sigma) = p(o) \]
where \( s \subseteq o \).
Subset construction

\[ G = \langle V, \hat{v}, \text{Succ} \rangle \]
\[ \langle \Sigma, \text{Obs} \rangle \]

Parity condition
\[ p : \text{Obs} \to \mathbb{N} \]

\[ G' = \langle V'_1, V'_2, \hat{v}', \text{Succ}' \rangle \]

\[ p' : V'_1 \cup V'_2 \to \mathbb{N} \]
\[ p'(s) = p'(s, \sigma) = p(o) \]
where \( s \subseteq o \).

Theorem
Player 1 is winning in \( G, p \) if and only if Player 1 is winning in \( G', p' \).
Imperfect information

$G, \text{Obs}$ \rightarrow $G'$ \rightarrow Winning region

Imperfect information

Perfect information

subset construction

Exponential blow-up

classical techniques
Imperfect information

G, Obs \[\rightarrow\] \[\rightarrow\] G' \[\rightarrow\] Winning region

Imperfect information \[\text{implicit}\] Perfect information

Direct symbolic algorithm
Symbolic algorithm

Controllable predecessor: \( 1\text{CPre} : 2^{V'_1} \rightarrow 2^{V'_1} \)

\[
1\text{CPre}(q) = \{ s \mid \exists (s, \sigma) \in \text{Succ}'(s) \cdot \forall s' \in \text{Succ}'(s, \sigma) : s' \in q \}
\]

\[
= \{ s \mid \exists \sigma \in \Sigma \cdot \forall o \in \text{Obs} : \text{Post}_\sigma(s) \cap o \in q \}
\]
Symbolic algorithm

\[ G = \langle V, \hat{v}, \text{Succ} \rangle \quad \langle \Sigma, \text{Obs} \rangle \]

1CPre(\{\{v_3, v_4\}, \{v_5, v_6\}\}) = \{\{v_1\}, \{v_2\}\}

\[ \neq \{\{v_1, v_2\}\} \]

The union of two controllable cells is not necessarily controllable, but…
Symbolic algorithm

\[
1\text{CPre}(q) = \{ s \mid \exists \sigma \in \Sigma \cdot \forall o \in \text{Obs} : \text{Post}_\sigma(s) \cap o \in q \}
\]

If a cell \( s \) is controllable (i.e. winning for Player 1), then all sub-cells \( s' \subseteq s \) are controllable.

```
S
\cup \cup \cup \cup \cup ... 
```

```
S'
copy the strategy from s
```

\[\text{copy the strategy from } s\]
Symbolic algorithm

\[ 1\text{CPre}(q) = \{ s \mid \exists \sigma \in \Sigma \cdot \forall o \in \text{Obs} : \text{Post}_\sigma(s) \cap o \in q \} \]

The sets of cells computed by the fixpoint iterations are **downward-closed**.

A set \( q \) of cells is downward-closed if \( s \in q \) and \( s' \subseteq s \) implies \( s' \in q \).
Symbolic algorithm

\[ 1\text{CPre}(q) = \{s \mid \exists \sigma \in \Sigma \cdot \forall o \in \text{Obs} : \text{Post}_\sigma(s) \cap o \in q \} \]

The sets of cells computed by the fixpoint iterations are **downward-closed**.

A set \(q\) of cells is downward-closed if \(s \in q\) and \(s' \subseteq s\) implies \(s' \in q\).

It is sufficient to keep only the **maximal cells**.
Antichains

Maximal cells in \( p \):

\[
[p] = \{ s \in p \mid \forall s' \in p : s \not\subset s' \}
\]

\([p]\) is an antichain, i.e. a set of \( \subset \)-incomparable cells.

\[
[p] = \{ s_1, s_2, s_3 \}
\]
Antichains

Maximal cells in $p$: $[p] = \{s \in p \mid \forall s' \in p : s \not\subset s'\}$

$[p]$ is an **antichain**, i.e. a set of $\subseteq$-incomparable cells.

For downward-closed set $p$, we have:

$1\operatorname{CPre}(p) = \{s \mid \exists \sigma \in \Sigma \cdot \forall o \in \text{Obs} : \text{Post}_\sigma(s) \cap o \in p\}$

$= \{s \mid \exists \sigma \in \Sigma \cdot \forall o \in \text{Obs} \cdot \exists s' \in [p] : \text{Post}_\sigma(s) \cap o \subseteq s'\}$

Hence, over antichains we define:

$1\operatorname{CPre}^A(q) = \big\{s \mid \exists \sigma \in \Sigma \cdot \forall o \in \text{Obs} \cdot \exists s' \in q : \text{Post}_\sigma(s) \cap o \subseteq s'\big\}$
Antichains

1CPre(·) is monotone with respect to the following order:

$q \sqsubseteq q'$ iff $\forall s \in q \cdot \exists s' \in q' : s \subseteq s'$

$\langle A, \sqsubseteq \rangle$ is a complete partial order.

Least upper bound and greatest lower bound are defined by:

$q \sqcup q' = \left[ \{ s \mid s \in q \lor s \in q' \} \right]
q \sqcap q' = \left[ \{ s \cap s' \mid s \in q \land s' \in q' \} \right]$
Symbolic algorithms

Let \( G = \langle V, \hat{v}, \text{Succ}, \Sigma, \text{Obs} \rangle \) be a 2-player game graph of imperfect information, and \( \mathcal{T} \subseteq \text{Obs} \) a set of observations.

Games of imperfect information can be solved by the same fixpoint formulas as for perfect information, namely:

**Theorem**

Player 1 has a winning strategy

\[
\begin{align*}
\text{in } \langle G, \text{Reach}(\mathcal{T}) \rangle & \quad \text{iff} \quad \{\hat{v}\} \subseteq \mu X \cdot \mathcal{T} \sqcup 1\text{CPre}(X) \\
\text{in } \langle G, \text{Safe}(\mathcal{T}) \rangle & \quad \text{iff} \quad \{\hat{v}\} \subseteq \nu X \cdot \mathcal{T} \cap 1\text{CPre}(X) \\
\text{in } \langle G, \text{Büchi}(\mathcal{T}) \rangle & \quad \text{iff} \quad \{\hat{v}\} \subseteq \nu Y \cdot \mu X \cdot 1\text{CPre}(X) \sqcup (\mathcal{T} \cap 1\text{CPre}(Y)) \\
\text{in } \langle G, \text{coBüchi}(\mathcal{T}) \rangle & \quad \text{iff} \quad \{\hat{v}\} \subseteq \mu Y \cdot \nu X \cdot 1\text{CPre}(X) \cap (\mathcal{T} \sqcup 1\text{CPre}(Y))
\end{align*}
\]
Solving safety games

\[ \mathcal{T} = \text{Obs} \setminus \{o_3\} \]

Objective: \( \text{Safe}(\mathcal{T}) \)
We compute the fixpoint \( \nu X \cdot \mathcal{T} \sqcap 1C\text{Pre}(X) \).
Solving safety games

\[ T = \text{Obs} \setminus \{o_3\} \]

Objective: \( \text{Safe}(T) \)

\[ X_0 = T = \{\{v_0, v_1\}, \{v_2\}\} \]
Solving safety games

\[ \mathcal{T} = \text{Obs} \setminus \{o_3\} \]

Objective: \( \text{Safe}(\mathcal{T}) \)

\[ X_0 = \{\{v_0, v_1\}, \{v_2\}\} \]

\[ X_1 = \text{CPre}(X_0) \cap \mathcal{T} = \left\{\{v_1\}_b, \{v_0, v_2\}_a\right\} \cap \mathcal{T} \]
Solving safety games

\[ \mathcal{T} = \text{Obs} \setminus \{o_3\} \]

Objective: \( \text{Safe}(\mathcal{T}) \)

\[ X_0 = \{\{v_0, v_1\}, \{v_2\}\} \]

\[ X_1 = \text{CPre}(X_0) \cap \mathcal{T} = \{\{v_1\}_b, \{v_0, v_2\}_a\} \cap \mathcal{T} \]
\( T = \text{Obs} \setminus \{o_3\} \)

Objective: Safe(\( T \))

\[
X_0 = \{\{v_0, v_1\}, \{v_2\}\}
\]

\[
X_1 = \{\{v_0\}, \{v_1\}, \{v_2\}\}
\]
Solving safety games

\[ \mathcal{T} = \text{Obs} \setminus \{o_3\} \]

Objective: \( \text{Safe}(\mathcal{T}) \)

\[ X_0 = \{\{v_0, v_1\}, \{v_2\}\} \]

\[ X_1 = \{\{v_0\}, \{v_1\}, \{v_2\}\} \]

\[ X_2 = \text{CPre}(X_1) \cap \mathcal{T} = \{\{v_0\}_a, \{v_1\}_b, \{v_2\}_a\} \cap \mathcal{T} \]
Solving safety games

\[ \mathcal{T} = \text{Obs} \setminus \{o_3\} \]

Objective: Safe(\(\mathcal{T}\))

\[ X_0 = \{\{v_0, v_1\}, \{v_2\}\} \]
\[ X_1 = \{\{v_0\}, \{v_1\}, \{v_2\}\} \]

\[ X_2 = \text{CPre}(X_1) \cap \mathcal{T} = \{\{v_0\}_a, \{v_1\}_b, \{v_2\}_a\} \cap \mathcal{T} \]
Solving safety games

\[ \mathcal{T} = \text{Obs} \setminus \{o_3\} \]

Objective: Safe(\(\mathcal{T}\))

\[ X_0 = \{\{v_0, v_1\}, \{v_2\}\} \]

\[ X_1 = \{\{v_0\}, \{v_1\}, \{v_2\}\} \]

\[ X_2 = \text{CPre}(X_1) \cap \mathcal{T} = \{\{v_0\}_a, \{v_1\}_b, \{v_2\}_a\} \cap \mathcal{T} \]
Solving safety games

\[ T = \text{Obs} \backslash \{o_3\} \]

Objective: Safe(\( T \))

\[ X_0 = \{\{v_0, v_1\}, \{v_2\}\} \]
\[ X_1 = \{\{v_0\}, \{v_1\}, \{v_2\}\} \]
\[ X_2 = \{\{v_0\}, \{v_1\}, \{v_2\}\} \]
Solving safety games

\( T = \text{Obs} \setminus \{o_3\} \)

Objective: Safe(\( T \))

\[
X_0 = \{ \{v_0, v_1\}, \{v_2\} \}
\]

\[
X_1 = \{ \{v_0\}, \{v_1\}, \{v_2\} \}
\]

\[
X_2 = \{ \{v_0\}, \{v_1\}, \{v_2\} \}
\]

Fixed point
Solving safety games

\[ T = \text{Obs} \setminus \{o_3\} \]

Objective: Safe(\(T\))

\[
X_0 = \{\{v_0, v_1\}, \{v_2\}\}
\]

\[
X_1 = \{\{v_0\}, \{v_1\}, \{v_2\}\}
\]

\[
X_2 = \{\{v_0\}, \{v_1\}, \{v_2\}\}
\]

Fixed point

Player 1 is winning since \(\{v_0\} \in X_2\)
Solving safety games

$T = \text{Obs} \setminus \{o_3\}$

Objective: Safe($T$)

$X_0 = \{\{v_0, v_1\}, \{v_2\}\}$

$X_1 = \{\{v_0\}, \{v_1\}, \{v_2\}\}$

$X_2 = \{\{v_0\}, \{v_1\}, \{v_2\}\}$

Fixed point

A winning strategy:
1. **Finite memory** may be necessary to win safety and reachability games of imperfect information, and therefore also for Büchi, coBüchi, and parity objectives.
Remarks

1. **Finite memory** may be necessary to win safety and reachability games of imperfect information, and therefore also for Büchi, coBüchi, and parity objectives.

2. Games of imperfect information are **not determined**.
Non determinacy

Any fixed strategy $\lambda_1$ of Player 1 can be spoiled by a strategy $\lambda_2$ of Player 2 as follows:

In $v_0$: $\lambda_2$ chooses $v_1$ if in the next step $\lambda_1$ plays $b$, and $\lambda_2$ chooses $v_2$ if in the next step $\lambda_1$ plays $a$. 

Objective: Reach($\{v_3\}$)
Player 1 cannot enforce $\text{Reach}(\{v_3\})$. 

Similarly, Player 2 cannot enforce $\text{Safe}(\{v_0, v_1, v_2\})$. 

because when a strategy $\lambda_2$ of Player 2 is fixed, either $\lambda_1(o_1o_1) = a$ or $\lambda'_1(o_1o_1) = b$ is a spoiling strategy for Player 1.
Remarks

1. **Finite memory** may be necessary to win safety and reachability games of imperfect information, and therefore also for Büchi, coBüchi, and parity objectives.

2. Games of imperfect information are **not determined**.

3. **Randomized** strategies are more powerful, already for reachability objectives.
The following strategy of Player 1 wins with probability 1:
At every step, play $a$ and $b$ uniformly at random.
After each visit to $\{v_1, v_2\}$, no matter the strategy of Player 2, Player 1 has probability $\frac{1}{2}$ to win (reach $v_3$).
Summary
Conclusion

• Games for controller synthesis: symbolic algorithms using fixpoint formulas.

• Imperfect information is more realistic, gives more robust controllers; but exponentially harder to solve.

• Antichains: exploit the structure of the subset construction.
Conclusion

• Games for controller synthesis: symbolic algorithms using fixpoint formulas.

• Imperfect information is more realistic, gives more robust controllers; but exponentially harder to solve.

• Antichains: exploit the structure of the subset construction.

It is sufficient to keep only the maximal elements.
Conclusion

• The antichain principle has applications in other problems where subset constructions are used:

  • Finite automata: language inclusion, universality, etc.  [De Wulf,D,Henzinger,Raskin 06]

  • Alternating Büchi automata: emptiness and language inclusion.  [D,Raskin 07]

  • LTL: satisfiability and model-checking.  [De Wulf,D,Maquet,Raskin 08]
Alaska

Antichains for Logic, Automata and Symbolic Kripke Structure Analysis

http://www.antichains.be
Acknowledgments

Credits

Antichains for games is a joint work with Krishnendu Chatterjee, Martin De Wulf, Tom Henzinger and Jean-François Raskin.

Special thanks to Jean-François Raskin for slides preparation.
Thank you!

Questions?