Infinite Games

Barbara Jobstmann

Cadence Design Systems
Ecole Polytechnique Fédérale de Lausanne

Build Correct HW/SW Systems

- Use logic to specify correctness properties, e.g.:
  - *every job sent to the printer is eventually printed*
  - *two jobs do not overlap (only one job is printed at a time)*
  - *a job that is canceled will be interupted*

  These are conditions on infinite sequences (system runs), and can be specified by automata and logical formulas.

Build Correct HW/SW Systems

- Use logic to specify correctness properties, e.g.:
  - *every job sent to the printer is eventually printed*
  - *two jobs do not overlap (only one job is printed at a time)*
  - *a job that is canceled will be interupted*

  These are conditions on infinite sequences (system runs), and can be specified by automata and logical formulas.
- Given a logical specification, we can do either:
  - VERIFICATION: prove that a given system satisfies the specification
  - SYNTHESIS: build a system that satisfies the specification

Example: Elevator

- Aim: build controller that moves elevator of 10 floor building
- Environment: Passengers pressing buttons to (1) call elevator and (2) request floor
- System state:
    1. Set of requested floor numbers: $\{0,1\}^{10}$
    2. Current position of lift: $\{1,\ldots,10\}$
    3. Indicator whose turn is next (assuming lift and passengers act in alternation) $\{0,1\}$

Infinite Games

Two players:

1. Controller is Player 0
2. Passengers are Player 1

A play of a game is an infinite sequence of states of elevator transition system, where the two players choose moves alternatively.

How does the transition system look like?

- State space: $\{0,1\}^{10} \times \{1,\ldots,10\} \times \{0,1\}$
- Transitions:
  - Player 0: $(r_1 \ldots r_{10}, j, 0) \to [r'_1 \ldots r'_{10}, j', 1]$ s.t. $r'_j = 0$, $\forall_{i \neq j} r_i = r'_i$
    Actions: open/closes doors and move lift
  - Player 1: $[r_1 \ldots r_{10}, j, 1] \to (r'_1 \ldots r'_{10}, j', 0)$ s.t. $j = j'$, $\forall i : r_i \leq r'_i$
    Actions: request floors

Desired Properties

- Every requested floor is eventually reached
- Floors along the way are served if requested
- If no floor is request, elevator goes to ground floor
- ...

These are conditions on infinite sequences!

Player 0 (controller) wins the play if all conditions are satisfied independent of the choices Player 1 makes. This corresponds to finding a winning strategy for Player 0 in an infinite game.

Our Aim

Solution of the Synthesis Problem

1. Decide whether there exists such a winning strategy - Realizability Problem
2. If "yes", then construct the system - Synthesis Problem

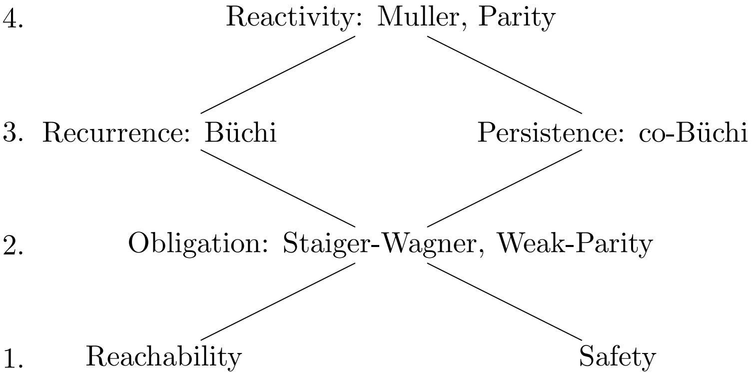
Main result:
The synthesis problem is algorithmically solvable for finite-state systems with respect to specifications given as ω-automata or linear-time temporal logic.

Other: Nicer and more intuitive proofs for logics over trees

Outline

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


4. Reactivity: Muller, Parity
3. Recurrence: Büchi
Persistence: co-Büchi
2. Obligation: Staiger-Wagner, Weak-Parity
1. Reachability
Safety

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.

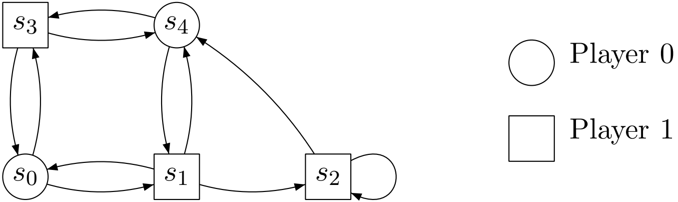
ϕ defines which plays are won by Player 0.

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.



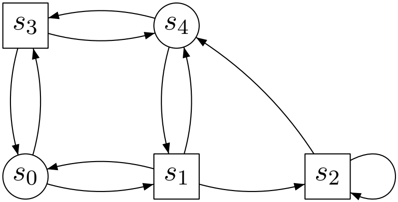
Plays

A play is an infinite sequence of states $\rho = s_0 s_1 s_2 \cdots \in S^\omega$ 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 $p : S \to \{0, 1, \ldots, d\}$, 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,...

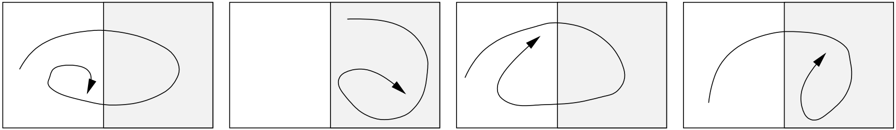
Types of Games

Given a play ρ, we define

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

Given a set $F \subseteq S$,

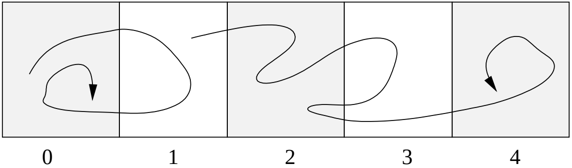
Reachability Game	$\phi = \{\rho \in S^\omega \mid \mathrm{Occ}(\rho) \cap F \neq \emptyset\}$
Safety Game	$\phi = \{\rho \in S^\omega \mid \mathrm{Occ}(\rho) \subseteq F\}$
Büchi Game	$\phi = \{\rho \in S^\omega \mid \mathrm{Inf}(\rho) \cap F \neq \emptyset\}$
Co-Büchi Game	$\phi = \{\rho \in S^\omega \mid \mathrm{Inf}(\rho) \subseteq F\}$



Types of Games

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

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

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

specifying for any sequence of states $s_0, s_1, \ldots s_k$ 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 $\rho = s_0 s_1 \ldots$ is compatible with strategy f if for all s_i we have that $s_{i+1} = f(s_0 s_1 \ldots s_i)$.
(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, $G_{f,g} \in \phi$ 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, $G_{f,g} \in \phi$ 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

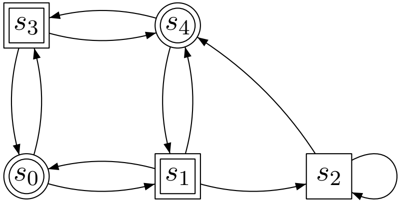
Solve a game (G, ϕ) with $G = (S, S_0, T)$:

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., $\mathrm{Occ}(\rho) \subseteq F$

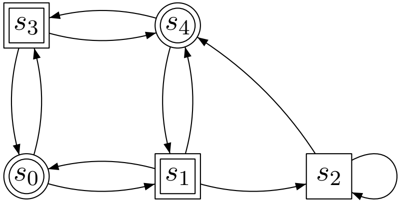
A winning strategy for Player 0 (from state s_0 and s_4):

- 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

Example



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

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

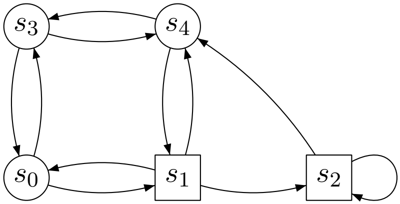
- From s_0 choose s_3 and from s_4 choose s_3

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

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

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 $f : S^+ \to S$.

(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 $w = s_0 \ldots s_n$ and $w' = s'_0 \ldots s'_m$ 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),
- $\delta : M \times S \to M$ is a transition function (the memory update fct),
- $\lambda : M \times S \to S$ 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

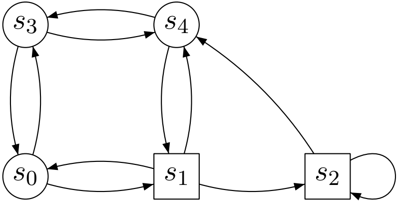
- M is a finite set of states (called memory),
- $m_0 \in M$ is an initial state (the initial memory content),
- $\delta : M \times S \to M$ is a transition function (the memory update fct),
- $\lambda : M \times S \to S$ is a labeling function (called the choice function).

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 $\delta(m_0, s_0 ... s_k) = \delta(\delta(m_0, s_0 ... s_{k-1}), s_k)$. Any strategy f, such that there exists an A with $f_A = f$, is called finite-state strategy.

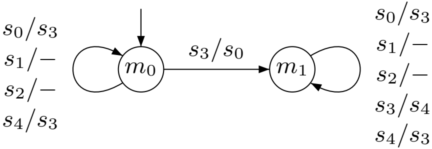
Recall Example



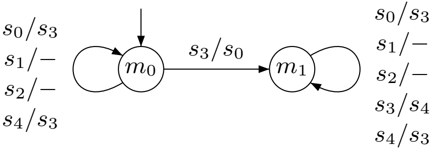
Objective: visit s_0 and s_4, i.e, $\{s_0, s_4\} \subseteq \mathrm{Occ}(\rho)$

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

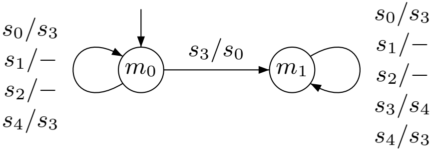
- From s_0 to s_3, from s_4 to s_3, and from s_3 to s_0 on first visit, otherwise to s_4.

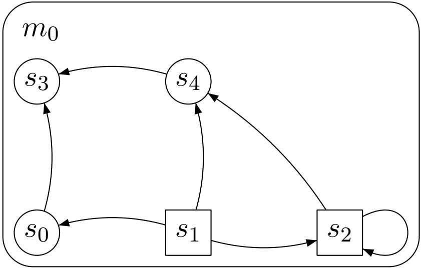


Extended Game

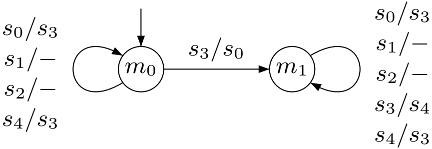


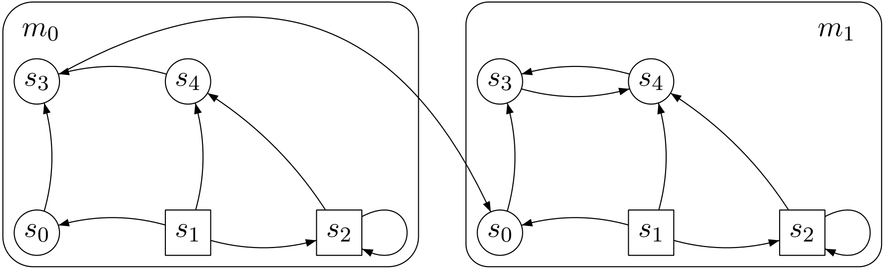
Extended Game



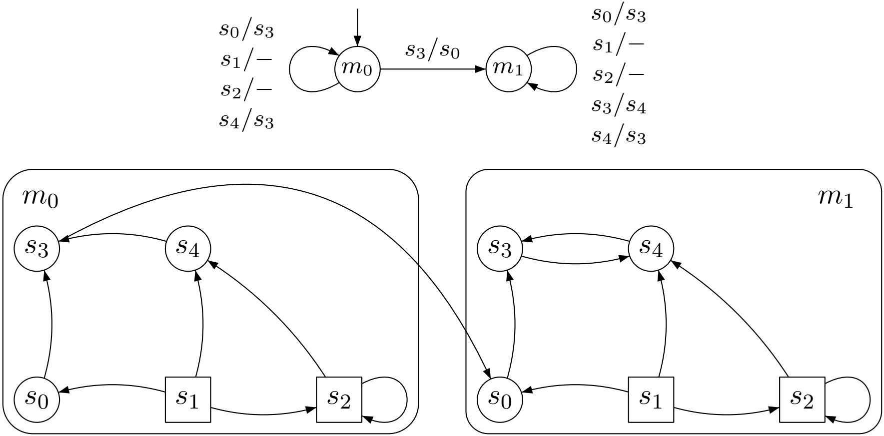


Extended Game





Extended Game



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

Reachability and Safety Games

Reachability and Safety Games

Theorem

Given a reachability game (G, F) with $G = (S, S_0, E)$ and $F \subseteq S$, then the winning regions W_0 and W_1 of Player 0 and 1, respectively, are computable, and both players have corresponding memoryless winning strategies.

Proof.

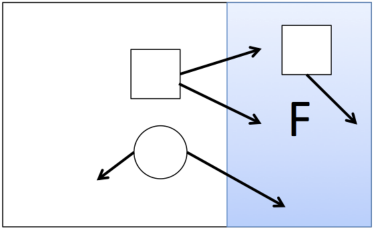
Define

$$\text{Attr}_0^i(F) := \{s \in S \mid \text{ Player 0 can force a visit from } s \text{ to } F \text{ in less than } i \text{ moves}\}$$

Force Visit in Next Step

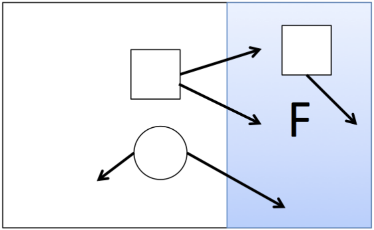
Given a set of states, compute the set of states $\text{ForceNext}_0(F)$ from which of Player 0 can force to visit F in the next step. I.e., for each state $s \in \text{ForceNext}_0(F)$ Player 0 can fix a strategy s.t. all plays starting in s visit F in the first step.

$$\begin{aligned}\text{ForceNext}_0(F) \quad = \quad & \{s \in S_0 \mid \exists s' \in S : (s, s') \in E \wedge s' \in F\} \cup \\ & \{s \in S_1 \mid \forall s' \in S : (s, s') \in E \rightarrow s' \in F\}\end{aligned}$$



Force Visit in Next Step

$$\begin{aligned}\text{ForceNext}_0(F) &= \{s \in S_0 \mid \exists s' \in S : (s, s') \in E \wedge s' \in F\} \cup \\ &\quad \{s \in S_1 \mid \forall s' \in S : (s, s') \in E \rightarrow s' \in F\}\end{aligned}$$



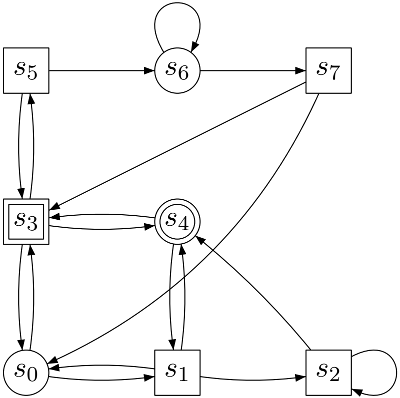
$$\text{ForceNext}_0(A \cup B) \supsetneq \text{ForceNext}_0(A) \cup \text{ForceNext}_0(B)$$

Computing the Attractor

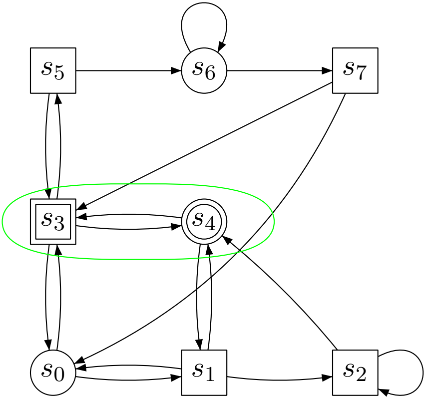
Construction of $\mathrm{Attr}_0^i(F)$:

$$\begin{aligned}\mathrm{Attr}_0^0(F) &= F \\ \mathrm{Attr}_0^{i+1}(F) &= \mathrm{Attr}_0^i(F) \cup \mathrm{ForceNext}_0(\mathrm{Attr}_0^i(F))\end{aligned}$$

Example

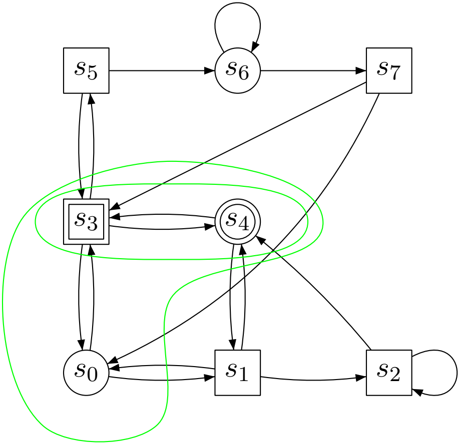


Example



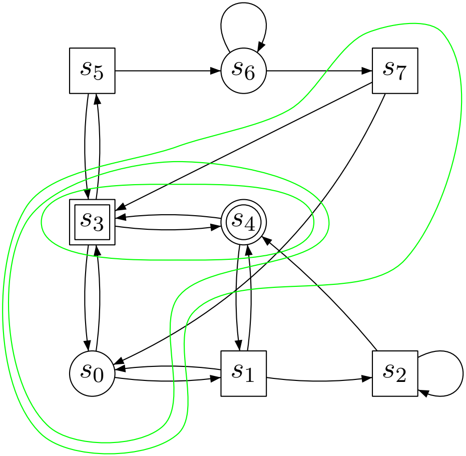
$$\mathrm{Attr}_0^0 = \{s_3, s_4\}$$

Example



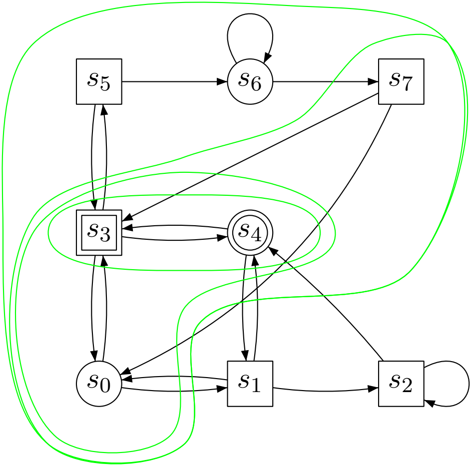
$$\mathrm{Attr}_0^0 = \{s_3, s_4\}$$
$$\mathrm{Attr}_0^1 = \{s_0, s_3, s_4\}$$

Example



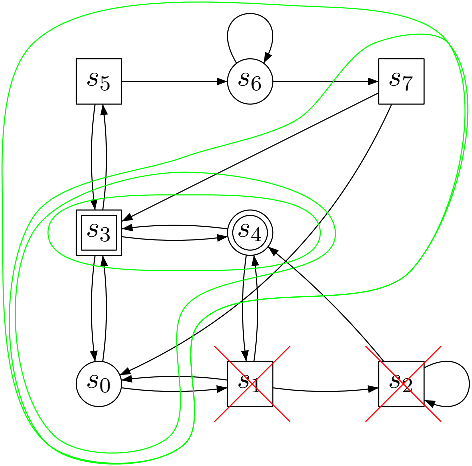
$\text{Attr}_0^0 = \{s_3, s_4\}$
$\text{Attr}_0^1 = \{s_0, s_3, s_4\}$
$\text{Attr}_0^2 = \{s_0, s_3, s_4, s_7\}$

Example



$$\mathrm{Attr}_0^0 = \{s_3, s_4\}$$
$$\mathrm{Attr}_0^1 = \{s_0, s_3, s_4\}$$
$$\mathrm{Attr}_0^2 = \{s_0, s_3, s_4, s_7\}$$
$$\mathrm{Attr}_0^3 = \{s_0, s_3, s_4, s_6, s_7\}$$
$$\mathrm{Attr}_0^4 = \{s_0, s_3, s_4, s_5, s_6, s_7\}$$

Example



$\mathrm{Attr}_0^0 = \{s_3, s_4\}$
$\mathrm{Attr}_0^1 = \{s_0, s_3, s_4\}$
$\mathrm{Attr}_0^2 = \{s_0, s_3, s_4, s_7\}$
$\mathrm{Attr}_0^3 = \{s_0, s_3, s_4, s_6, s_7\}$
$\mathrm{Attr}_0^4 = \{s_0, s_3, s_4, s_5, s_6, s_7\}$

Computing the Attractor

Construction of $\text{Attr}_0^i(F)$:

$$\begin{aligned} \text{Attr}_0^0(F) &= F \\ \text{Attr}_0^{i+1}(F) &= \text{Attr}_0^i(F) \cup \text{ForceNext}_0(\text{Attr}_0^i(F)) \end{aligned}$$

Then $\text{Attr}_0^0(F) \subseteq \text{Attr}_0^1(F) \subseteq \text{Attr}_0^2(F) \subseteq \ldots$ and since S is finite, there exists $k \leq |S|$ s.t. $\text{Attr}_0^k(F) = \text{Attr}_0^{k+1}(F)$.

The 0-Attractor is defined as:

$$\text{Attr}_0(F) := \bigcup_{i=0}^{|S|} \text{Attr}_0^i(F)$$

Computing the Attractor

Construction of $\mathrm{Attr}_0^i(F)$:

$$\begin{aligned}\mathrm{Attr}_0^0(F) &= F\\ \mathrm{Attr}_0^{i+1}(F) &= \mathrm{Attr}_0^i(F) \cup \mathrm{ForceNext}_0(\mathrm{Attr}_0^i(F))\end{aligned}$$

Then $\mathrm{Attr}_0^0(F) \subseteq \mathrm{Attr}_0^1(F) \subseteq \mathrm{Attr}_0^2(F) \subseteq \ldots$ and since S is finite, there exists $k \leq |S|$ s.t. $\mathrm{Attr}_0^k(F) = \mathrm{Attr}_0^{k+1}(F)$.

The 0-Attractor is defined as:

$$\mathrm{Attr}_0(F) := \bigcup_{i=0}^{|S|} \mathrm{Attr}_0^i(F)$$

Claim: $W_0 = \mathrm{Attr}_0(F)$ and $W_1 = S \setminus \mathrm{Attr}_0(F)$

Duality Between Players

Assume we have a partition of the state space $S = P_0 \cup P_1$ (i.e., $P_0 \cap P_1 = \emptyset$) and we want to prove $W_0 = P_0$ and $W_1 = P_1$.

We want to prove $P_0 \supseteq W_0$, $P_0 \subseteq W_0$, $P_1 \supseteq W_1$, and $P_1 \subseteq W_1$.

Since we know that $W_0 \cap W_1 = \emptyset$ holds, it is sufficient to prove $P_0 \subseteq W_0$ and $P_1 \subseteq W_1$.

$$\begin{aligned} P_0 &\subseteq W_0 \\ S \setminus P_0 &\supseteq S \setminus W_0 \\ P_1 &\supseteq S \setminus W_0 \supseteq W_1 \\ P_1 &\supseteq W_1 \end{aligned}$$

$$\begin{aligned} P_1 &\subseteq W_1 \\ S \setminus P_1 &\supseteq S \setminus W_1 \\ P_0 &\supseteq S \setminus W_1 \supseteq W_0 \\ P_0 &\supseteq W_0 \end{aligned}$$

0-Attractor

To show $W_0 = \mathrm{Attr}_0(F)$ and $W_1 = S \setminus \mathrm{Attr}_0(F)$, we construct winning strategies for Player 0 and 1.

0-Attractor

To show $W_0 = \text{Attr}_0(F)$ and $W_1 = S \setminus \text{Attr}_0(F)$, we construct winning strategies for Player 0 and 1.

Proof.

$\text{Attr}_0(F) \subseteq W_0$

We prove for every i and for every state $s \in \text{Attr}_0^i(F)$ that Player 0 has a positional winning strategy to reach F in $\leq i$ steps.

- (Base) $s \in \text{Attr}_0^0(F) = F$
- (Induction) $s \in \text{Attr}_0^{i+1}(F)$

  If $s \in \text{Attr}_0^i(F)$, then we apply induction hypothesis.

  Otherwise $s \in \text{ForceNext}_0(\text{Attr}_0^i(F)) \setminus \text{Attr}_0^i(F)$ and Player 0 can force a visit to $\text{Attr}_0^i(F)$ in one step and from there she needs at move i steps by induction hypothesis. So, F is reached after a finite number of moves.

0-Attractor cont.

Proof cont.

$S \setminus \text{Attr}_0(F) \subseteq W_1$

Assume $s \in S \setminus \text{Attr}_0(F)$, then $s \notin \text{ForceNext}_0(\text{Attr}_0(F))$ and we have two cases:

(a) $s \in S_0 \cap S \setminus \text{Attr}_0(F)\ \forall s' \in S\colon (s, s') \in E \rightarrow s' \notin \text{Attr}_0(F)$

(b) $s \in S_1 \cap S \setminus \text{Attr}_0(F)\ \exists s' \in S\colon (s, s') \in E \wedge s' \notin \text{Attr}_0(F)$

In $S \setminus \text{Attr}_0(F)$ Player 1 can choose edges according to (b) leading again to $S \setminus \text{Attr}_0(F)$ and by (a) Player 0 cannot escape from $S \setminus \text{Attr}_0(F)$. So, F will be avoided forever.

$$W_0 = \text{Attr}_0(F) \text{ and } W_1 = S \setminus \text{Attr}_0(F)$$

Safety Games

Given a safety game (G, F) with $G = (S, S_0, E)$, i.e.,

$$\phi_S = \{\rho \in S^\omega \mid \mathrm{Occ}(\rho) \subseteq F\},$$

consider the reachability game $(G, S \setminus F)$, i.e.,

$$\phi_R = \{\rho \in S^\omega \mid \mathrm{Occ}(\rho) \cap (S \setminus F) \neq \emptyset\}.$$

$$\begin{aligned} \text{Then, } S^\omega \setminus \phi_R &= \{\rho \in S^\omega \mid \mathrm{Occ}(\rho) \cap (S \setminus F) = \emptyset\} \\ &= \{\rho \in S^\omega \mid \mathrm{Occ}(\rho) \subseteq F\}. \end{aligned}$$

Player 0 has a safety objective in (G, F) $\iff$

Player 1 has a reachability objective in $(G, S \setminus F)$.

So, W_0 in the safety game (G, F) corresponds to W_1 in the reachability game $(G, S \setminus F)$.

Summary

We know how to solve reachability and safety games by positional winning strategies.

The strategies are

- Player 0: Decrease distance to F
- Player 1: Stay outside of $\mathrm{Attr}_0(F)$

In LTL, $\Diamond F$ = reachability and $\Box F$ = safety.

Next, $\Box\Diamond F$ = Büchi and $\Diamond\Box F$ = Co-Büchi.

Exercise

1. Given a reachability game (G, F) with $G = (S, S_0, E)$ and $F \subseteq Q$, give an algorithm that computes the 0-Attractor(F) in time $O(|E|)$.

Exercise

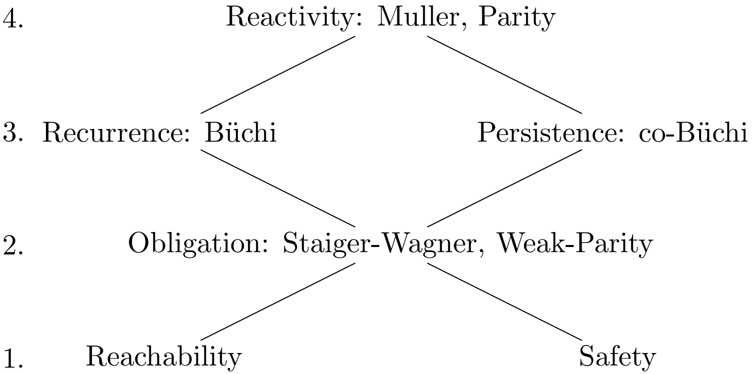
1. Given a reachability game (G, F) with $G = (S, S_0, E)$ and $F \subseteq Q$, give an algorithm that computes the 0-Attractor(F) in time $O(|E|)$.

Solution:

1. Preprocessing: Compute for every state $s \in S_1$ outdegree $\text{out}(s)$
2. Set $n(s) := \text{out}(s)$ for each $s \in S_1$
3. To breadth-first search backwards from F with the following conventions:
   - mark all $s \in F$
   - mark $s \in S_0$ if reached from marked state
   - mark $s \in S_1$ if $n(s) = 0$, other set $n(s) := n(s) - 1$.

The marked vertices are the ones of $\text{Attr}_0(F)$.

Hierarchy


4. Reactivity: Muller, Parity
3. Recurrence: Büchi
Persistence: co-Büchi
2. Obligation: Staiger-Wagner, Weak-Parity
1. Reachability
Safety

Büchi and co-Büchi Games

Büchi Game

Given a Büchi game (G, F) over the game graph $G = (S, S_0, E)$ with the set $F \subseteq S$ of Büchi states, we aim to

- determine the winning regions of Player 0 and 1
- compute their respective winning strategies

Recall, Player 0 wins ρ iff she visits infinitely often states in F, i.e., $\phi = \{\rho \in S^\omega \mid \inf(\rho) \cap F \neq \emptyset\}$.

Idea

Compute for $i \geq 1$ the set Recur_0^i of accepting states $s \in F$ from which Player 0 can force at least i revisits to F.

Then, we will show that

$$F \supseteq \text{Recur}_0^1(F) \supseteq \text{Recur}_0^2(F) \supseteq \dots$$

and we compute the winning region of Player 0 with

$$\text{Recur}_0(F) := \bigcap_{i \leq 1} Recur_0^i(F)$$

Again, since F is finite, there exists k such that $\text{Recur}_0(F) = \text{Recur}_0^k(F)$.

Claim: $W_0 = \text{Attr}_0(Recur_0(F))$

Idea

Compute for $i \geq 1$ the set Recur_0^i of accepting states $s \in F$ from which Player 0 can force at least i revisits to F.

Then, we will show that

$$F \supseteq \text{Recur}_0^1(F) \supseteq \text{Recur}_0^2(F) \supseteq \dots$$

and we compute the winning region of Player 0 with

$$\text{Recur}_0(F) := \bigcap_{i \leq 1} Recur_0^i(F)$$

Again, since F is finite, there exists k such that $\text{Recur}_0(F) = \text{Recur}_0^k(F)$.

Claim: $W_0 = \text{Attr}_0(Recur_0(F))$

First, we define Recur_0 formally using a modified version of Attractor.

One-Step Attractor

We count revisits, so we need the set of states from which Player 0 can force a revisit to F, i.e., state from which she can force a visit in ≥ 1 steps.

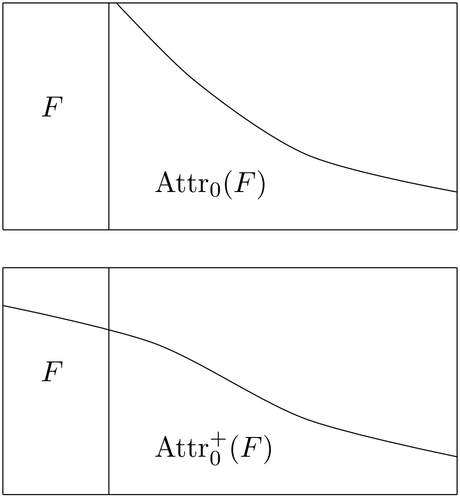
We define a slightly modified attractor:

$$\begin{aligned} A_0^0 &= \emptyset \\ A_0^{i+1} &= A_0^i \cup \text{ForceNext}_0(A_0^i \cup F) \end{aligned}$$

$$\text{Attr}_0^+(F) = \bigcup_{i \geq 0} A_0^i$$

$\text{Attr}_0^+(F)$ is the set of states from which Player 0 can force a revisit to F.

Visit versus Revisit



Recurrence Set

We define

$$\mathrm{Recur}_0^0(F) := F$$
$$\mathrm{Recur}_0^{i+1}(F) := F \cap \mathrm{Attr}_0^+(\mathrm{Recur}_0^i(F))$$
$$\mathrm{Recur}_0(F) := \bigcap_{i \geq 0} \mathrm{Recur}_0^i(F)$$

We show that there exists k such that $\mathrm{Recur}_0(F) := \bigcap_{i \geq 0}^k \mathrm{Recur}_0^i(F)$ by proving $\mathrm{Recur}_0^{i+1}(F) \subseteq \mathrm{Recur}_0^i(F)$ for all $i \geq 0$.

Proof.

- $i = 0$: $F \cap \mathrm{Attr}_0^+(F) \subseteq F$
- $i \to i + 1$:
  $\mathrm{Recur}_0^{i+1}(F) = F \cap \mathrm{Attr}_0^+(\mathrm{Recur}_0^i(F)) \subseteq F \cap \mathrm{Attr}_0^+(\mathrm{Recur}_0^{i-1}(F))$
  $= \mathrm{Recur}_0^i(F)$ since (i) $\mathrm{Recur}_0^i(F) \subseteq \mathrm{Recur}_0^{i-1}(F)$ by ind. hyp. and (ii) Attr_0^+ is monotone.

Recurrence Set cont.

We show that all states in $\text{Attr}_0(\text{Recur}_0(F))$ are winning for Player 0, i.e., $\text{Attr}_0(\text{Recur}_0(F)) \subseteq W_0$. We construct a memoryless winning strategy for Player 0 for all states in $\text{Attr}_0(\text{Recur}_0(F))$.

Proof.

We know that there exists k such that
$\text{Recur}_0^{k+1}(F) = \text{Recur}_0^k(F) = F \cap \text{Attr}_0^+(\text{Recur}_0^k(F))$. So,

- for $s \in \text{Recur}_0^k(F) \cap S_0$ Player 0 can choose an edge back to $\text{Attr}_0^+(\text{Recur}_0^k(F))$ and
- for $s \in \text{Recur}_0^k(F) \cap S_1$ all edges lead back to $\text{Attr}_0^+(\text{Recur}_0^k(F))$.

For all states in $\text{Attr}_0(\text{Recur}_0(F)) \setminus \text{Recur}_0(F)$, Player 0 can follow the attractor strategy to reach $\text{Recur}_0(F)$.

□

Recurrence Set cont.

We show $S \setminus \mathrm{Attr}_0(\mathrm{Recur}_0(F)) \subseteq W_1$.

Proof.

Show: Player 1 can force $\leq i$ visits to F from $s \notin \mathrm{Attr}_0(\mathrm{Recur}_0^i(F))$

$i = 0$: $s \notin \mathrm{Attr}_0(F)$, so Player 1 can avoid visiting F at all.

$i \to i+1$: $s \notin \mathrm{Attr}_0(\mathrm{Recur}_0^{i+1}(F))$.

- $s \notin \mathrm{Attr}_0(\mathrm{Recur}_0^i(F))$, Player 1 plays according to ind. hypothese
- Otherwise, $s \in \mathrm{Attr}_0(\mathrm{Recur}_0^i(F)) \setminus \mathrm{Attr}_0(\mathrm{Recur}_0^{i+1}(F))$ and Player 1 can avoid $\mathrm{Attr}_0(\mathrm{Recur}_0^{i+1}(F))$.

□

Büchi games

We have shown that Player 0 has a (memoryless) winning strategy from every state in $\mathrm{Attr}_0(\mathrm{Recur}_0(F))$, so $\mathrm{Attr}_0(\mathrm{Recur}_0(F)) \subseteq W_0$. And, Player 1 has a (memoryless) winning strategy from every state in $S \setminus \mathrm{Attr}_0(\mathrm{Recur}_0(F))$, so $S \setminus \mathrm{Attr}_0(\mathrm{Recur}_0(F)) \subseteq W_1$. This implies the following theorem.

Theorem

Given a Büchi game $((S, S_0, E), F)$, the winning regions W_0 and W_1 are computable and form a partition, i.e., $W_0 \cup W_1 = S$. Both players have memoryless winning strategies.

Co-Büchi Games

Given a Co-Büchi Game $((S, S_0, E), F)$, i.e.,

$$\phi_C = \{\rho \in S^\omega \mid \mathrm{Inf}(\rho) \subseteq F\}$$

consider the Büchi Game $((S, S_0, E), S \setminus F)$, i.e,

$$\phi_B = \{\rho \in S^\omega \mid \mathrm{Inf}(\rho) \cap (S \setminus F) \neq \emptyset\}.$$

$$\begin{aligned} \text{Then, } S^\omega \setminus \phi_B &= \{\rho \in S^\omega \mid \mathrm{Inf}(\rho) \cap (S \setminus F) = \emptyset\} \\ &= \{\rho \in S^\omega \mid \mathrm{Inf}(\rho) \subseteq F\}. \end{aligned}$$

Player 0 has a co-Büchi objective in (G, F) $\iff$

Player 1 has a Büchi objective in $(G, S \setminus F)$.

So, W_0 in the co-Büchi game (G, F) corresponds to W_1 in the Büchi game $(G, S \setminus F)$.

Summary

We know how to solve Büchi and Co-Büchi games by positional winning strategies.

In LTL,

- $\Diamond F$ = reachability
- $\Box F$ = safety
- $\Box\Diamond F$ = Büchi
- $\Diamond\Box F$ = Co-Büchi

Exercise

2. Consider the game graph shown in below and the following winning conditions:
   (a) $\mathrm{Occ}(\rho) \cap \{1\} \neq \emptyset$ and
   (b) $\mathrm{Occ}(\rho) \subseteq \{1, 2, 3, 4, 5, 6\}$ and
   (c) $\mathrm{Inf}(\rho) \cap \{4, 5\} \neq \emptyset$.

Compute the winning regions and corresponding winning strategies showing the intermediate steps (i.e., the Attractor and Recurrence sets) of the computation.