

Terminology

Barbara Jobstmann

Cadence Design Systems

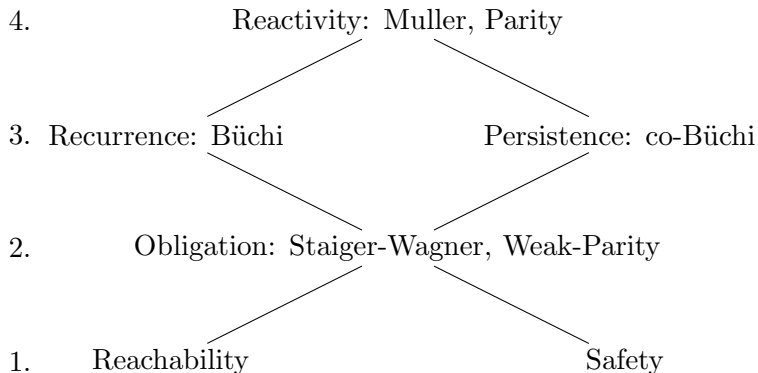
Ecole Polytechnique Fédérale de Lausanne

Grenoble, December 2018

Outline of the course

1. Terminology
2. Games
 - 2.1 Reachability games
 - 2.2 Buchi games
 - 2.3 Obligation games
 - 2.4 Muller games
3. About games and tree automata

Hierarchy



Terminology

Terminology

Two-player games between Player 0 and 1

An **infinite game** $\langle G, \phi \rangle$ consists of

- ▶ a **game graph** G and
- ▶ a **winning condition** ϕ .

G defines the “playground”, in which the two players compete.

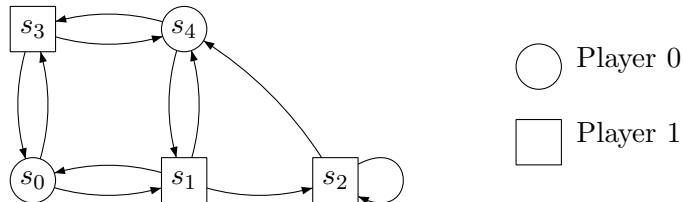
ϕ defines which plays are won by Player 0.

If a play does not satisfy ϕ , then Player 1 wins on this play.

Game Graphs

A **game graph** is a tuple $G = \langle S, S_0, T \rangle$ where:

- ▶ S is a finite set of **states**,
- ▶ $S_0 \subseteq S$ is the set of **Player-0 states** ($S_1 = S \setminus S_0$ are the **Player-1 states**),
- ▶ $T \subseteq S \times S$ is a **transition relation**. We assume that each state has at least one successor.



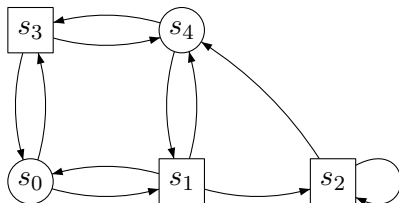
Plays

A **play** is an infinite sequence of states $\rho = s_0 s_1 s_2 \dots \in S^\omega$ such that for all $i \geq 0$ $\langle s_i, s_{i+1} \rangle \in T$.

It starts in s_0 and it is built up as follows:

If $s_i \in S_0$, then Player 0 chooses an edge starting in s_i , otherwise Player 1 picks such an edge.

Intuitively, a token is moved from state to state via edges: From S_0 -states Player 0 moves the token, from S_1 -states Player 1 moves the token.



Winning Condition

The winning condition describes the plays won by Player 0.

A **winning condition or winning objective** ϕ is a subset of plays, i.e., $\phi \subseteq S^\omega$.

We use logical conditions (e.g., LTL formulas) or automata theoretic acceptance conditions to describe ϕ .

Example:

- ▶ $\Box\Diamond s$ for some state $s \in S$
- ▶ All plays that stay within a **safe region** $F \subseteq S$ are in ϕ .
- ▶ Given a priority function $p : S \rightarrow \{0, 1, \dots, d\}$, all plays in which the smallest priority visited is even.

Games are named after their winning condition, e.g., Safety game, Reachability game, LTL game, Parity game,...

Types of Games

Given a play ρ , we define

- ▶ $\text{Occ}(\rho) = \{s \in S \mid \exists i \geq 0 : s_i = s\}$
- ▶ $\text{Inf}(\rho) = \{s \in S \mid \forall i \geq 0 \exists j > i : s_j = s\}$

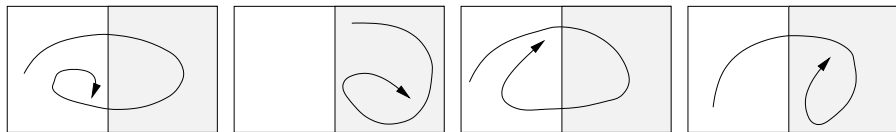
Given a set $F \subseteq S$,

Reachability Game $\phi = \{\rho \in S^\omega \mid \text{Occ}(\rho) \cap F \neq \emptyset\}$

Safety Game $\phi = \{\rho \in S^\omega \mid \text{Occ}(\rho) \subseteq F\}$

Büchi Game $\phi = \{\rho \in S^\omega \mid \text{Inf}(\rho) \cap F \neq \emptyset\}$

Co-Büchi Game $\phi = \{\rho \in S^\omega \mid \text{Inf}(\rho) \subseteq F\}$



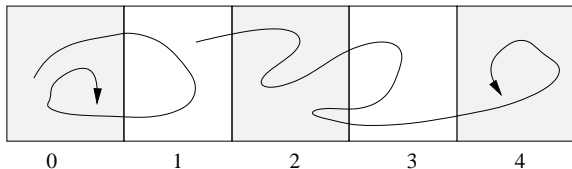
Types of Games

Given a priority function $p : S \rightarrow \{0, 1, \dots, d\}$ or an LTL formula φ

Weak-Parity Game $\phi = \{\rho \in S^\omega \mid \min_{s \in \text{Occ}(\rho)} p(s) \text{ is even}\}$

Parity Game $\phi = \{\rho \in S^\omega \mid \min_{s \in \text{Inf}(\rho)} p(s) \text{ is even}\}$

LTL Game $\phi = \{\rho \in S^\omega \mid \rho \models \varphi\}$



We will refer to the type of a game and give F , p , or φ instead of defining ϕ .

We will also talk about Muller and Rabin games.

Strategies

A **strategy** for Player 0 from state s is a (partial) function

$$f : S^*S_0 \rightarrow S$$

specifying for any sequence of states s_0, s_1, \dots, s_k with $s_0 = s$ and $s_k \in S_0$ a successor state s_j such that $(s_k, s_j) \in T$.

A play $\rho = s_0s_1\dots$ is **compatible** with strategy f if for all $s_i \in S_0$ we have that $s_{i+1} = f(s_0s_1\dots s_i)$.

(Definitions for Player 1 are analogous.)

Given strategies f and g from s for Player 0 and 1, respectively. We denote by $G_{f,g}$ the (unique) play that is compatible with f and g .

Winning Strategies and Regions

Given a game (G, ϕ) with $G = (S, S_0, E)$, a strategy f for Player 0 from s is called a **winning strategy** if for all Player-1 strategies g from s , if $G_{f,g} \in \phi$ holds. Analogously, a Player-1 strategy g is winning if for all Player-0 strategies f , $G_{f,g} \notin \phi$ holds.

Player 0 (resp. 1) wins from s if s/he has a winning strategy from s .

Winning Strategies and Regions

Given a game (G, ϕ) with $G = (S, S_0, E)$, a strategy f for Player 0 from s is called a **winning strategy** if for all Player-1 strategies g from s , if $G_{f,g} \in \phi$ holds. Analogously, a Player-1 strategy g is winning if for all Player-0 strategies f , $G_{f,g} \notin \phi$ holds.

Player 0 (resp. 1) wins from s if s/he has a winning strategy from s .

The **winning regions** of Player 0 and 1 are the sets

$$W_0 = \{s \in S \mid \text{Player 0 wins from } s\}$$

$$W_1 = \{s \in S \mid \text{Player 1 wins from } s\}$$

Note each state s belongs at most to W_0 or W_1 . Otherwise pick winning strategies f and g from s for Player 0 and 1, respectively, then $G_{f,g} \in \phi$ and $G_{f,g} \notin \phi$: Contradiction.

Questions About Games

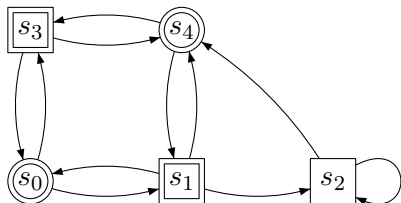
Solve a game (G, ϕ) with $G = (S, S_0, T)$:

1. Decide for each state $s \in S$ if $s \in W_0$.
2. If yes, construct a suitable winning strategy from s .

Further interesting question:

- ▶ Optimize construction of winning strategy (e.g., time complexity)
- ▶ Optimize parameters of winning strategy (e.g., size of memory)

Example



Safety game (G, F) with $F = \{s_0, s_1, s_3, s_4\}$, i.e., $\text{Occ}(\rho) \subseteq F$

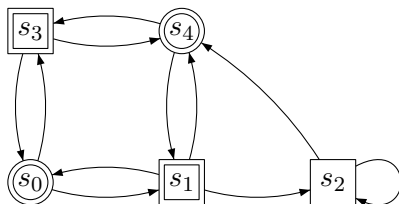
A winning strategy for Player 0 (from state s_0 , s_3 , and s_4):

- ▶ From s_0 choose s_3 and from s_4 choose s_3

A winning strategy for Player 1 (from state s_1 and s_2):

- ▶ From s_1 choose s_2 , from s_2 choose s_4 , and from s_3 choose s_4

Example



Safety game (G, F) with $F = \{s_0, s_1, s_3, s_4\}$, i.e., $\text{Occ}(\rho) \subseteq F$

A winning strategy for Player 0 (from state s_0 , s_3 , and s_4):

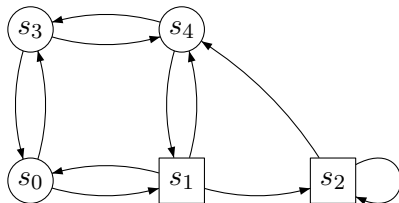
- ▶ From s_0 choose s_3 and from s_4 choose s_3

A winning strategy for Player 1 (from state s_1 and s_2):

- ▶ From s_1 choose s_2 , from s_2 choose s_4 , and from s_3 choose s_4

$W_0 = \{s_0, s_3, s_4\}$, $W_1 = \{s_1, s_2\}$

Another Example



LTL game (G, φ) with $\varphi = \diamond s_0 \wedge \diamond s_4$ (visit s_0 and s_4)

Winning strategy for Player 0 from s_0 :

- ▶ From s_0 to s_3 , from s_3 to s_4 , and from s_4 to s_1 .

Note: this strategy is not winning from s_3 or s_4 .

Winning strategy for Player 0 from s_3 :

- ▶ From s_0 to s_3 , from s_4 to s_3 , and from s_3 to s_0 on first visit, otherwise to s_4 .

Determinacy

Recall: the winning regions are disjoint, i.e., $W_0 \cap W_1 = \emptyset$

Question: Is every state winning for some player?

A game (G, ϕ) with $G = (S, S_0, E)$ is called **determined** if $W_0 \cup W_1 = S$ holds.

Remarks:

1. We will show that all automata theoretic games we consider here are determined.
2. There are games which are not determined (e.g., concurrent games: even/odd sum, paper-rock-scissors)

Strategy Types

In general, a strategy is a function $f : S^+ \rightarrow S$.

(Note that sometimes we might define f only partially.)

1. **Computable or recursive strategies:** f is computable
2. **Finite-state strategies:** f is computable with a finite-state automaton meaning that f has bounded information about the past (history).
3. **Memoryless or positional strategies:** f only depends on the current state of the game (no knowledge about history of play)

Positional Strategies

Given a game (G, ϕ) with $G = (S, S_0, E)$, a strategy $f : S^+ \rightarrow S$ is called **positional or memoryless** if for all words $w, w' \in S^+$ with $w = s_0 \dots s_n$ and $w' = s'_0 \dots s'_m$ such that $s_n = s'_m$, $f(w) = f(w')$ holds.

A positional strategy for Player 0 is representable as

1. a function $f : S_0 \rightarrow S$
2. a set of edges containing for every Player-0 state s exactly one edge starting in s (and for every Player-1 state s' all edges starting in s')

Finite-state Strategies

A **strategy automaton** over a game graph $G = (S, S_0, E)$ is a finite-state machine $A = (M, m_0, \delta, \lambda)$ (Mealy machine) with input and output alphabet S , where

- ▶ M is a finite set of states (called **memory**),
- ▶ $m_0 \in M$ is an initial state (the initial **memory content**),
- ▶ $\delta : M \times S \rightarrow M$ is a transition function (the **memory update fct**),
- ▶ $\lambda : M \times S \rightarrow S$ is a labeling function (called the **choice function**).

Finite-state Strategies

A **strategy automaton** over a game graph $G = (S, S_0, E)$ is a finite-state machine $A = (M, m_0, \delta, \lambda)$ (Mealy machine) with input and output alphabet S , where

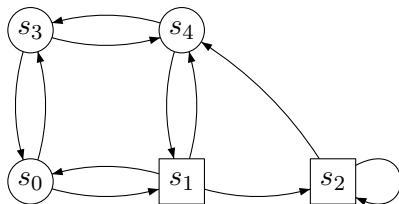
- ▶ M is a finite set of states (called **memory**),
- ▶ $m_0 \in M$ is an initial state (the initial **memory content**),
- ▶ $\delta : M \times S \rightarrow M$ is a transition function (the **memory update fct**),
- ▶ $\lambda : M \times S \rightarrow S$ is a labeling function (called the **choice function**).

The strategy for Player 0 computed by A is the function

$$f_A(s_0 \dots s_k) := \lambda(\delta(m_0, s_0 \dots s_{k-1}), s_k) \text{ with } s_k \in S_0$$

and the usual extension of δ to words: $\delta(m_0, \epsilon) = m_0$ and $\delta(m_0, s_0 \dots s_k) = \delta(\delta(m_0, s_0 \dots s_{k-1}), s_k)$. Any strategy f , such that there exists an A with $f_A = f$, is called **finite-state strategy**.

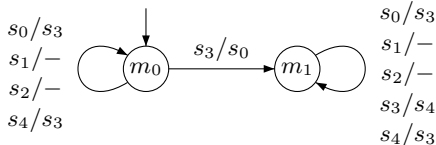
Recall Example



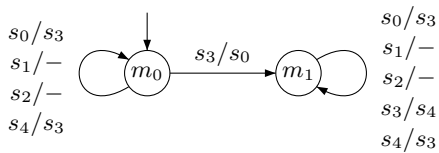
Objective: visit s_0 and s_4 , i.e., $\{s_0, s_4\} \subseteq \text{Occ}(\rho)$

Winning strategy for Player 0 from s_0 , s_3 and s_4 :

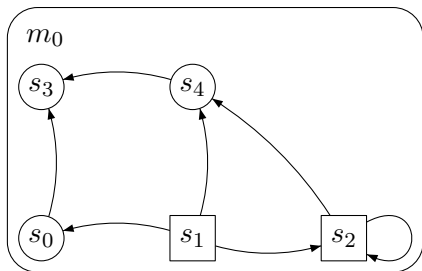
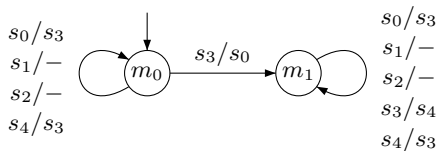
- ▶ From s_0 to s_3 , from s_4 to s_3 , and from s_3 to s_0 on first visit, otherwise to s_4 .



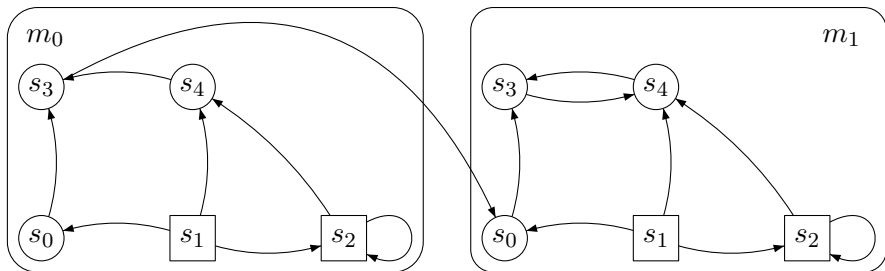
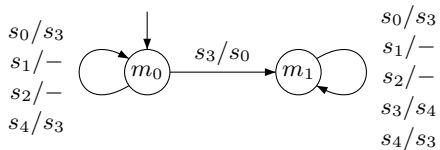
Extended Game



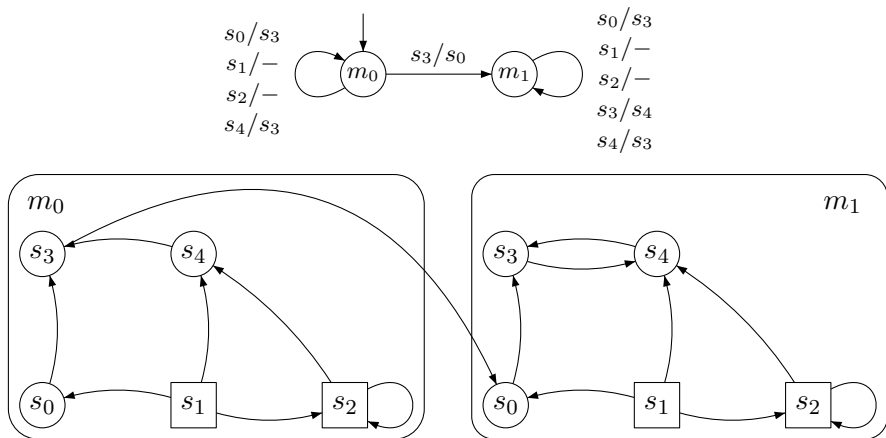
Extended Game



Extended Game



Extended Game



Note: the strategy in the extended game graph is memoryless.