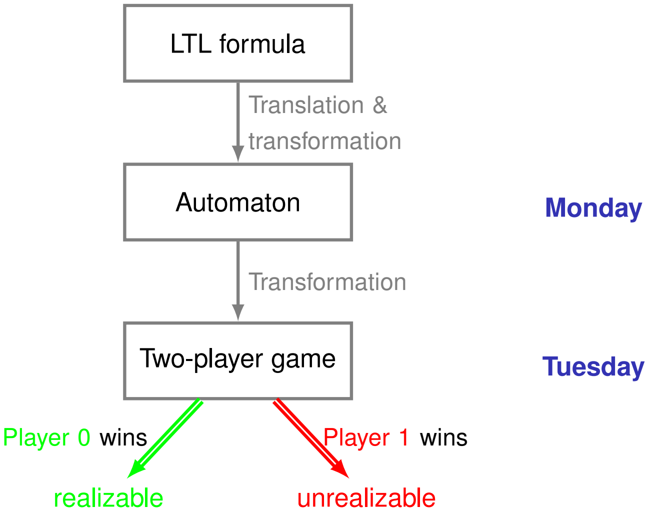
Synthesis Algorithms

Rayna Dimitrova
University of Leicester

Midlands Graduate School 2019

Acknowledgement: Many slides courtesy of Bernd Finkbeiner.

Synthesis from LTL specifications



LTL Realizability

Given φ, does there exist a strategy f that satisfies φ?

Approach: Construct a game between environment (providing input), and system (choosing output). Check if system has a winning strategy.

Attempt: Let $\mathcal{A}_\varphi = (\Sigma, Q, Q_0, \delta, F)$ be an NBA for φ

- System chooses output value $o \in \Sigma_O$
- Environment chooses output value $i \in \Sigma_I$
- Round: system and environment set their variables
- Play: infinite word in Σ^ω
- System wins if infinite play accepted by $\mathcal{A}_\varphi$

Problem: In a nondeterministic automaton, an accepted word can also have rejecting runs. Mismatch between nondeterminism and strategic choice: the system can lose just because the wrong run was chosen.

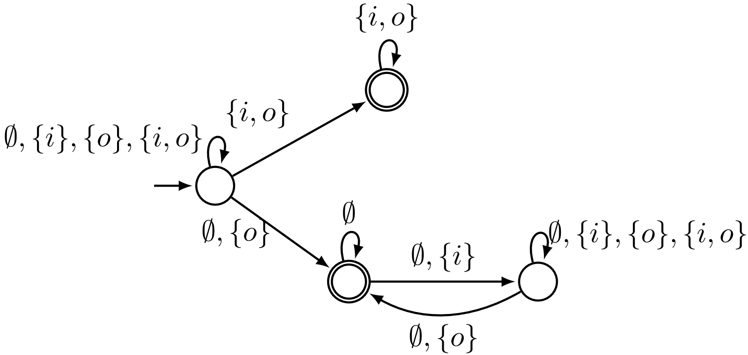
Why are nondeterministc automata not suitable?

Nondeterministic automata have perfect foresight.

Strategies have no foresight.

Example: $(\mathsf{FG}(i \wedge \mathsf{X}o)) \vee (\mathsf{GF}(\neg i \wedge \mathsf{X}\neg o))$

- System has winning strategy (copy input to output).
- The system cannot choose between the two disjuncts.



Deterministic Büchi automata (DBA)

A NBA $\mathcal{A}$ is a **DBA** iff

$$|Q_0| \leq 1 \quad \text{and} \quad |\delta(q,\sigma)| \leq 1 \quad \text{for all } q \in Q \text{ and } \sigma \in \Sigma$$

A DBA $\mathcal{A}$ is **complete** iff

$$|Q_0| = 1 \quad \text{and} \quad |\delta(q,\sigma)| = 1 \quad \text{for all } q \in Q \text{ and } \sigma \in \Sigma$$

Complete DBAs have a **unique** run for every input word.

Not every LTL formula can be translated into an equivalent DBA.

Acceptance conditions: Parity

A **parity condition** is a coloring function $\alpha : Q \to \mathbb{N}$.

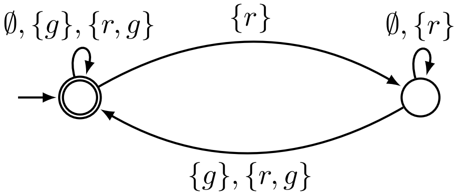
An infinite sequence $q_0 q_1 q_2 \ldots \in Q^\omega$ is max-parity-accepted iff the highest color k that appears infinitely often is **even**.

Acceptance conditions: Parity

A **parity condition** is a coloring function $\alpha : Q \to \mathbb{N}$.

An infinite sequence $q_0 q_1 q_2 \ldots \in Q^\omega$ is max-parity-accepted iff the highest color k that appears infinitely often is **even**.

DBA:

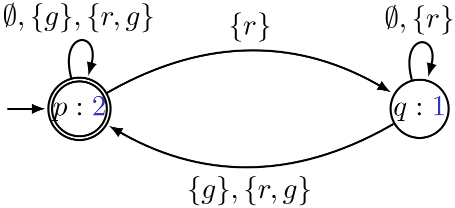


Acceptance conditions: Parity

A **parity condition** is a coloring function $\alpha : Q \to \mathbb{N}$.

An infinite sequence $q_0 q_1 q_2 \ldots \in Q^\omega$ is max-parity-accepted iff the highest color k that appears infinitely often is **even**.

DPA:



Deterministic parity automata (DPA)

Theorem [McNaughton 1966]

For each NBA there is an equivalent deterministic ω-automaton.

Theorem [Piterman'07]

For every NBA $\mathcal{N}$ with n states there exists a DPA $\mathcal{D}$ with $2n^n n!$ states and $2n$ colors such that $\mathcal{L}_\omega(\mathcal{D}) = \mathcal{L}_\omega(\mathcal{N})$.

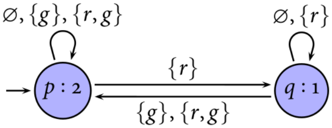
The exponential blow-up is unavoidable.

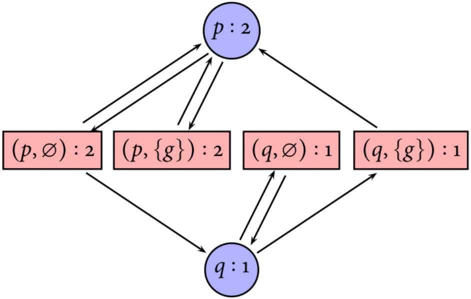
From deterministic ω-automata to games

A **game arena** $\mathcal{A} = (V, V_0, V_1, E)$ consists of

- a finite set of states V,
- a subset $V_0 \subseteq V$ of states owned by Player 0 (circles),
- a subset $V_1 = V \setminus V_0$ of states owned by Player 1 (boxes),
- an edge relation $E \subseteq V \times V$, such that every $v \in V$ has at least one outgoing edge $(v, v') \in E$. E represents the possible moves.

Player 0 = system, Player 1 = environment





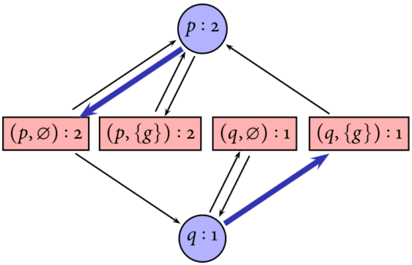
Plays and strategies in a graph game

A **play** is an infinite path through $\mathcal{A}$.

A **strategy** for player i is a function $f_i : V^* \cdot V_i \to V$ such that $(v, v') \in E$ whenever $f(\pi \cdot v) = v'$.

A play $\pi = v_0, v_1, \ldots$ **conforms to** strategy f_i of player i if $v_{n+1} = f_i(v_0, \ldots, v_n)$ whenever $v_n \in V_i$.

Strategy for Player 0



Winning conditions

- A **parity game** $\mathcal{G} = (\mathcal{A}, \alpha)$ consists of an arena $\mathcal{A}$ and a coloring function $\alpha : V \to \mathbb{N}$. Player 0 wins play π if the highest color that is seen infinitely often is even, otherwise Player 1 wins.
- A **Büchi game** $\mathcal{G} = (\mathcal{A}, F)$ consists of an arena $\mathcal{A}$ and a set $F \subseteq V$. Player 0 wins a play π if π visits F infinitely often, otherwise Player 1 wins.
- A **reachability game** $\mathcal{G} = (\mathcal{A}, R)$ consists of a game arena $\mathcal{A}$ and a set of states $R \subseteq V$. Player 0 wins a play π if π visits R at least once, otherwise Player 1 wins.

Winning regions

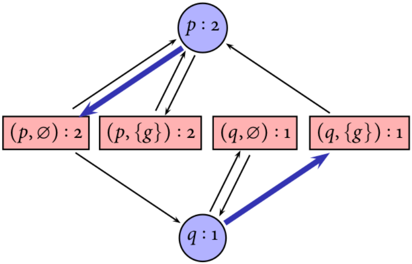
A strategy f_i is **winning** for player i from some state v if all plays that conform to f_i and that start in v are won by Player i.

The **winning region** W_i for player i is the set of states from which Player i has a winning strategy.

A game is **determined** if $V = W_0 \cup W_1$.

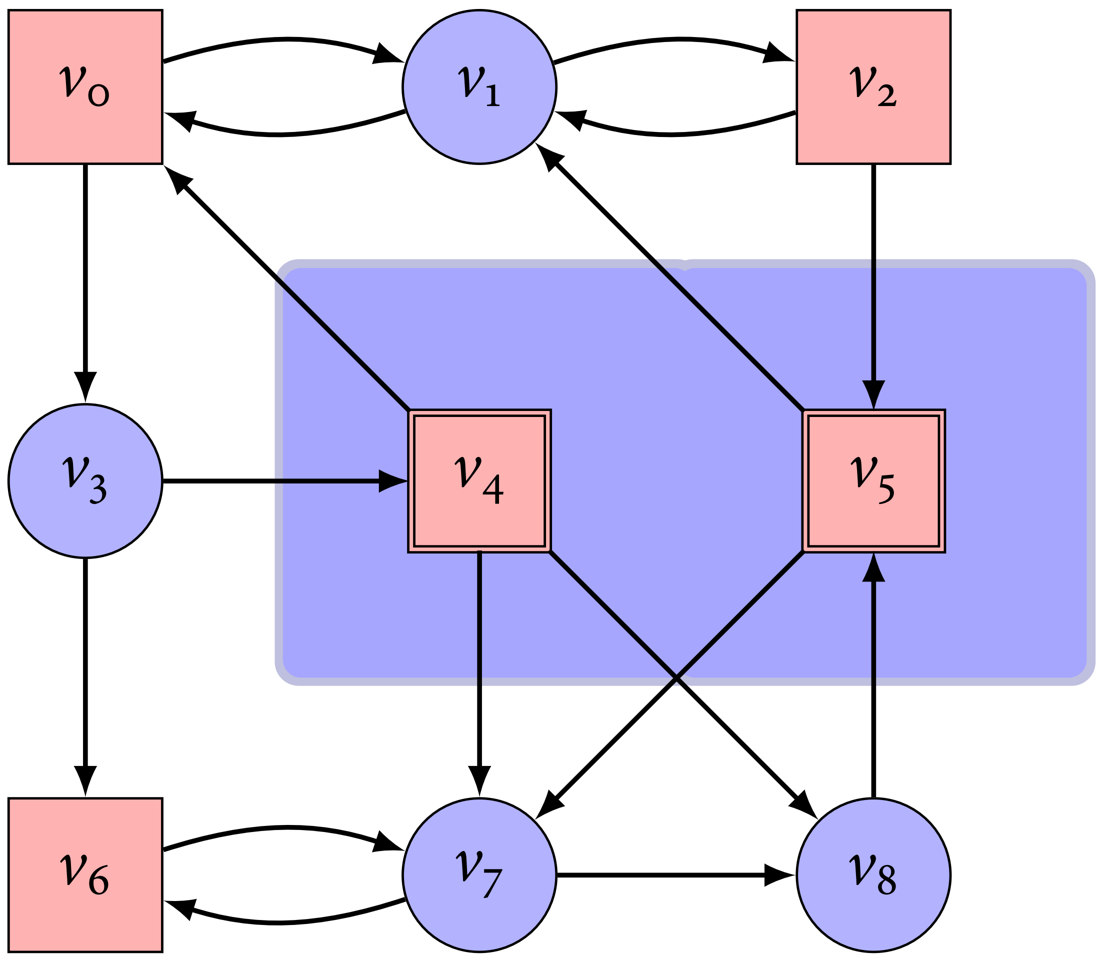
Solving a game means to determine the winning region and the wining strategies.

Winning strategy for Player 0



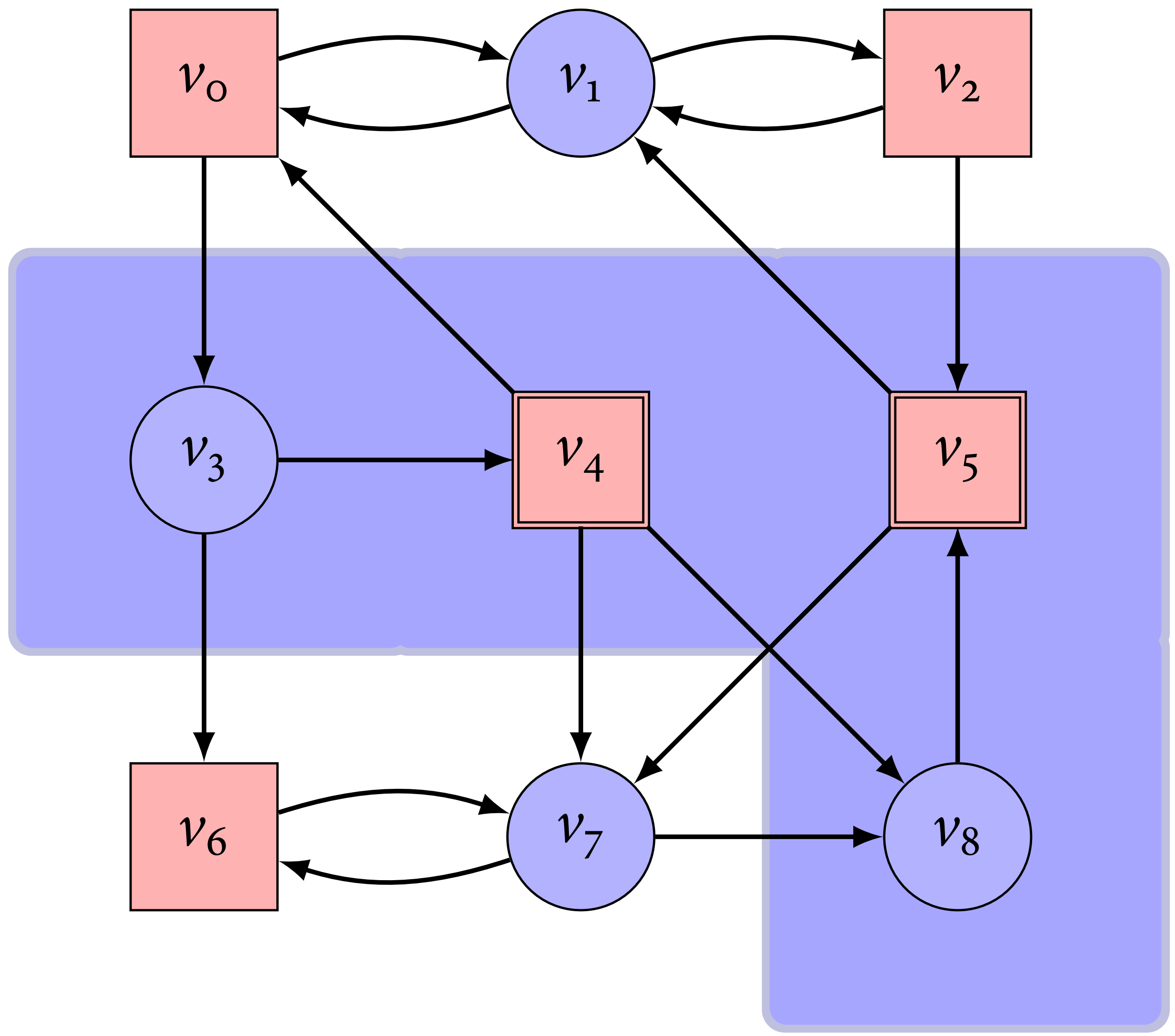
Reachability games

Reachability game: Player 0 wins a play π if π visits R at least once



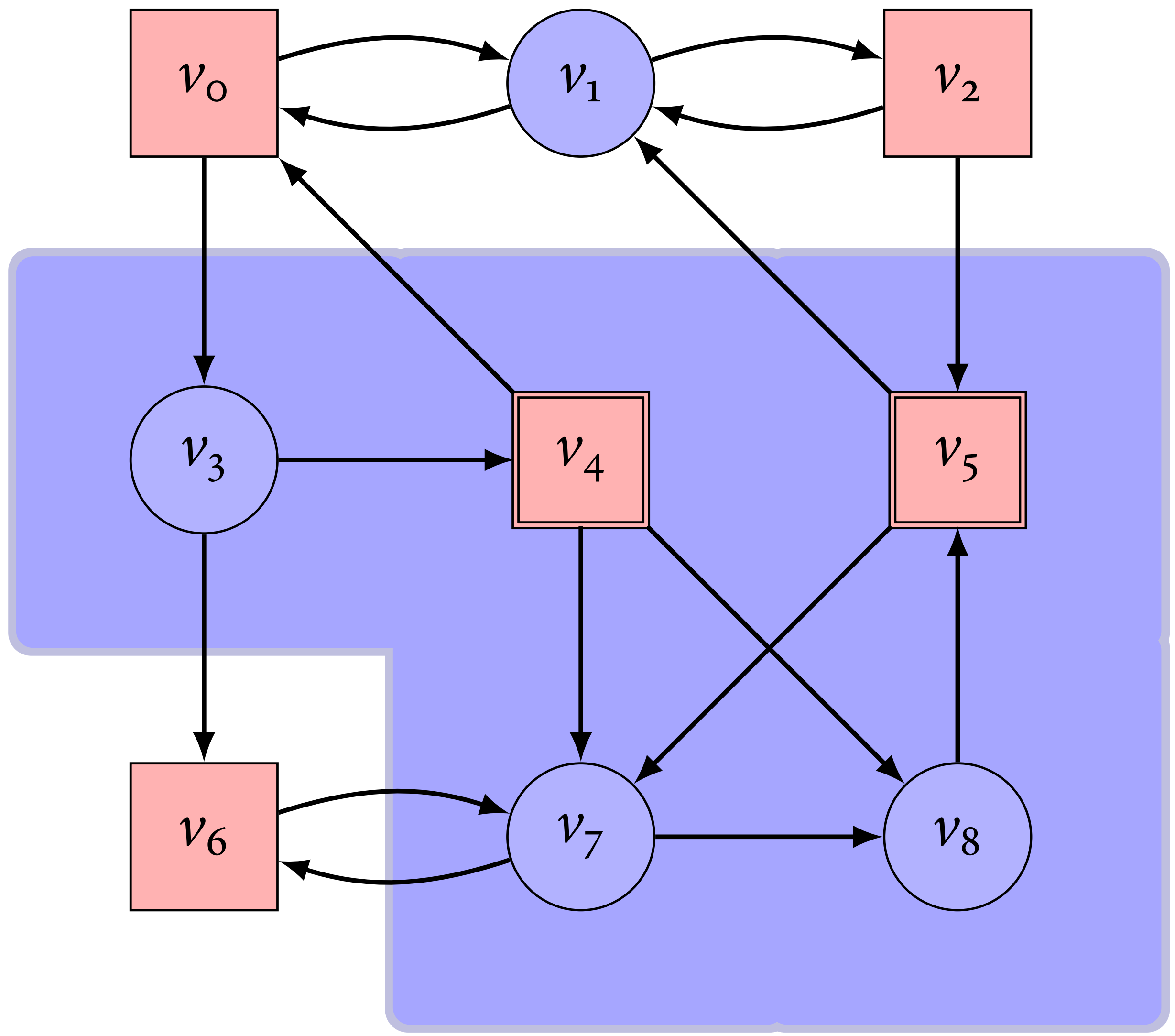
Reachability games

Reachability game: Player 0 wins a play π if π visits R at least once



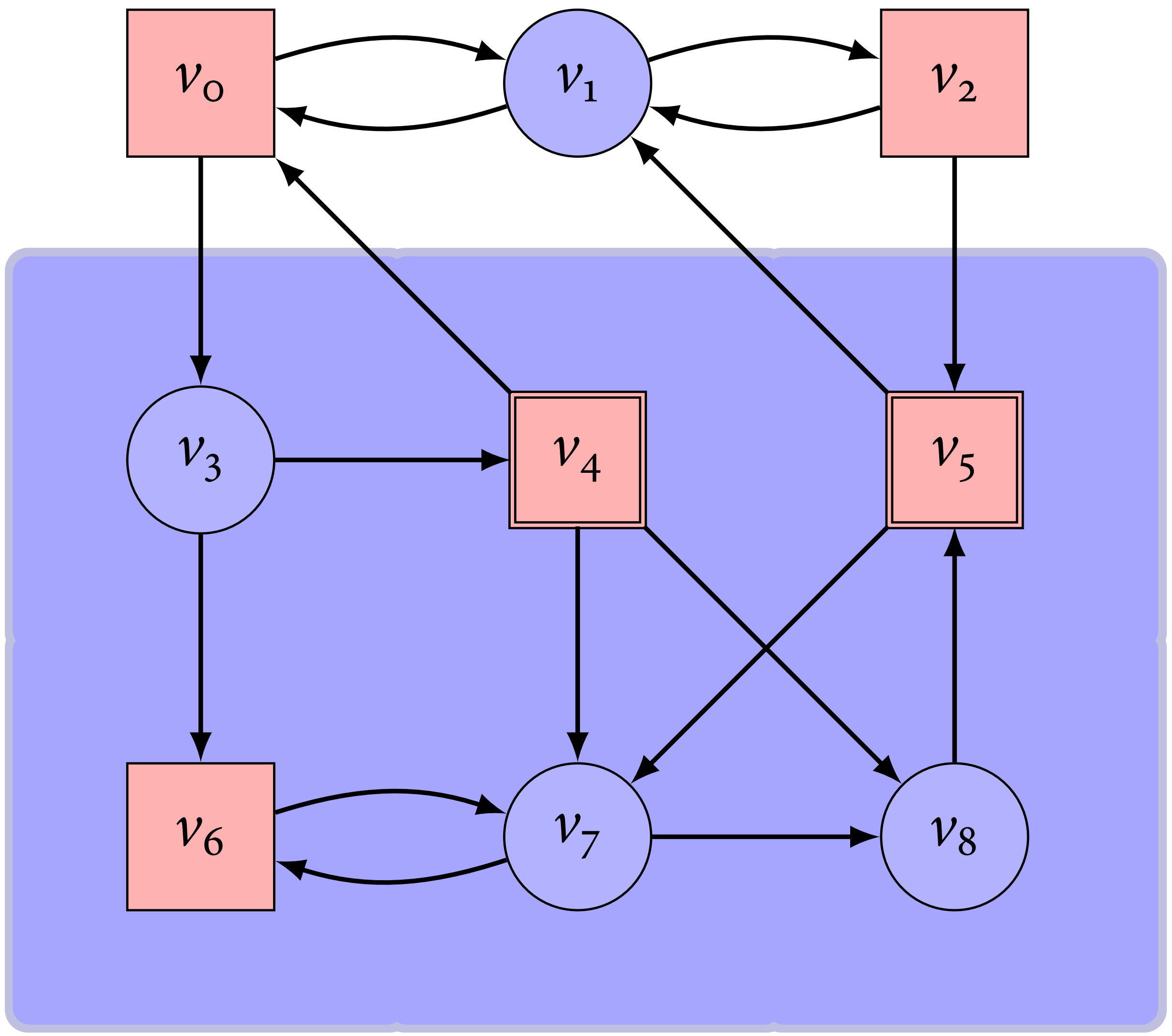
Reachability games

Reachability game: Player 0 wins a play π if π visits R at least once



Reachability games

Reachability game: Player 0 wins a play π if π visits R at least once



Attractor construction

$$
\begin{aligned}
Attr_i^0(R) &= R \\
Attr_i^{n+1}(R) &= Attr_i^n(R) \cup CPre_i(Attr_i^n(R))
\end{aligned}
$$

$$
Attr_i(R) = \bigcup_{n \in \mathbb{N}} Attr_i^n(R)
$$

where

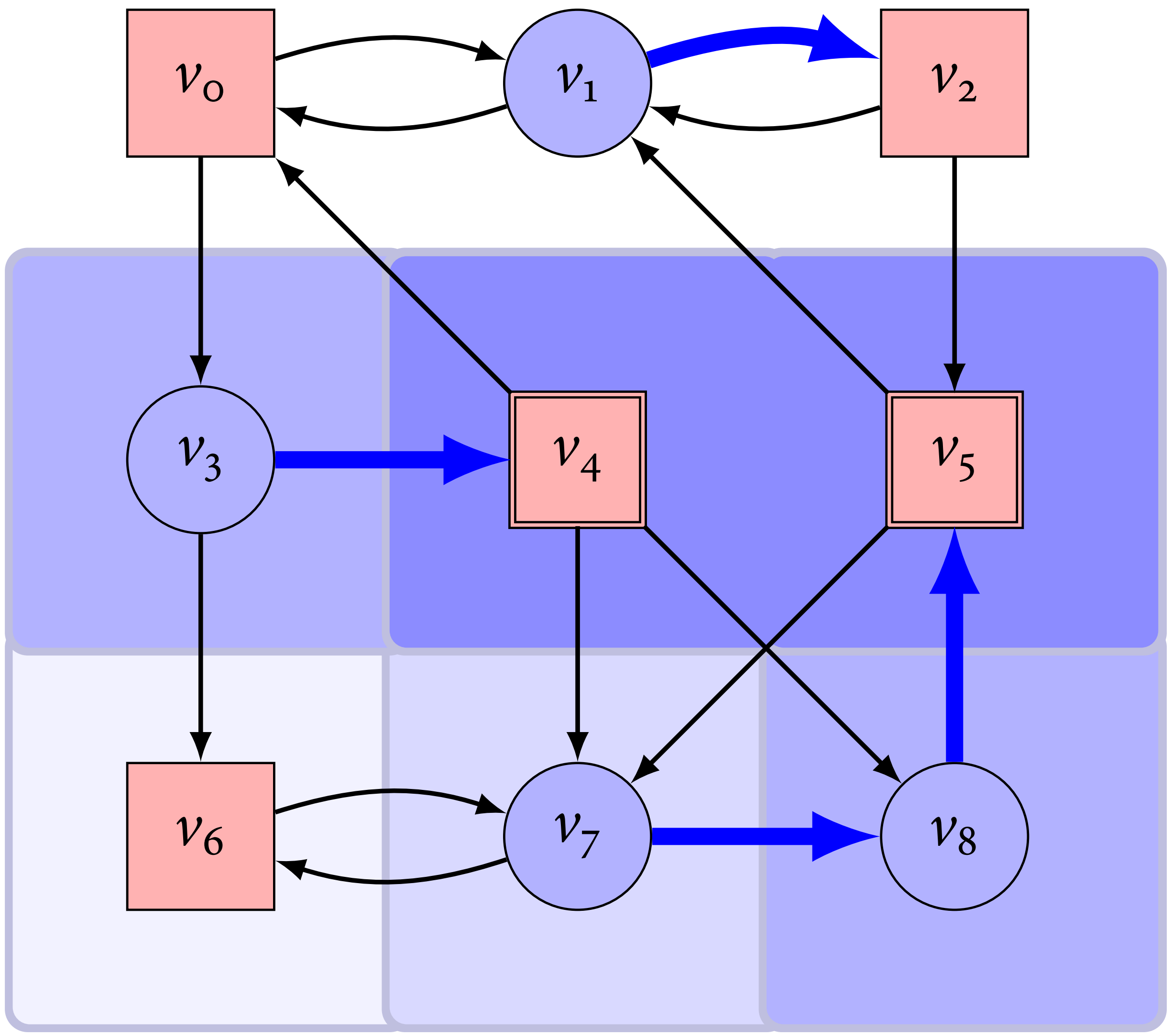
$$
\begin{aligned}
CPre_i(R) = \; & \{ v \in V_i \mid \exists v' \in V.\, (v, v') \in E \wedge v' \in R \} \\
& \cup \{ v \in V_{1-i} \mid \forall v' \in V.\, (v, v') \in E \Rightarrow v' \in R \}
\end{aligned}
$$

Winning regions of a reachability game

Winning region of Player 0: $W_0(\mathcal{G}) = Attr_0(R)$

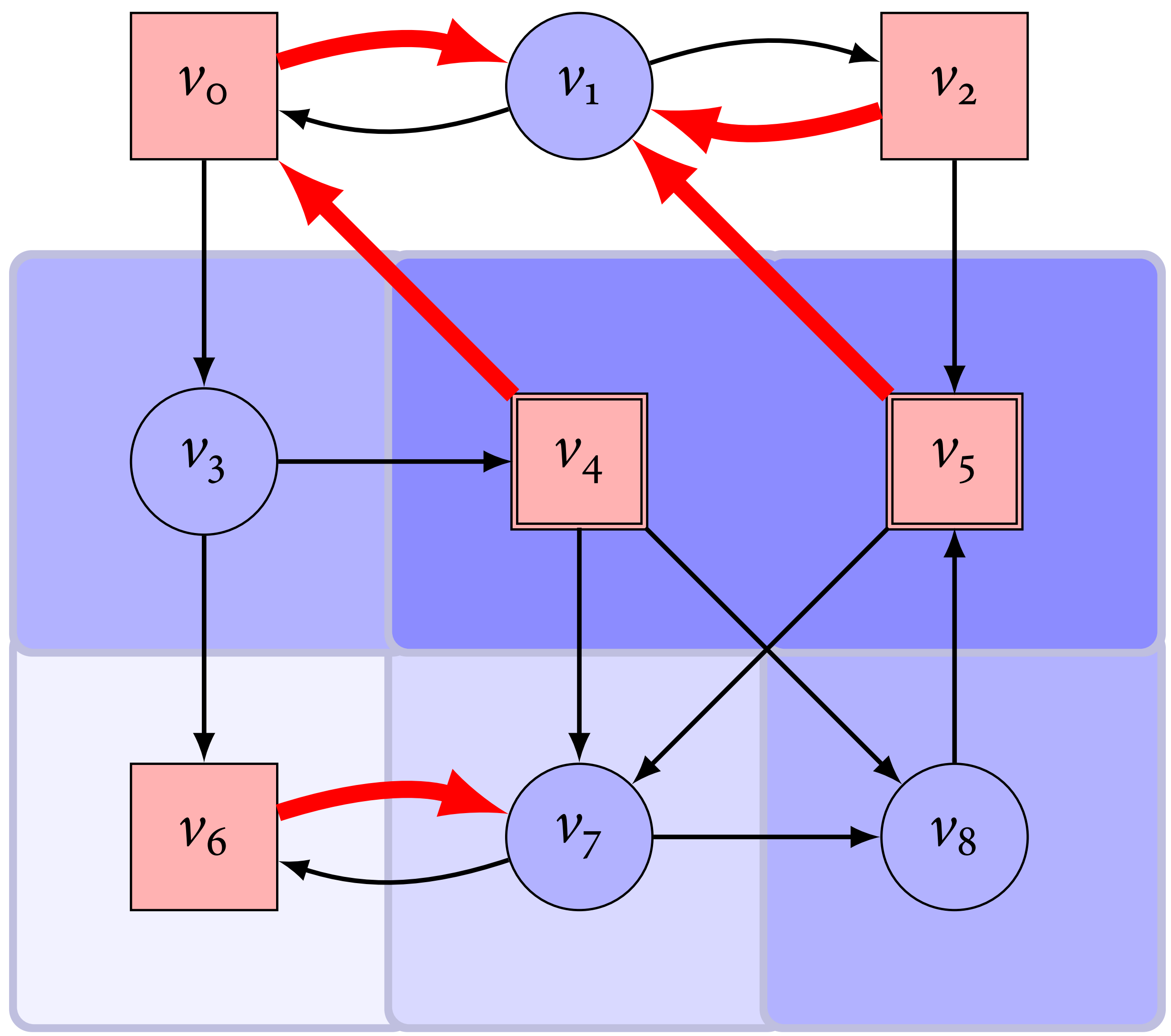
Winning region of Player 1: $W_1(\mathcal{G}) = V \setminus Attr_0(R)$

Winning strategy of Player o (Attractor Strategy)



The winning strategy for Player o always moves to $Attr_{o}^{n}(R)$ for the smallest possible n.

Winning strategy of Player 1 (Safety Strategy)

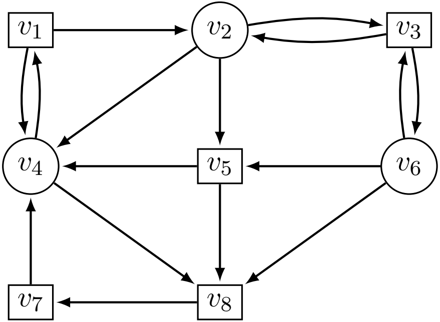


The winning strategy for Player 1 always avoids $Attr_0(R)$.

Exercise

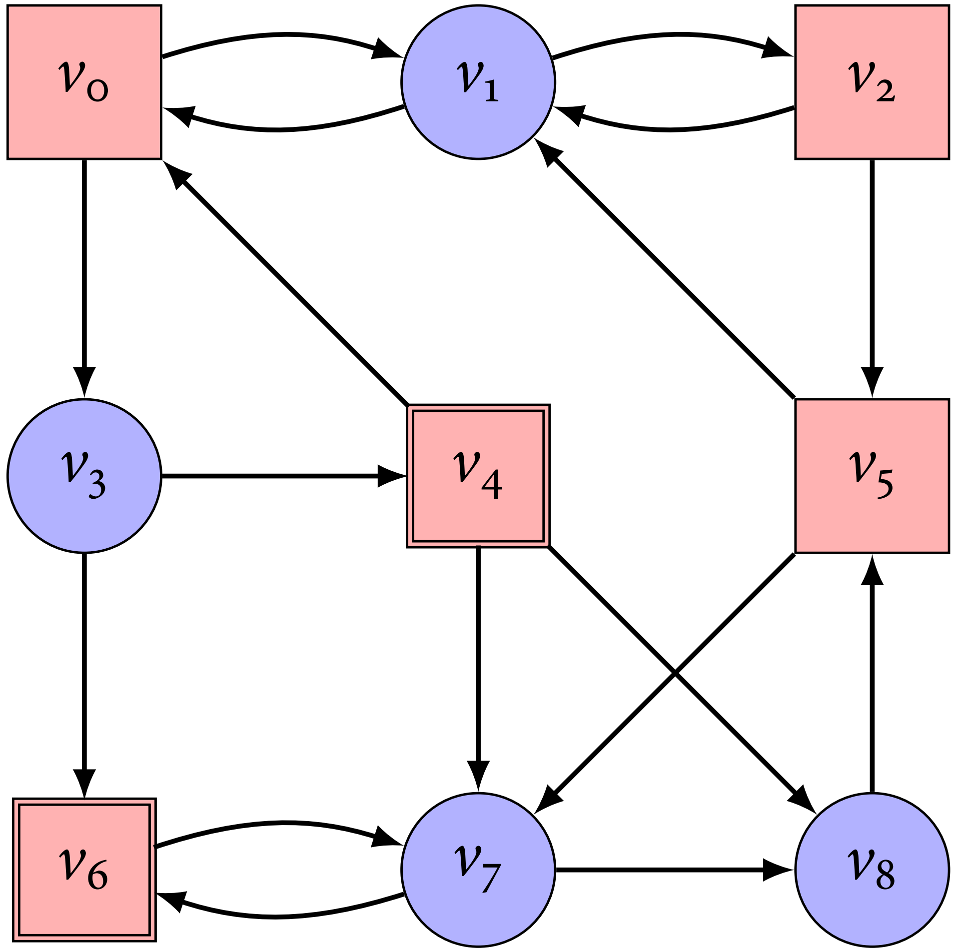
Consider the game arena given below and the winning condition defined as follows: Player 0 wins a play π if π visits the state v_4 at least once <u>and</u> π never leaves the set of nodes $\{v_1, v_2, v_3, v_4, v_5, v_6\}$.

Describe how you can compute the winning region for Player 0.



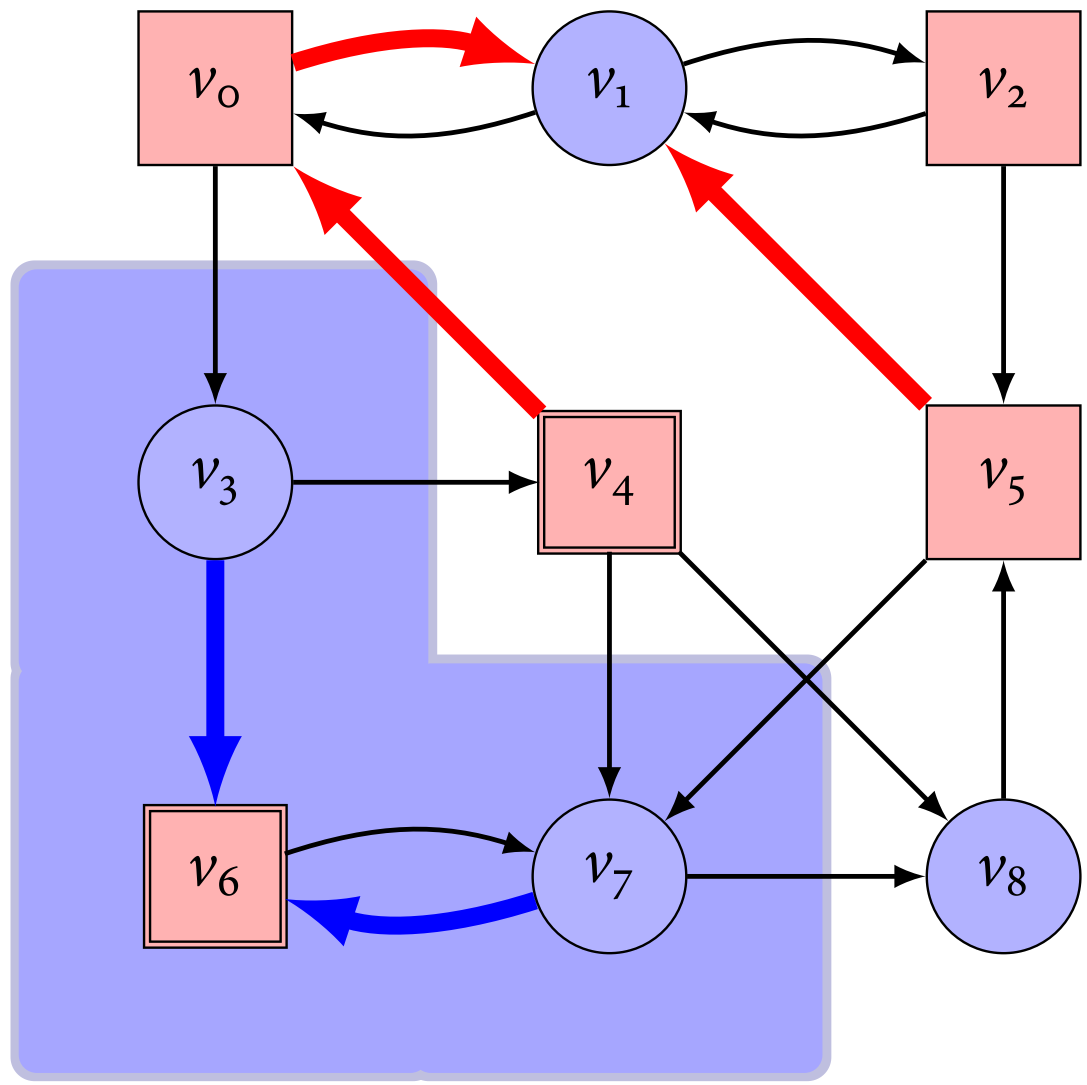
Büchi games

Büchi game: Player 0 wins a play π if π visits F infinitely often, otherwise Player 1 wins.



Büchi games

Büchi game: Player 0 wins a play π if π visits F infinitely often, otherwise Player 1 wins.



Recurrence construction

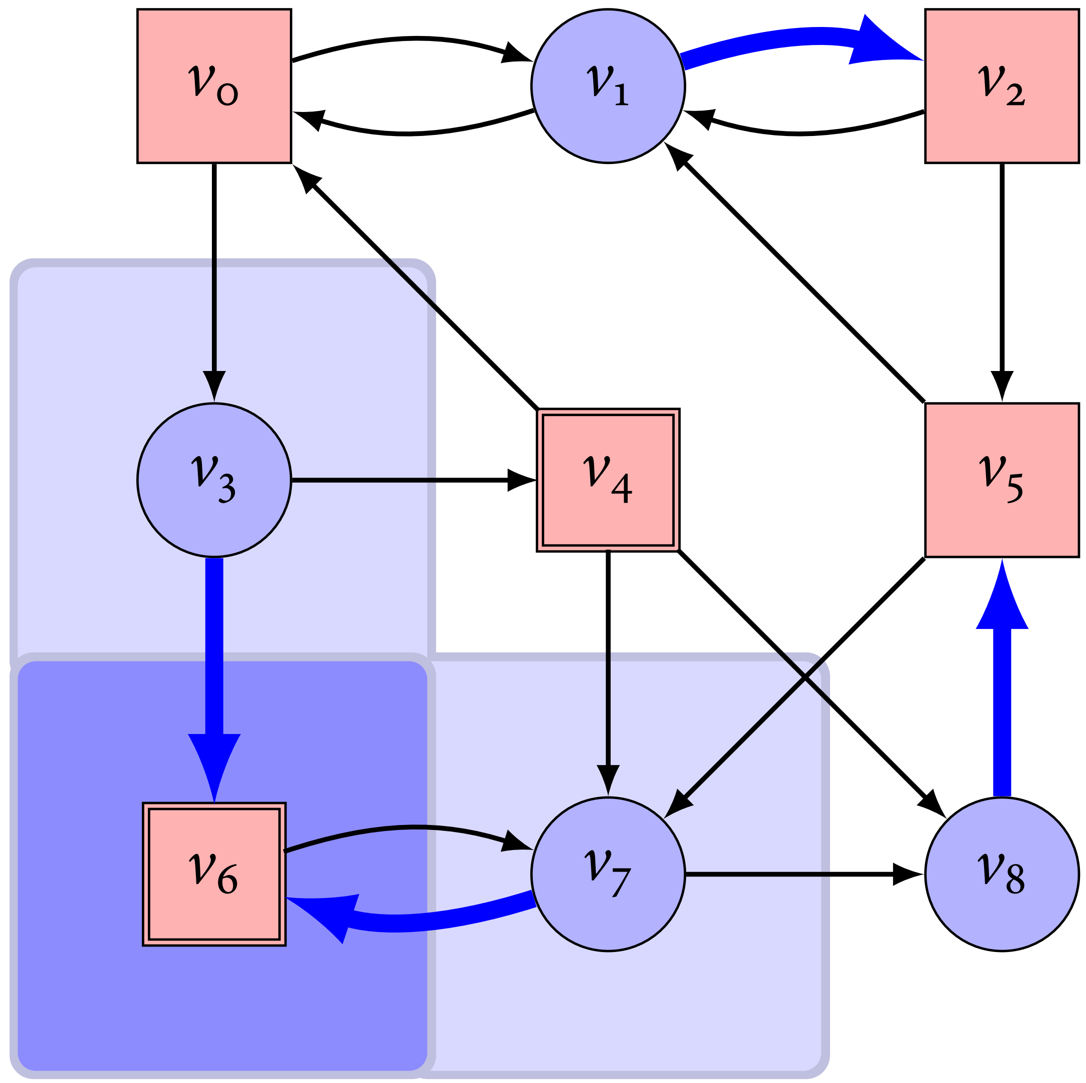
$$
\begin{aligned}
Recur^0(F) &= F \\
W_1^n(F) &= V \setminus Attr_0(Recur^n(F)) \\
Recur^{n+1}(F) &= Recur^n(F) \setminus CPre_1(W_1^n(F)) \\
\\
Recur(F) &= \bigcap_{n \in \mathbb{N}} Recur^n(F)
\end{aligned}
$$

Winning regions of a Büchi game

Winning region of Player 0: $W_0(\mathcal{G}) = Attr_0(Recur(F))$

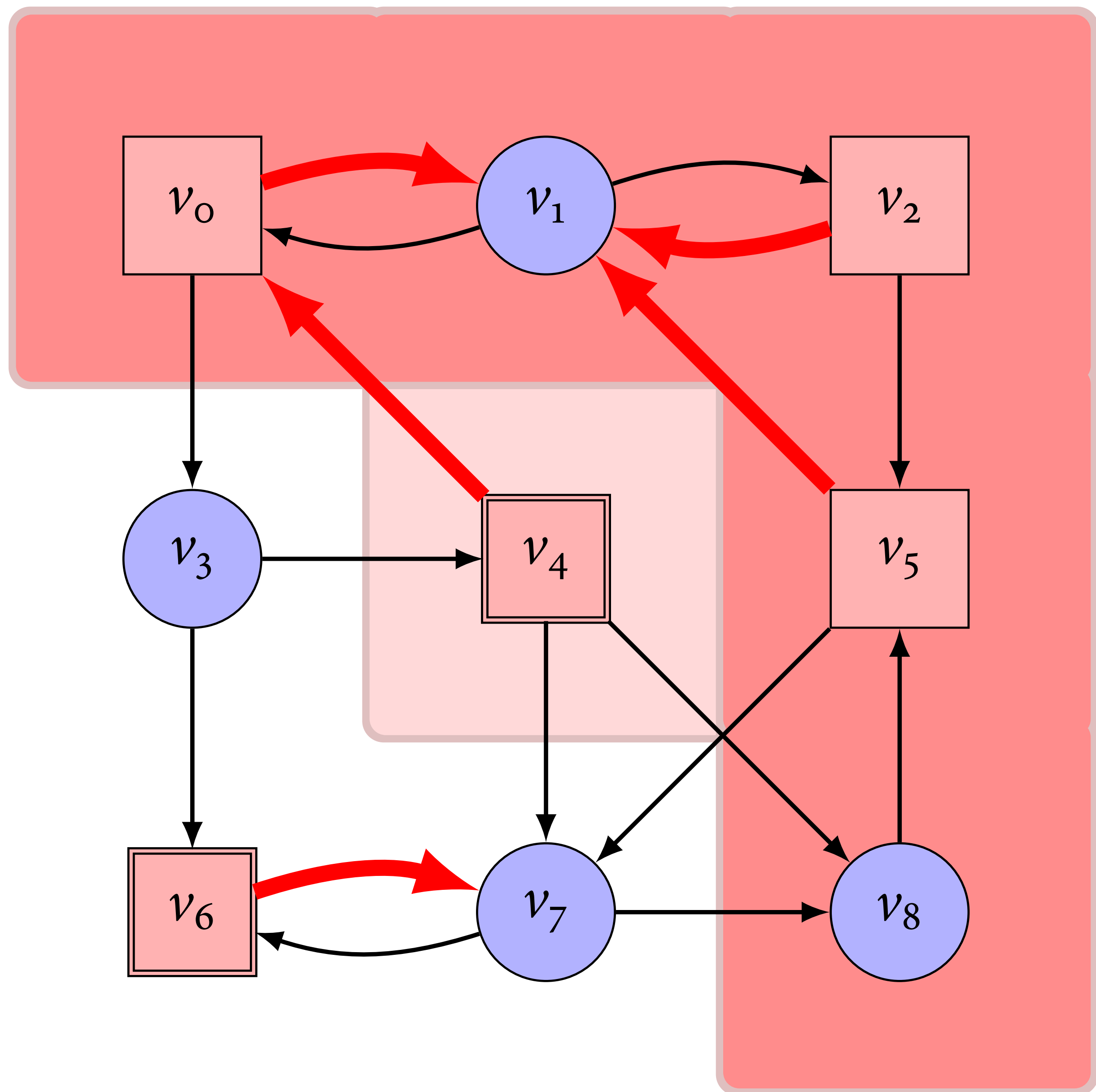
Winning region of Player 1: $W_1(\mathcal{G}) = V \setminus Attr_0(Recur(F))$

Winning strategy of Player 0 (Büchi Strategy)



The winning strategy for Player 0 always moves to $Attr_0^n(Recur(F))$ for the smallest possible n.

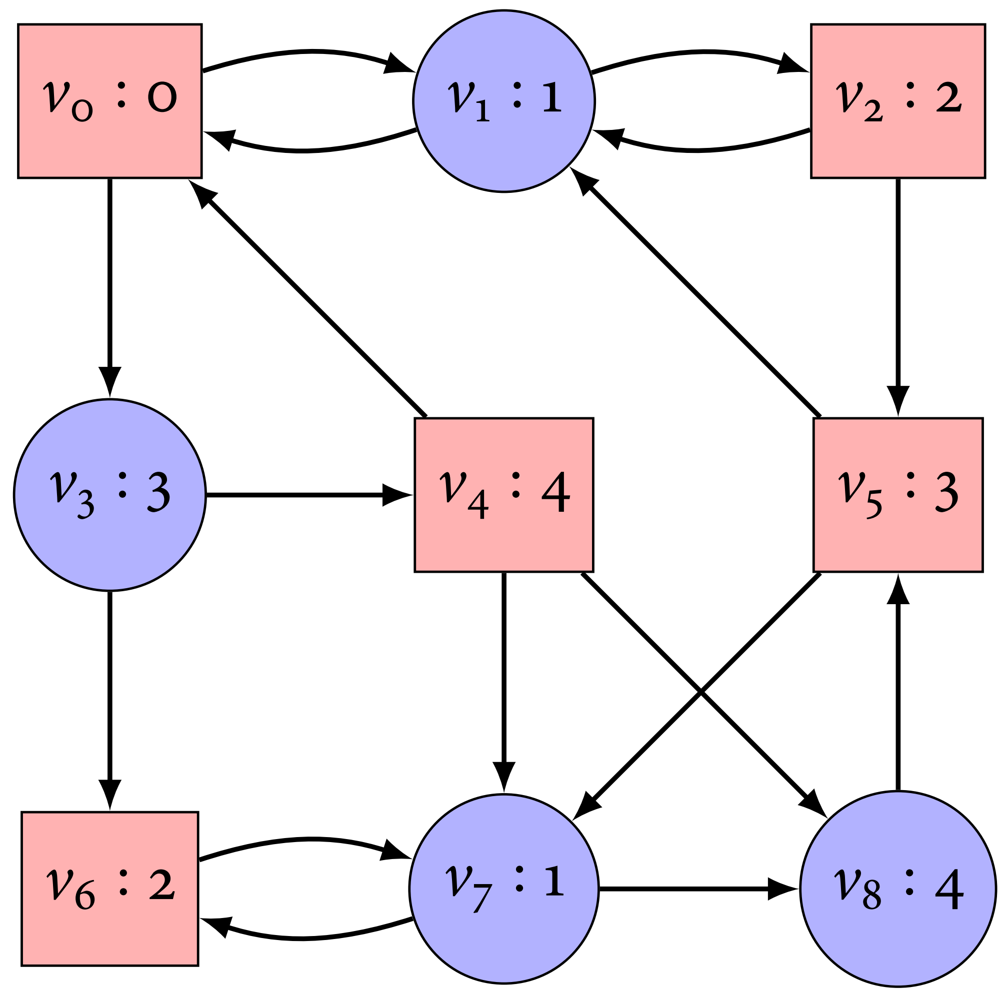
Winning strategy of Player 1 (co-Büchi Strategy)



The winning strategy for Player 1 moves from a state in W_1^n to W_1^{n-1} whenever possible, and stays in W_1^n otherwise.

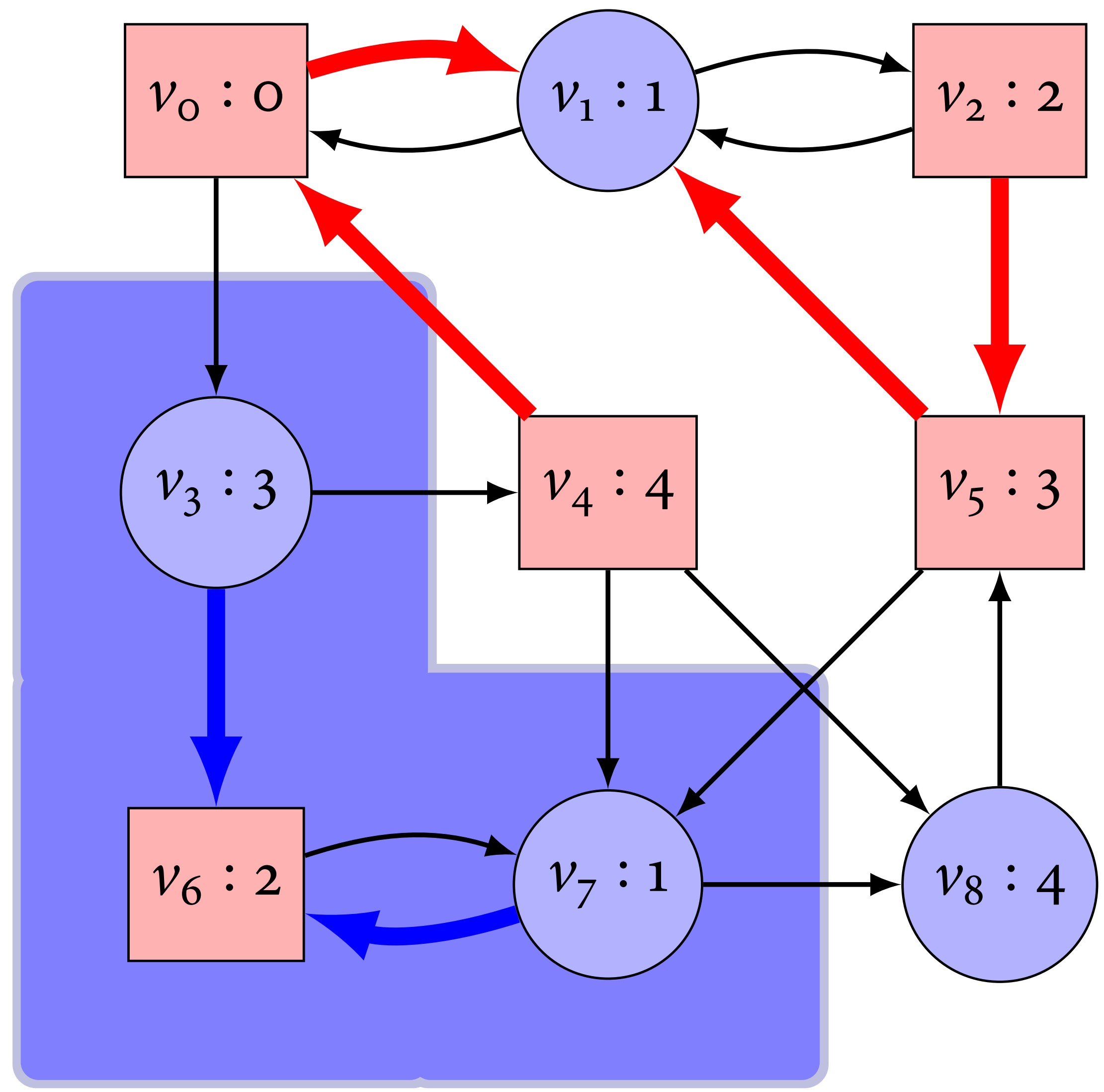
Parity games

Parity game: Player 0 wins a play π if the highest color that is seen infinitely often is even.



Parity games

Parity game: Player 0 wins a play π if the highest color that is seen infinitely often is even.



McNaughton's Algorithm: Solving parity games

$McNaughton(\mathcal{G})$

1. $c :=$ highest color in $\mathcal{G}$
2. if $c = 0$ or $V = \emptyset$
   then return $(V, \emptyset)$
3. set i to $c \mod 2$
4. set W_{1-i} to $\emptyset$
5. repeat

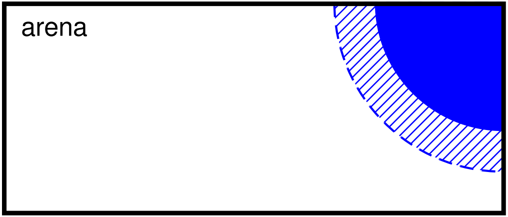
   5.1 $\mathcal{G}' := \mathcal{G} \setminus Attr_i(\alpha^{-1}(c), \mathcal{G})$
   5.2 $(W'_0, W'_1) := McNaughton(\mathcal{G}')$
   5.3 if $(W'_{1-i} = \emptyset)$ then

   5.3.1 $W_i := V \setminus W_{1-i}$
   5.3.2 return (W_0, W_1)

   5.4 $W_{1-i} := W_{1-i} \cup Attr_{(1-i)}(W'_{1-i}, \mathcal{G})$
   5.5 $\mathcal{G} := \mathcal{G} \setminus Attr_{(1-i)}(W'_{1-i}, \mathcal{G})$

McNaughton's Algorithm: Solving parity games

$McNaughton(\mathcal{G})$



1. $c :=$ highest color in $\mathcal{G}$
2. if $c = 0$ or $V = \emptyset$
   then return $(V, \emptyset)$
3. set i to $c \mod 2$
4. set W_{1-i} to $\emptyset$
5. repeat

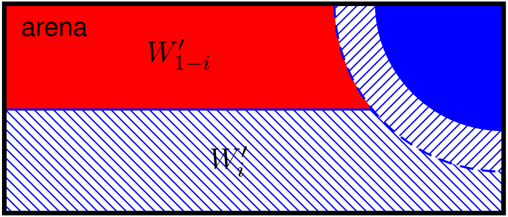
   5.1 $\mathcal{G}' := \mathcal{G} \setminus Attr_i(\alpha^{-1}(c), \mathcal{G})$
   5.2 $(W_0', W_1') := McNaughton(\mathcal{G}')$
   5.3 if $(W_{1-i}' = \emptyset)$ then

      5.3.1 $W_i := V \setminus W_{1-i}$
      5.3.2 return (W_0, W_1)

   5.4 $W_{1-i} := W_{1-i} \cup Attr_{(1-i)}(W_{1-i}', \mathcal{G})$
   5.5 $\mathcal{G} := \mathcal{G} \setminus Attr_{(1-i)}(W_{1-i}', \mathcal{G})$

McNaughton's Algorithm: Solving parity games

$McNaughton(\mathcal{G})$



1. $c :=$ highest color in $\mathcal{G}$
2. if $c = 0$ or $V = \emptyset$
   then return $(V, \emptyset)$
3. set i to $c \mod 2$
4. set W_{1-i} to $\emptyset$
5. repeat

   5.1 $\mathcal{G}' := \mathcal{G} \setminus Attr_i(\alpha^{-1}(c), \mathcal{G})$
   5.2 $(W'_0, W'_1) := McNaughton(\mathcal{G}')$
   5.3 if $(W'_{1-i} = \emptyset)$ then
      5.3.1 $W_i := V \setminus W_{1-i}$
      5.3.2 return (W_0, W_1)
   5.4 $W_{1-i} := W_{1-i} \cup Attr_{(1-i)}(W'_{1-i}, \mathcal{G})$
   5.5 $\mathcal{G} := \mathcal{G} \setminus Attr_{(1-i)}(W'_{1-i}, \mathcal{G})$

McNaughton's Algorithm: Solving parity games

$McNaughton(\mathcal{G})$

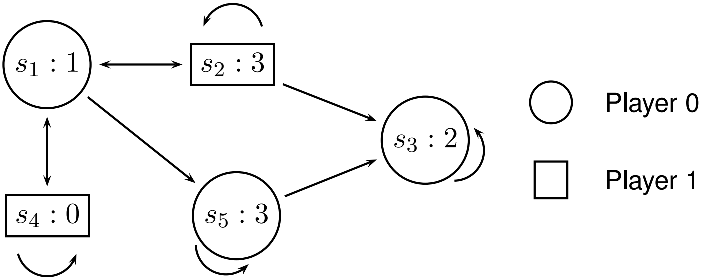
1. $c :=$ highest color in $\mathcal{G}$
2. if $c = 0$ or $V = \emptyset$
   then return $(V, \emptyset)$
3. set i to $c \mod 2$
4. set W_{1-i} to $\emptyset$
5. repeat

   5.1 $\mathcal{G}' := \mathcal{G} \setminus Attr_i(\alpha^{-1}(c), \mathcal{G})$
   5.2 $(W'_0, W'_1) := McNaughton(\mathcal{G}')$
   5.3 if $(W'_{1-i} = \emptyset)$ then

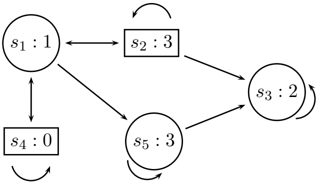
      5.3.1 $W_i := V \setminus W_{1-i}$
      5.3.2 return (W_0, W_1)

   5.4 $W_{1-i} := W_{1-i} \cup Attr_{(1-i)}(W'_{1-i}, \mathcal{G})$
   5.5 $\mathcal{G} := \mathcal{G} \setminus Attr_{(1-i)}(W'_{1-i}, \mathcal{G})$

Example



Example

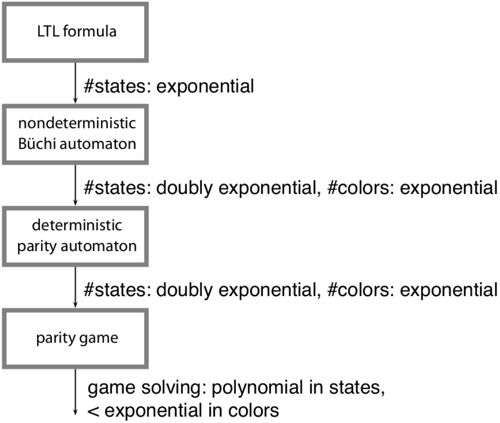


$McNaughton(\mathcal{G})$

1. $c :=$ highest color in $\mathcal{G}$
2. if $c = 0$ or $V = \emptyset$
   then return $(V, \emptyset)$
3. set i to $c \mod 2$
4. set W_{1-i} to $\emptyset$
5. repeat
   - 5.1 $\mathcal{G}' := \mathcal{G} \setminus Attr_i(\alpha^{-1}(c), \mathcal{G})$
   - 5.2 $(W_0', W_1') := McNaughton(\mathcal{G}')$
   - 5.3 if $(W_{1-i}' = \emptyset)$ then
     - 5.3.1 $W_i := V \setminus W_{1-i}$
     - 5.3.2 return (W_0, W_1)
   - 5.4 $W_{1-i} := W_{1-i} \cup Attr_{(1-i)}(W_{1-i}', \mathcal{G})$
   - 5.5 $\mathcal{G} := \mathcal{G} \setminus Attr_{(1-i)}(W_{1-i}', \mathcal{G})$

- $k = 3$, $c^{-1}(3) = \{s_2, s_5\}$, $i = 1$
- $\mathcal{G}' := \mathcal{G} \setminus Attr_1(\{s_2, s_5\}, \mathcal{G}) = \{s_1, s_4, s_3\}$
- $(\{s_3\}, \{s_1, s_4\}) = McNaughton(\mathcal{G}')$
- $W_0 := \emptyset \cup Attr_0(\{s_3\}, \mathcal{G}) = \{s_1, s_3, s_5\}$
- $\mathcal{G} := \mathcal{G} \setminus Attr_0(\{s_3\}, \mathcal{G}) = \{s_2, s_4\}$
- $\mathcal{G}' := \mathcal{G} \setminus Attr_1(\{s_2\}, \mathcal{G}) = \{s_4\}$
- $(\{s_4\}, \emptyset) = McNaughton(\mathcal{G}')$
- $W_0 = \{s_1, s_3, s_5\} \cup \{s_4\}$
- $\mathcal{G} := \mathcal{G} \setminus \{s_4\} = \{s_2\}$
- $\mathcal{G}' = \mathcal{G} \setminus Attr_1(\{s_2\}, \mathcal{G}) = \emptyset$
- $(\emptyset, \emptyset) = McNaughton(\mathcal{G}')$
- $W_1 = V \setminus \{s_1, s_3, s_4, s_5\} = \{s_2\}$
- return $(\{s_1, s_3, s_4, s_5\}, \{s_2\})$

LTL synthesis



LTL synthesis

LTL synthesis is 2EXPTIME-complete.