Fourth Lecture:
On Optimal Strategies in Timed Games

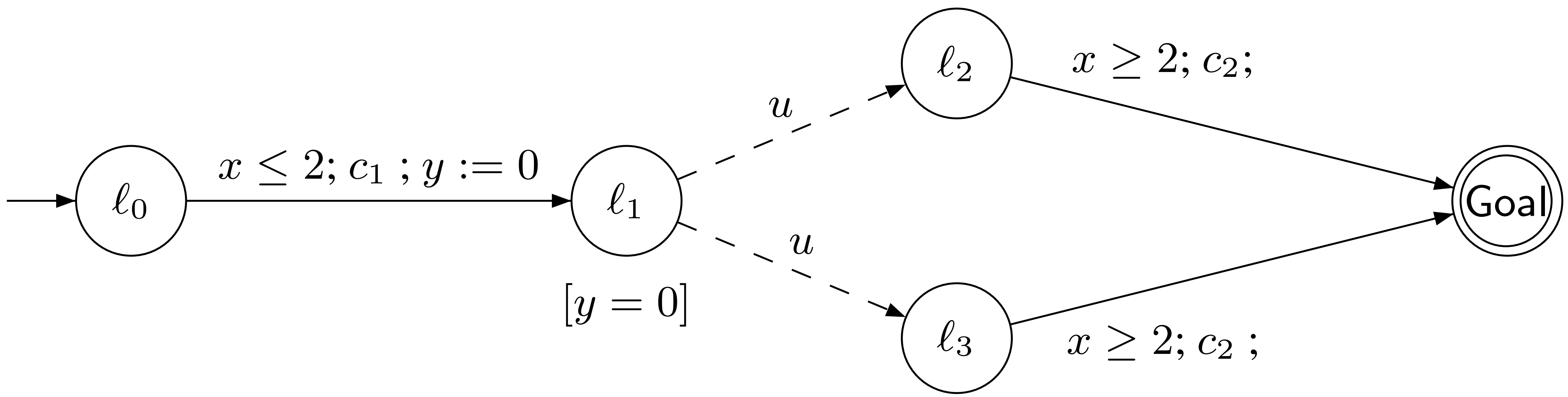
Jean-François Raskin
Université Libre de Bruxelles
Belgium

Artist2 Asian Summer School - Shanghai - July 2008

Preliminaries and motivations

Timed Automata

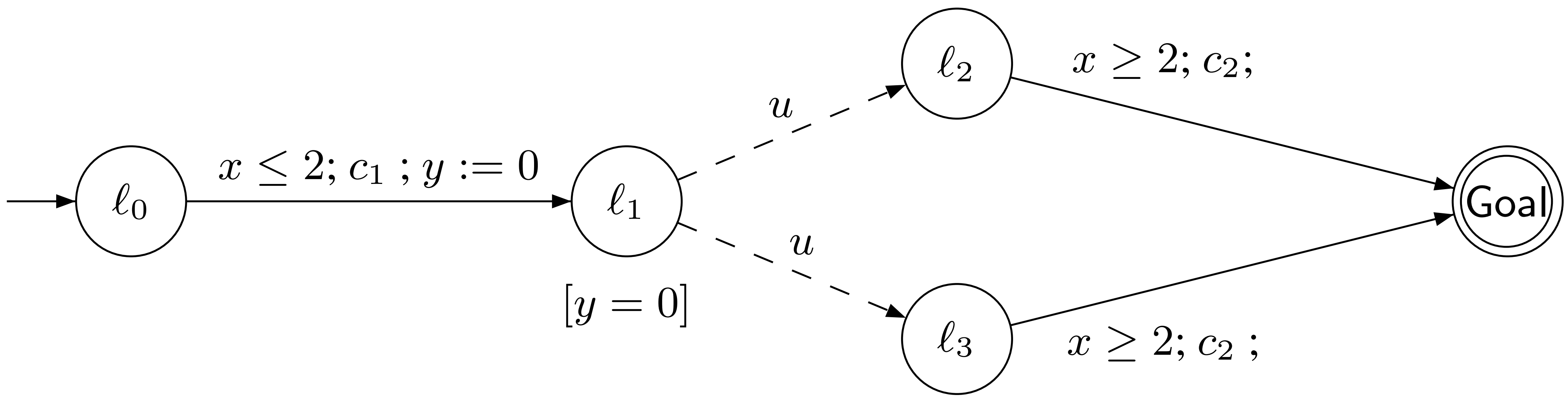
[Alur&Dill 94]



- ➢ A ***timed automaton*** = finite state automaton + a set ***C*** of *clocks*;
- ➢ A ***state*** is a pair **(*l*,*v*)** where ***l*** is a *location*, ***v*** is a *clock valuation* (***v*** : **C**→*R*$^+$);
- ➢ ***Clocks***: evolve with time (***x'*=1**),
  can be reset (***x*:=0**)
  and compared to constants (**x ~ c**).

Timed Automata

[Alur&Dill 94]



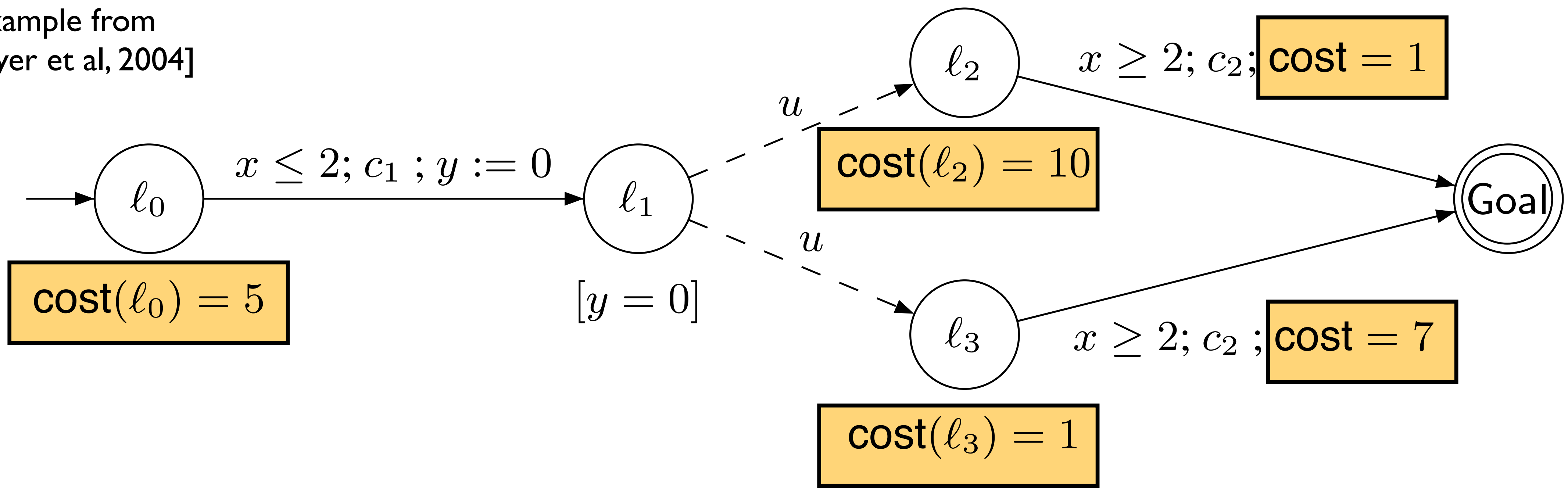
Run of TA:

$(l_0,(0,0)) \xrightarrow{0.8} (l_0,(0.8,0.8)) \xrightarrow{c_1} (l_1,(0.8,0)) -u- (l_2,(0.8,0)) \xrightarrow{3.1} (l_2,(3.9,3.1)) \xrightarrow{c_2} (Goal,(3.9,3.1))$

Weighted Timed Automata

[Alur et al. & Larsen et al., 2001]

Example from
[Bouyer et al, 2004]

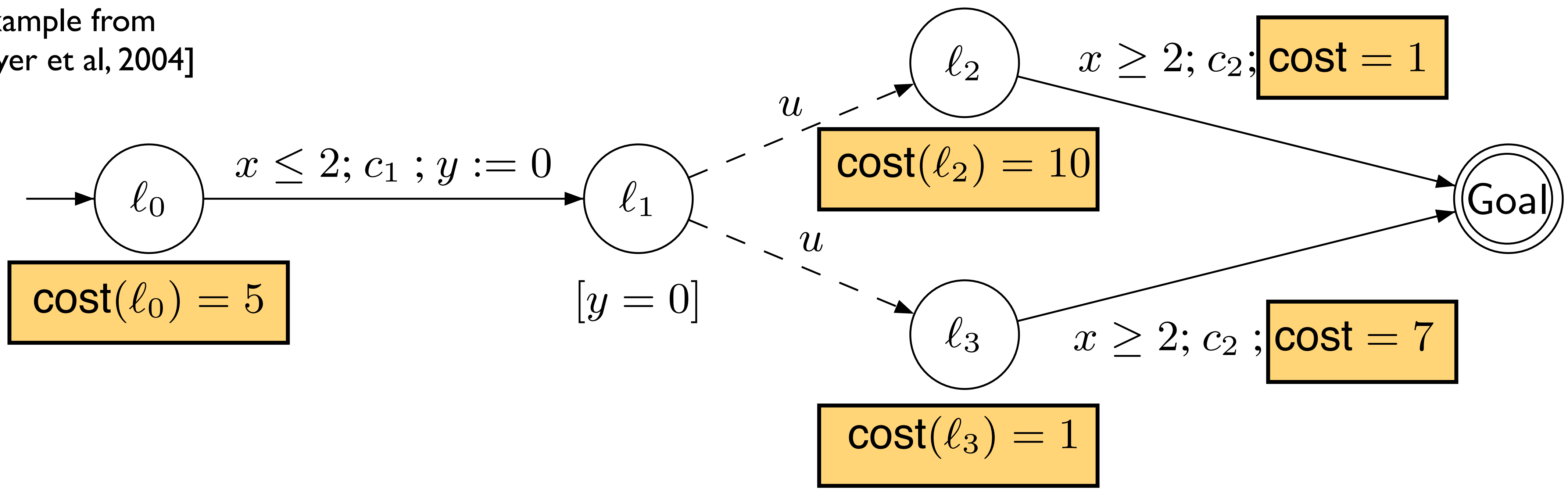


Locations are annotated with a ***cost (weight)*** per time unit (*derivative*)
Transitions are annotated with a ***cost (weight).***

Weighted Timed Automata

[Alur et al. & Larsen et al., 2001]

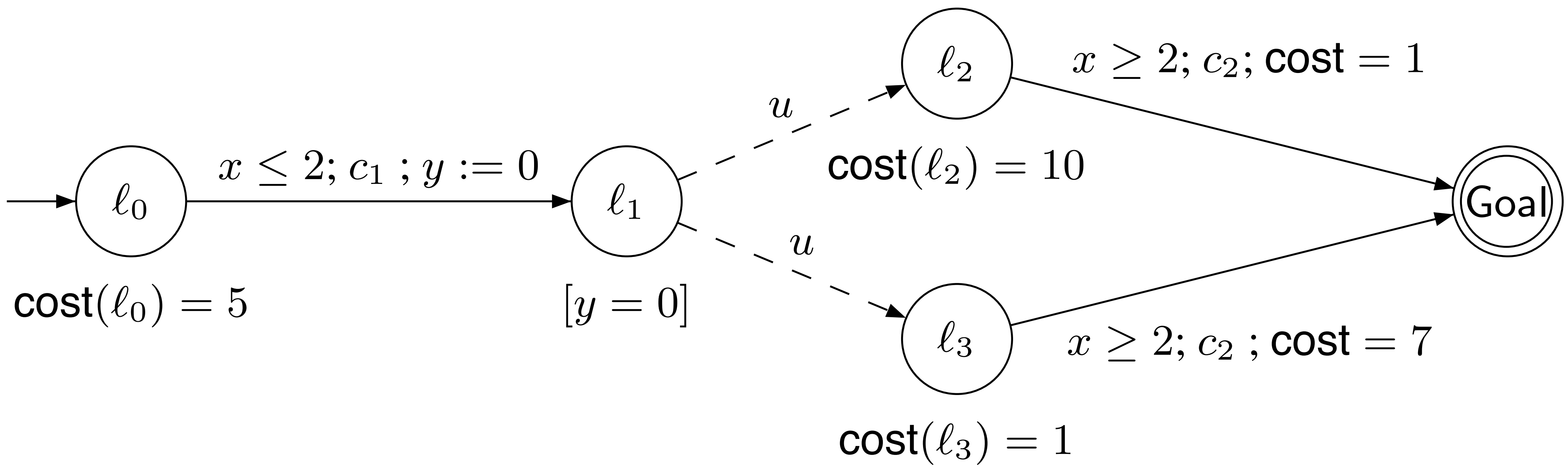
Example from
[Bouyer et al, 2004]



$(l_0,(0,0),0) -_{0.8}\rightarrow (l_0,(0.8,0.8),4) -c_1\rightarrow (l_1,(0.8,0),4) -u- (l_2,(0.8,0),4) -_{3.1}\rightarrow (l_2,(3.9,3.1),35) -c_2\rightarrow (Goal,(3.9,3.1),36)$

Costs are accumulated but are not “tested” along the run.

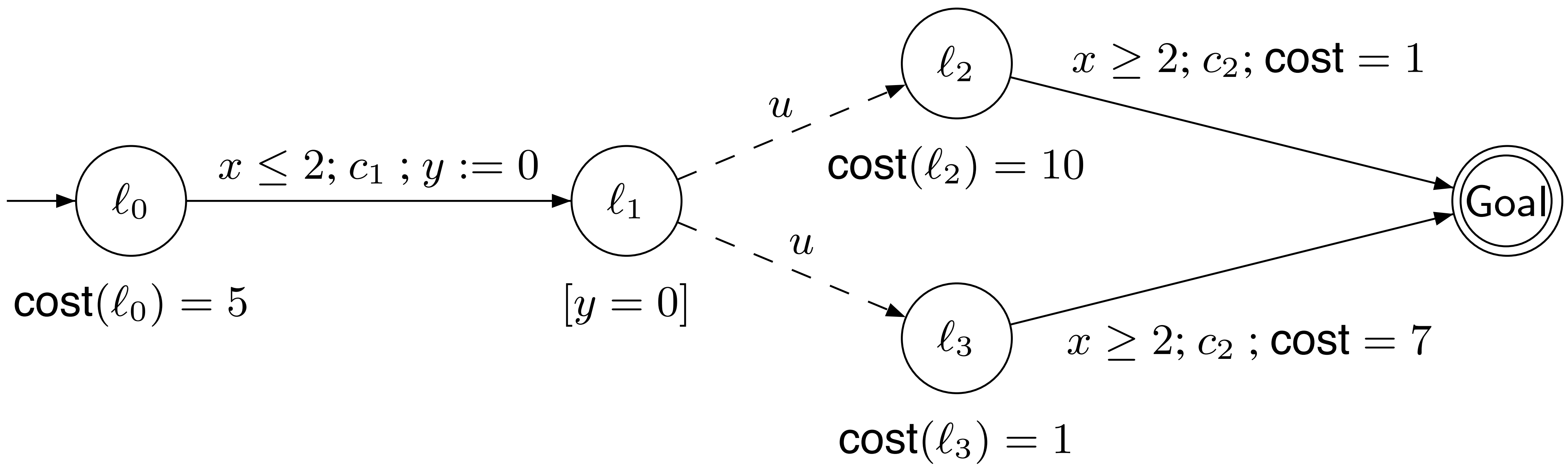
Weighted Timed Automata



Optimal Reachability Problem:

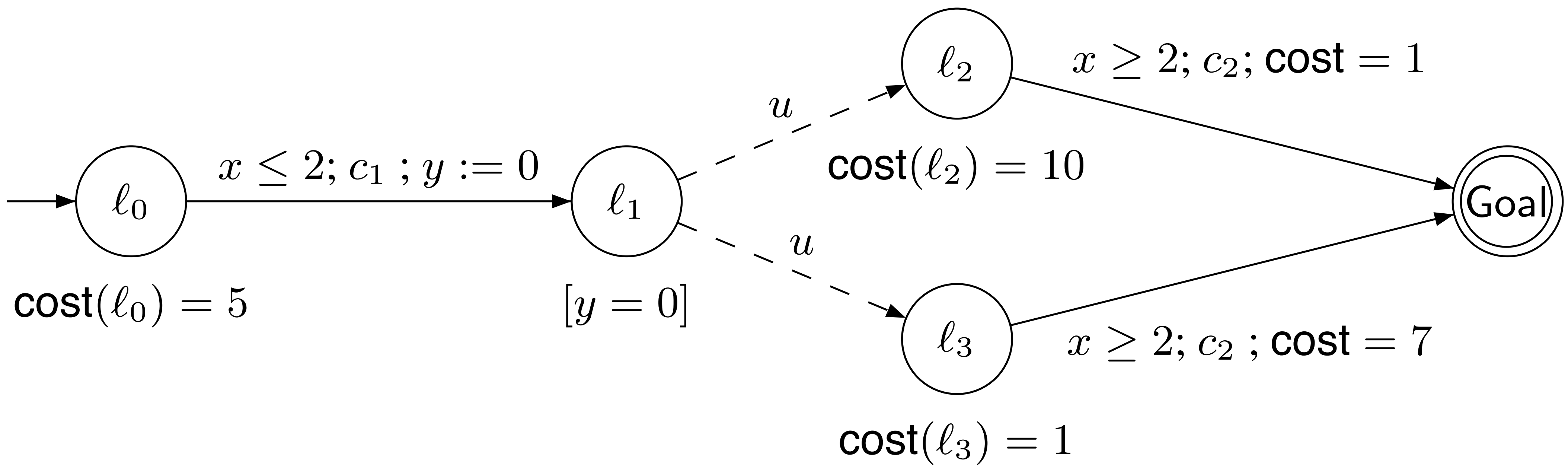
given a state ***(l,v)***, a set of states ***Goal*** and a cost ***c***, decide if there exists a run from ***(l,v)*** to ***Goal*** with **cost** bounded by ***c***.

Weighted Timed Automata



What is the ***optimal run*** from $(l_0,(0,0))$ to Goal ?

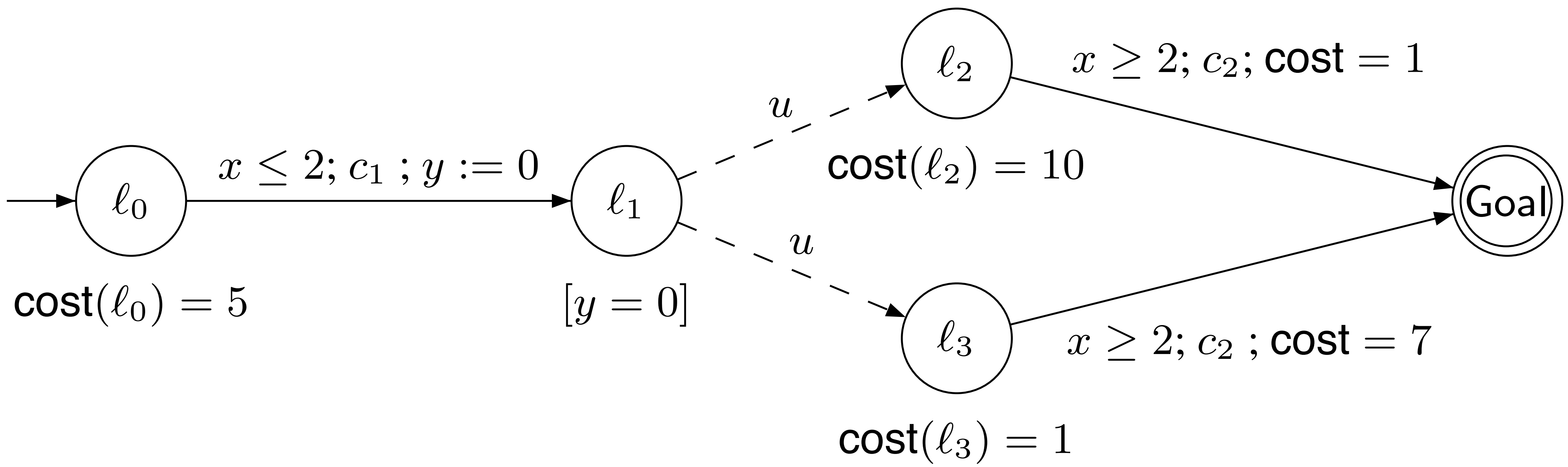
Weighted Timed Automata



What is the ***optimal run*** from $(l_0,(0,0))$ to Goal ?
We have to decide:

- ➢ how much time to stay in l_0 (noted *t*) ?
- ➢ which branch to take in l_1 ?

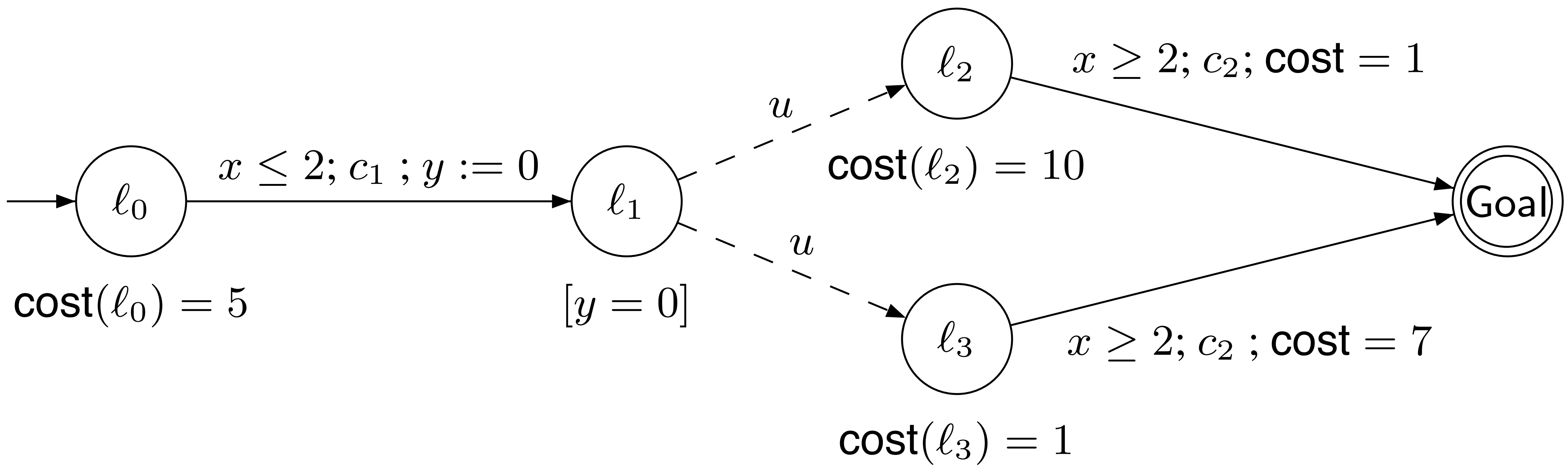
Weighted Timed Automata



So, the **minimal weight** to reach Goal is equal to

$$\text{Min}_t(5t + 10(2 - t) + 1, 5t + (2 - t) + 7) \text{ with } 0 \leq t \leq 2$$

Weighted Timed Automata

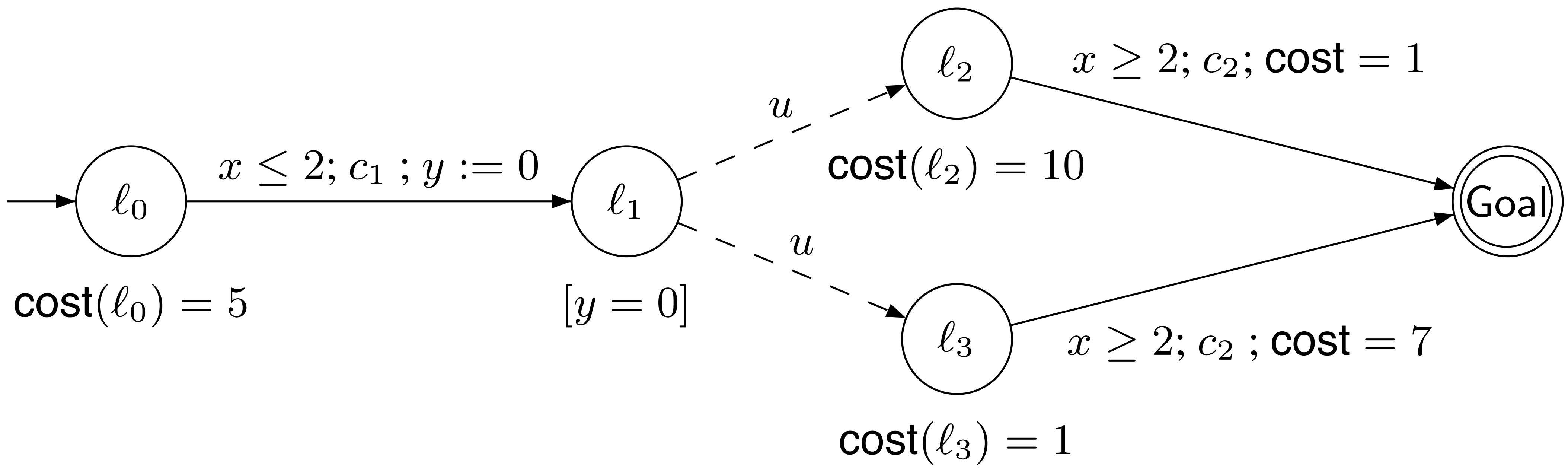


So, the **minimal weight** to reach Goal is equal to

$$\mathrm{Min}_t(5t + 10(2 - t) + 1, 5t + (2 - t) + 7) \text{ with } 0 \leq t \leq 2$$

t=0, branch down, optimal weight equals **9**.

Weighted Timed Automata



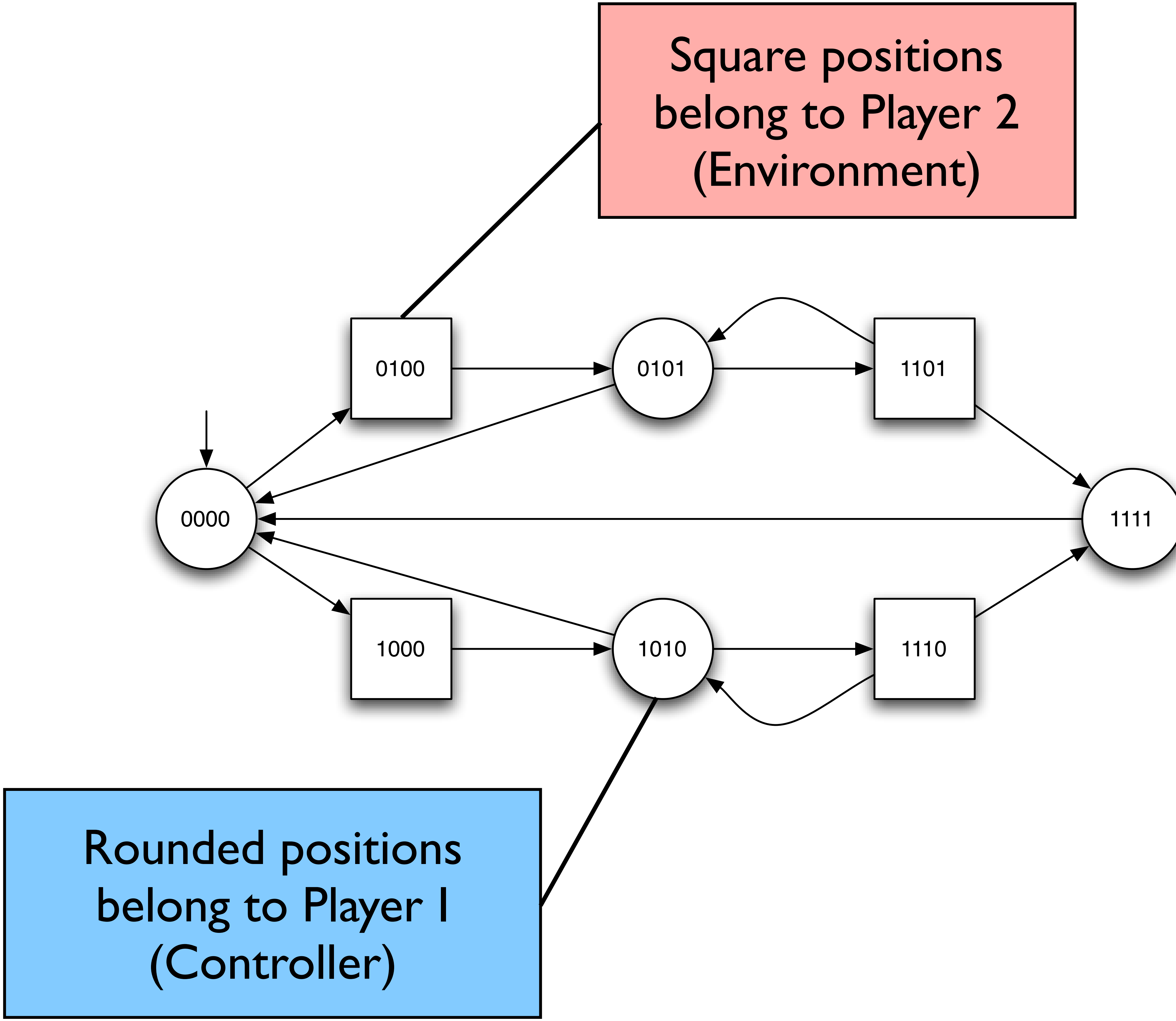
Optimal reachability is **decidable** [ALP01,BFHLPRV01], it is **PSpace Complete** [BBBR07].
Need extensions of regions (theoretical complexity) and zones (for useful symbolic algorithms, see UppAll-CORA).

Weighted Timed Automata

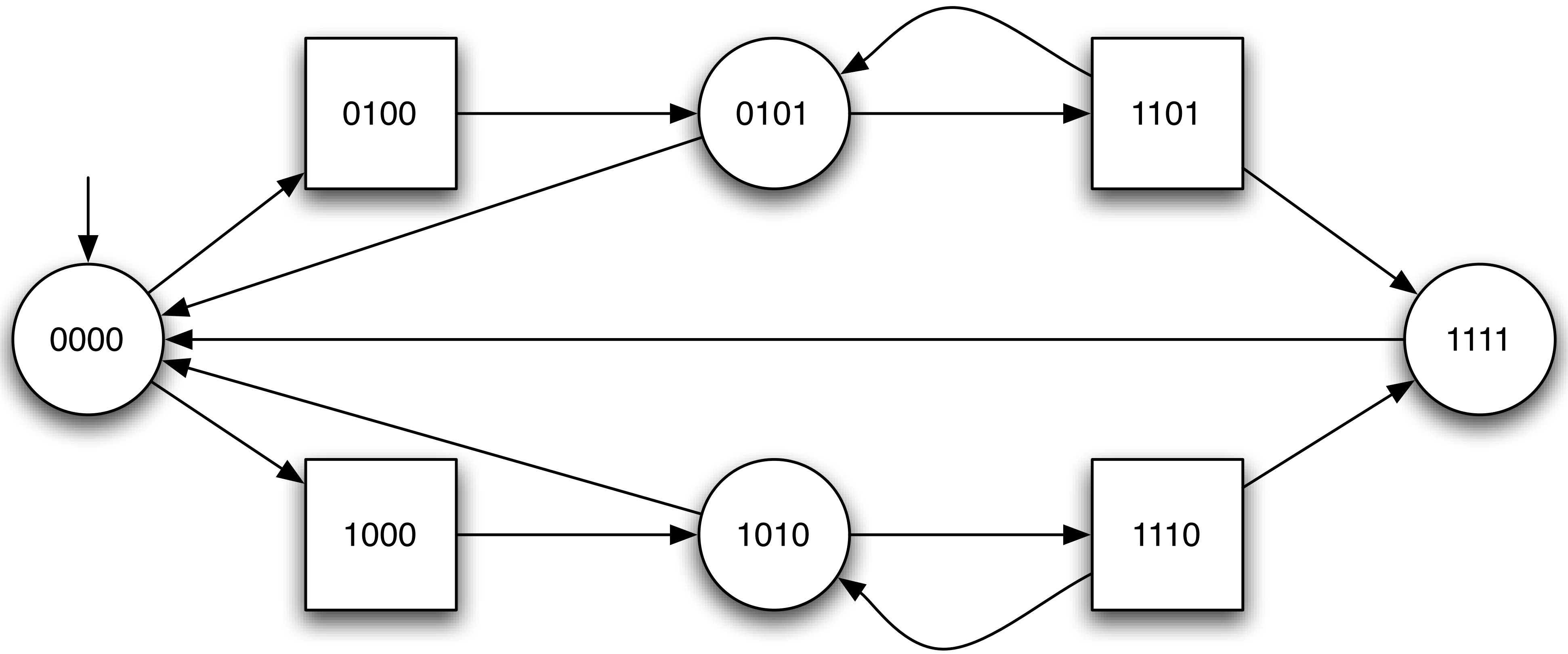
- ... are natural models for timed systems with ***resource constraints*** ;
- ... useful to model ***embedded digital controllers*** ;
- so, we should consider ***games*** on **WTA** for controller **synthesis**.

What is on the menu ?

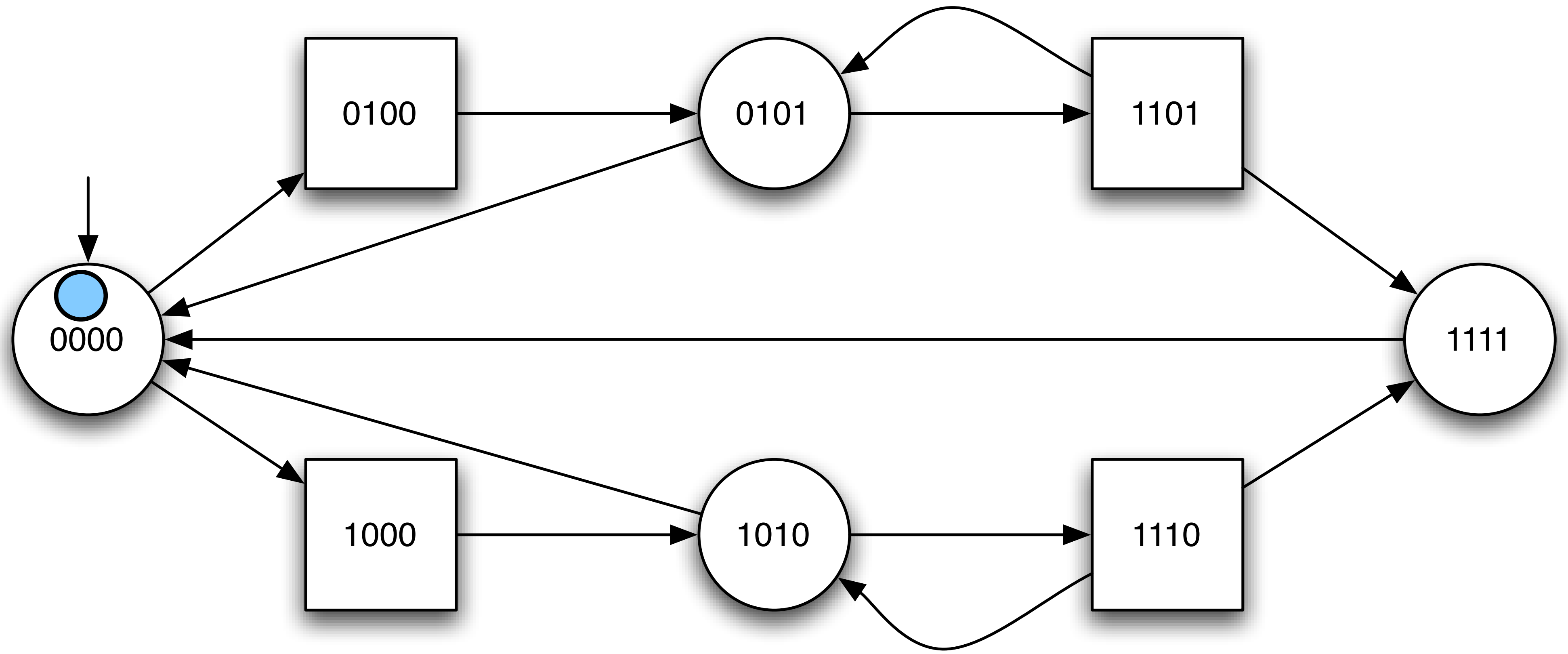
- Premilinaries and motivations
- Two-player games - Fixed point algorithms
- Games on Weighted Timed Automata
- Symbolic semi-algorithm
- Undecidability
- Decidable subcases
- Conclusion - Bibliography


Square positions belong to Player 2 (Environment)
0100
0101
1101
0000
1111
1000
1010
1110
Rounded positions belong to Player 1 (Controller)

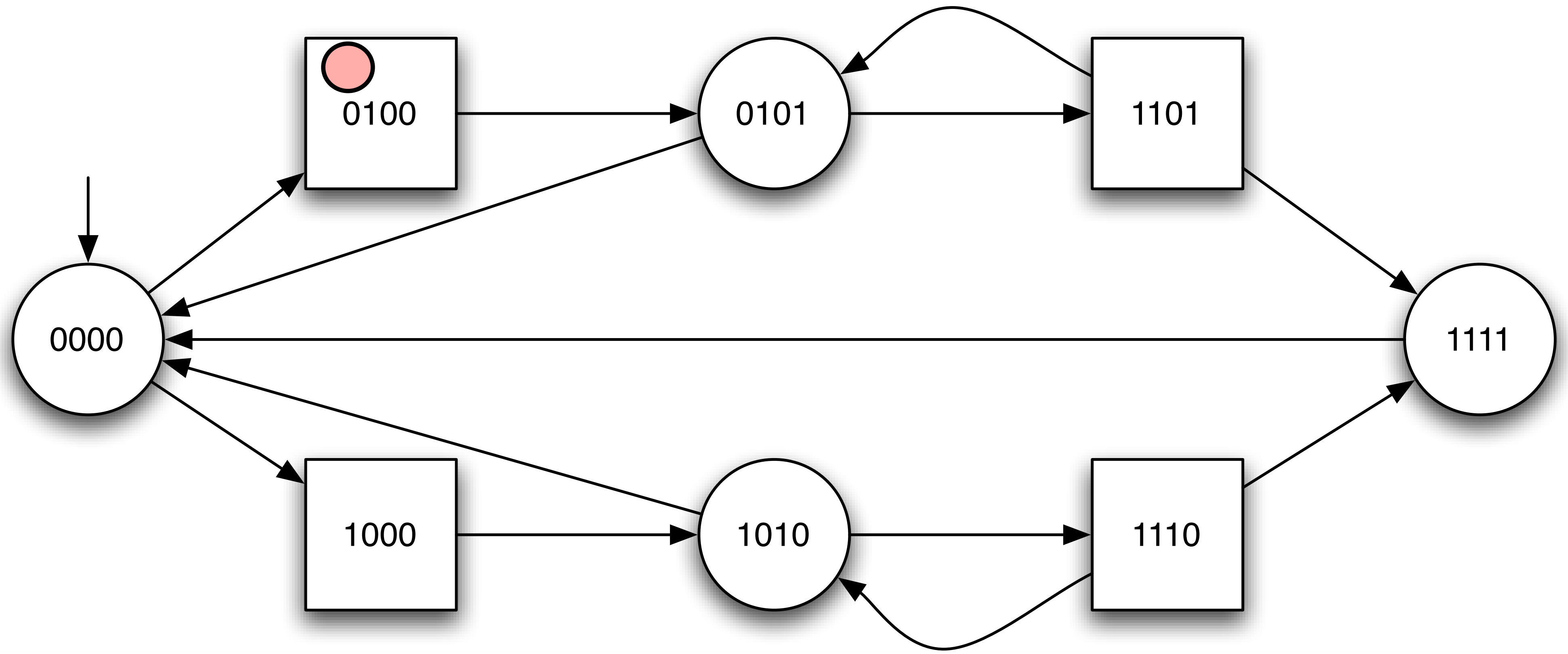
Rounded positions belong to Player I
Square positions belong to Player 2



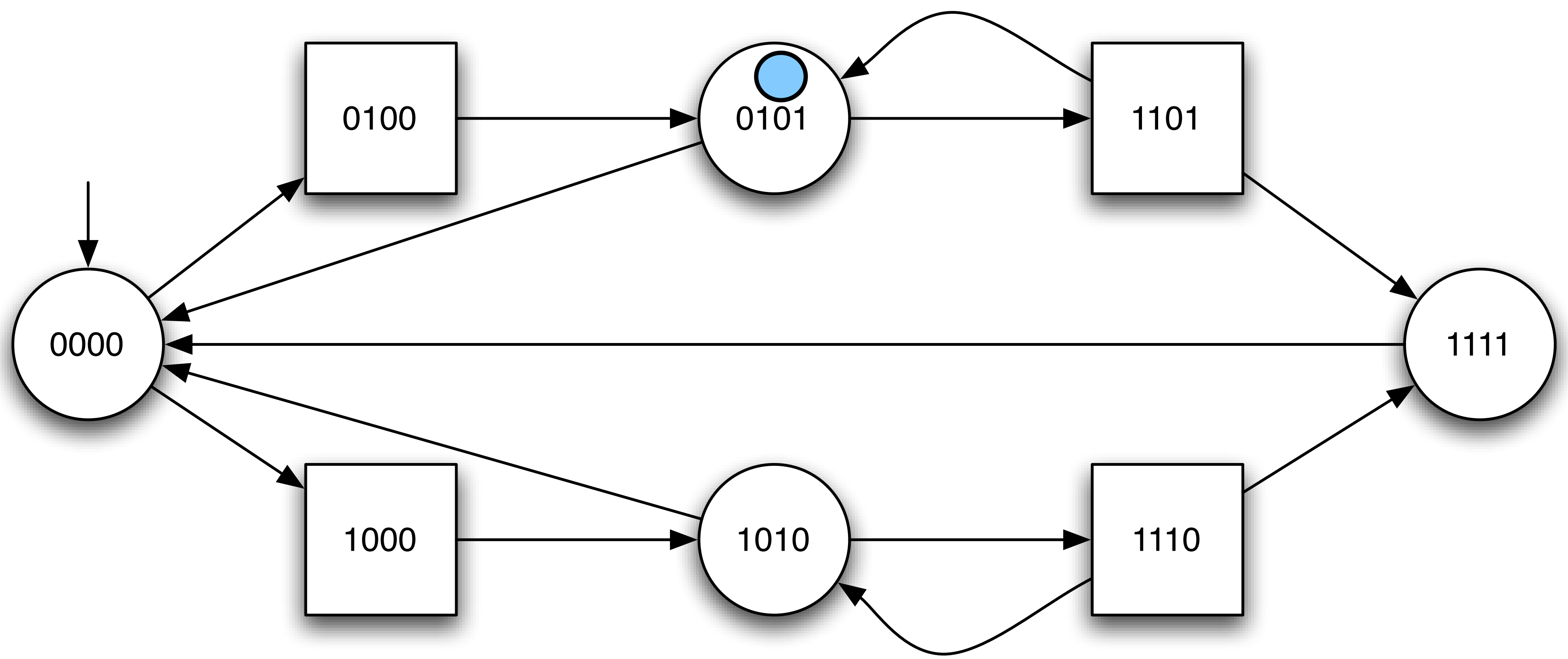
A game is played as follows: in each **round**, the game is in a **position**, if the game is in a rounded position, Player I resolves the **choice** for the next state, if the game is in a square position, Player 2 resolves the choice. The game is played for an **infinite number of rounds**.



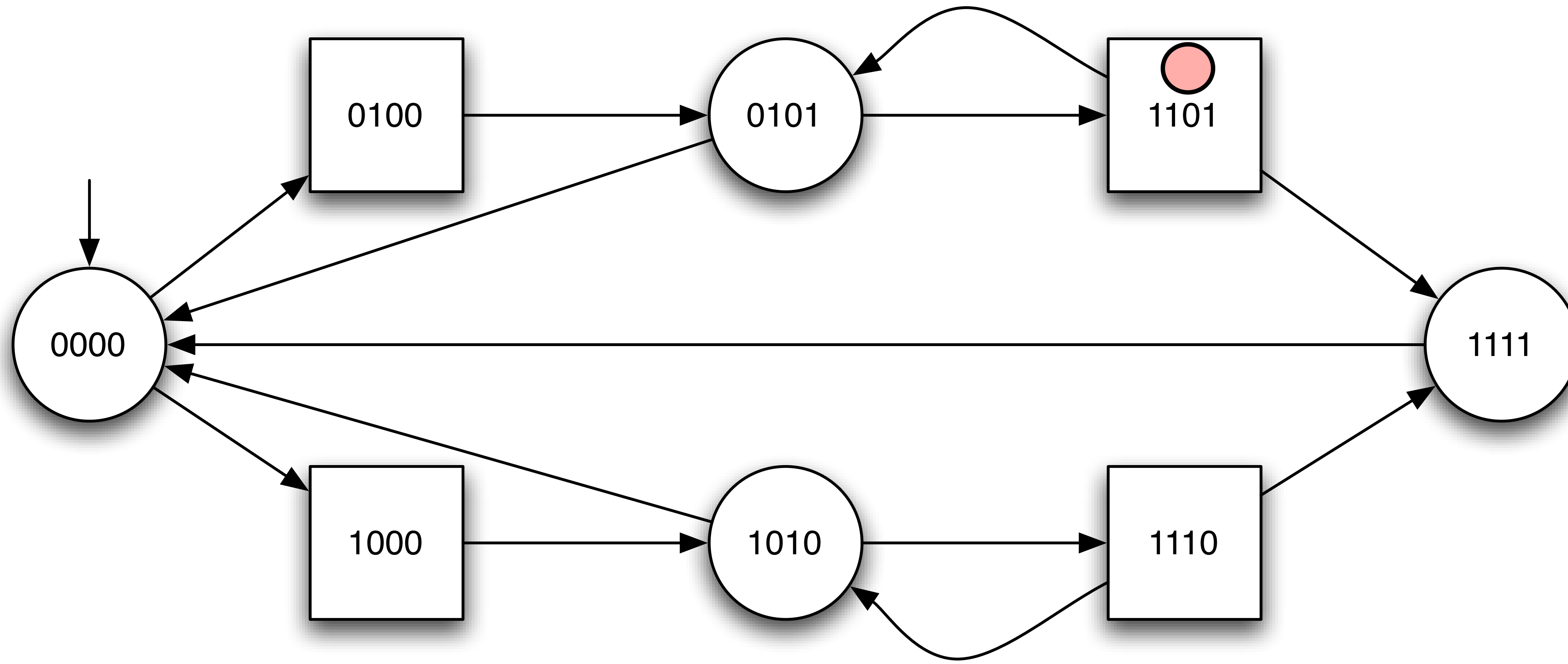
Play : 0000



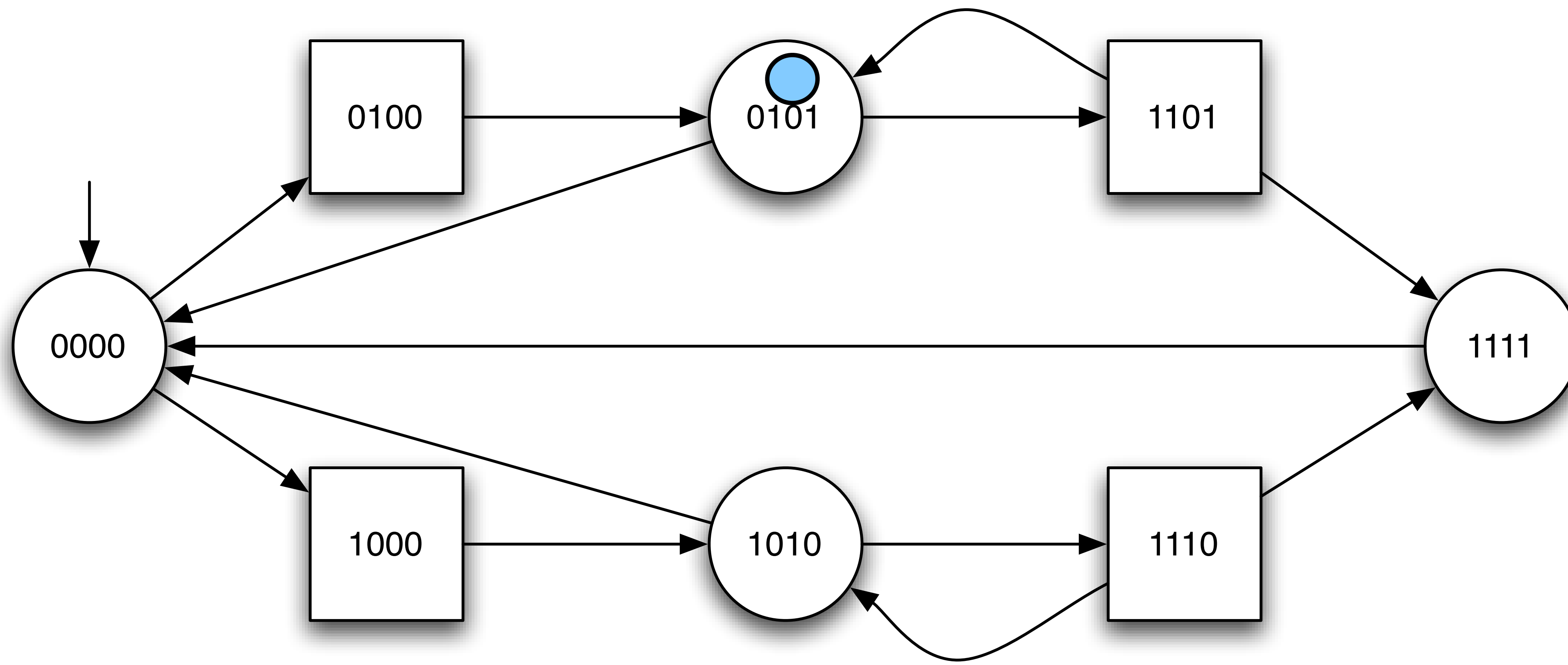
Play : 0000 0100



Play : 0000 0100 0101

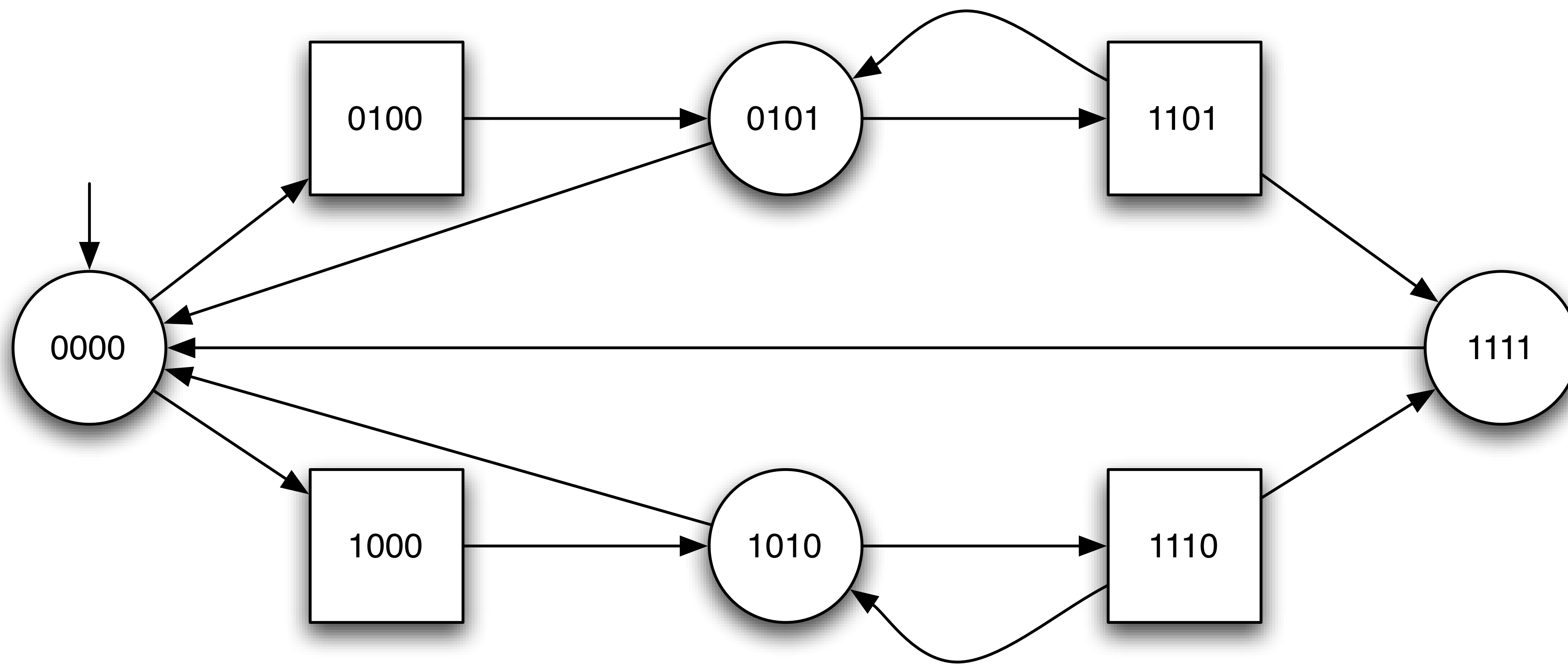


Play : 0000 0100 0101 1101



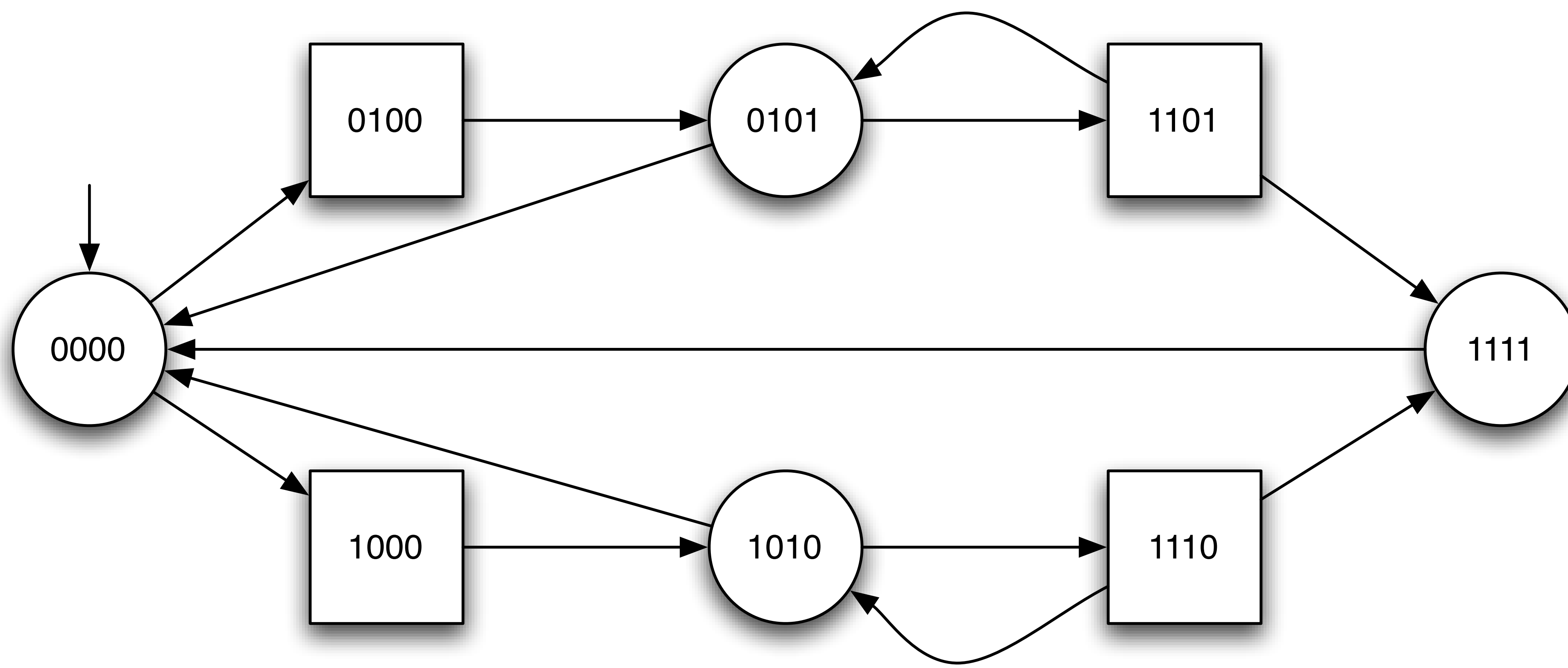
Play : 0000 0100 0101 1101 ...

Who is winning ?



Play : 0000 0100 0101 1101 ...

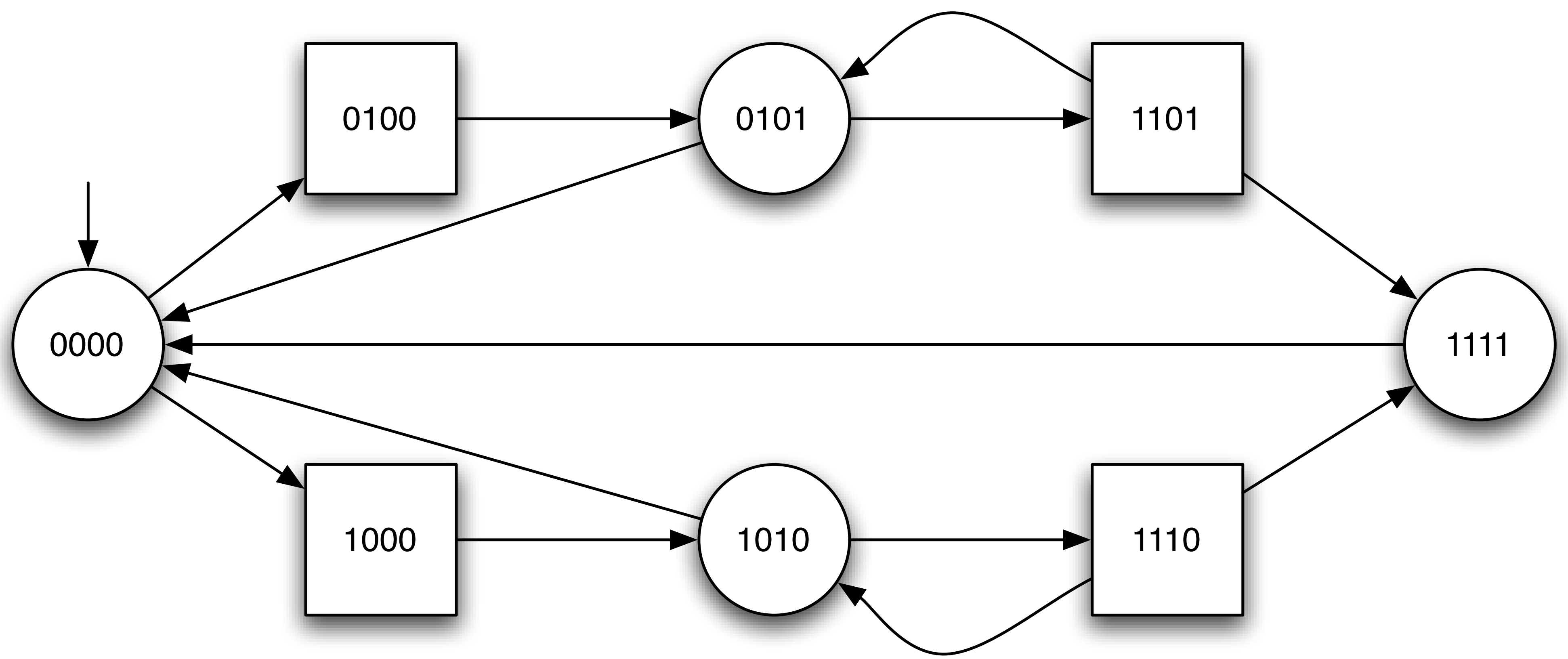
Who is winning ?



Play : 0000 0100 0101 1101 ...

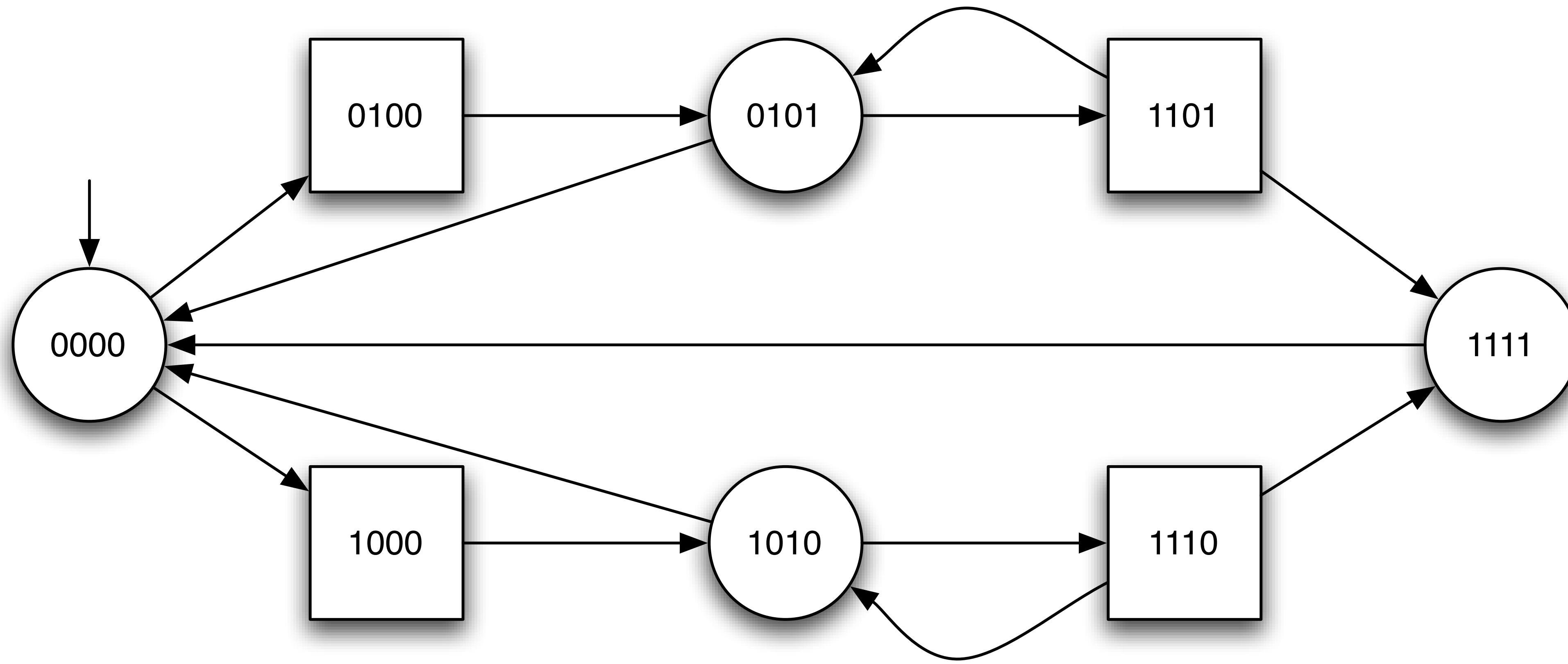
Is this a **good** or a **bad** play for **Player 1** ?

Who is winning ?



A **winning condition** (for Player 1)
is a set of plays
$$W \subseteq (Q_1 \cup Q_2)^\omega$$

Who is winning ?



Example of a winning condition:
The set of plays that reach 1111
This is called a **reachability objective**.

Strategies

Players are playing **according to strategies.**

A **strategy for Player I** is a function that, given a *sequence of positions* (visited so far) that ends in a *Player I's position*, returns the *choice for the next position*.

$$\lambda_I(\underbrace{0011\ 1001\ 1101\ \overbrace{0011}^{\text{Player I's position}}}_{\text{prefix of play}})=\underbrace{1110}_{\text{Choice for the next position}}$$

Strategies

Players are playing **according to strategies.**

A **strategy for Player I** is a function that, given a *sequence of positions* (visited so far) that ends in a *Player I's position*, returns the *choice for the next position*.

Strategies for Player II
are defined symetrically

Outcome of strategies

If we **fix** a strategy for the two players and we let the two players apply their strategies, we get a play:

$$\text{Outcome}(\lambda_1,\lambda_2)=1100\ 0011\ 0001\ 0011\ \ldots$$

If we fix a strategy **only** for Player I, we get a set of plays

$$\text{Outcome}(\lambda_1)=\cup_{\lambda_2}\ \text{Outcome}\ (\lambda_1,\lambda_2)$$

A strategy for Player I is **winning** for objective *W* iff

$$\text{Outcome}(\lambda_1) \subseteq W$$

Outcome of strategies

A strategy for Player I is **winning** for objective W iff

$$\text{Outcome}(\lambda_I) \subseteq W$$

That is, *no matter how Player II resolves his choices*, when player I **plays according to** λ_I the resulting play belongs to W.

Player I can **force** the play to be in W.

Winning strategies

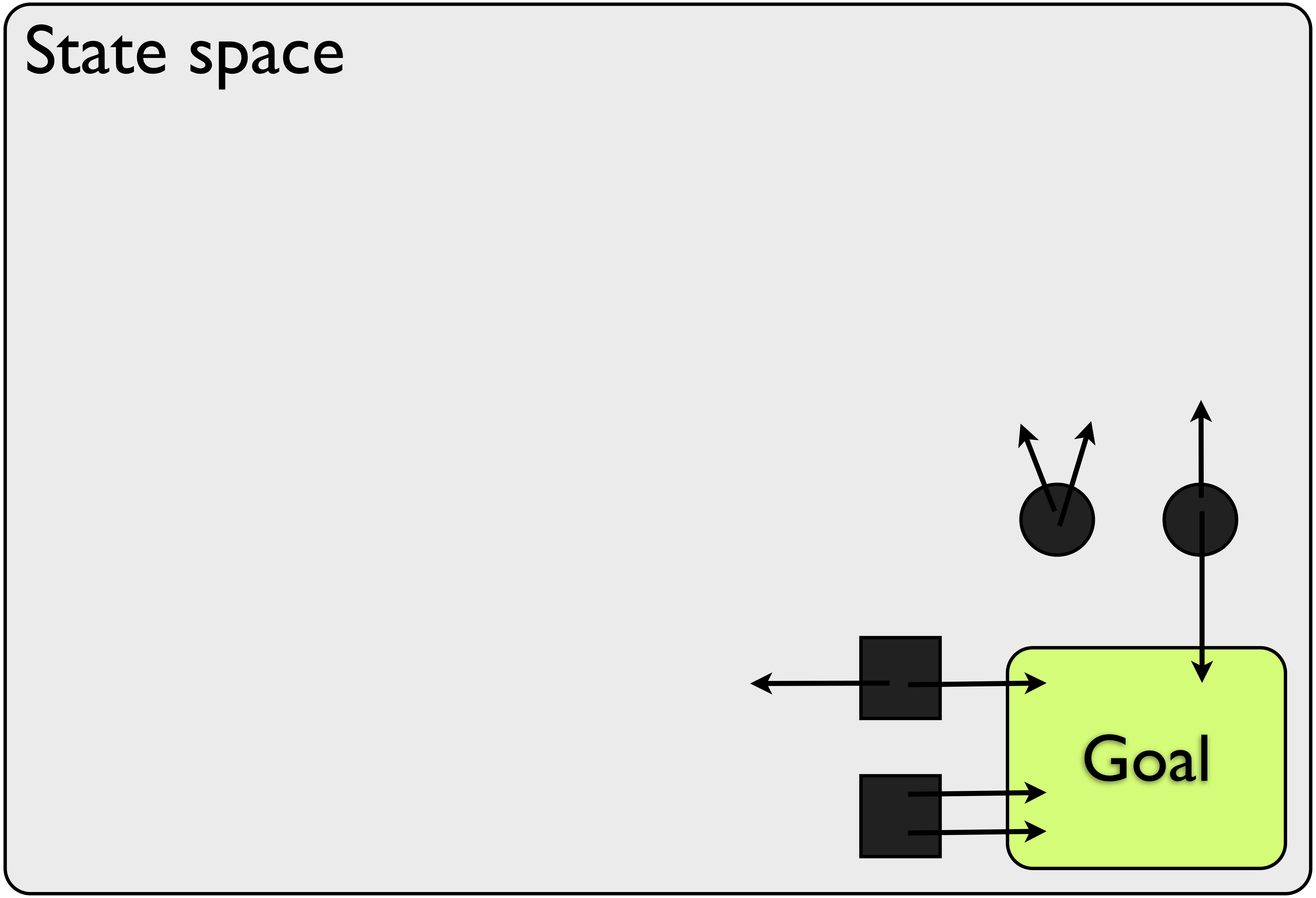
=

Controllers that enforce winning plays

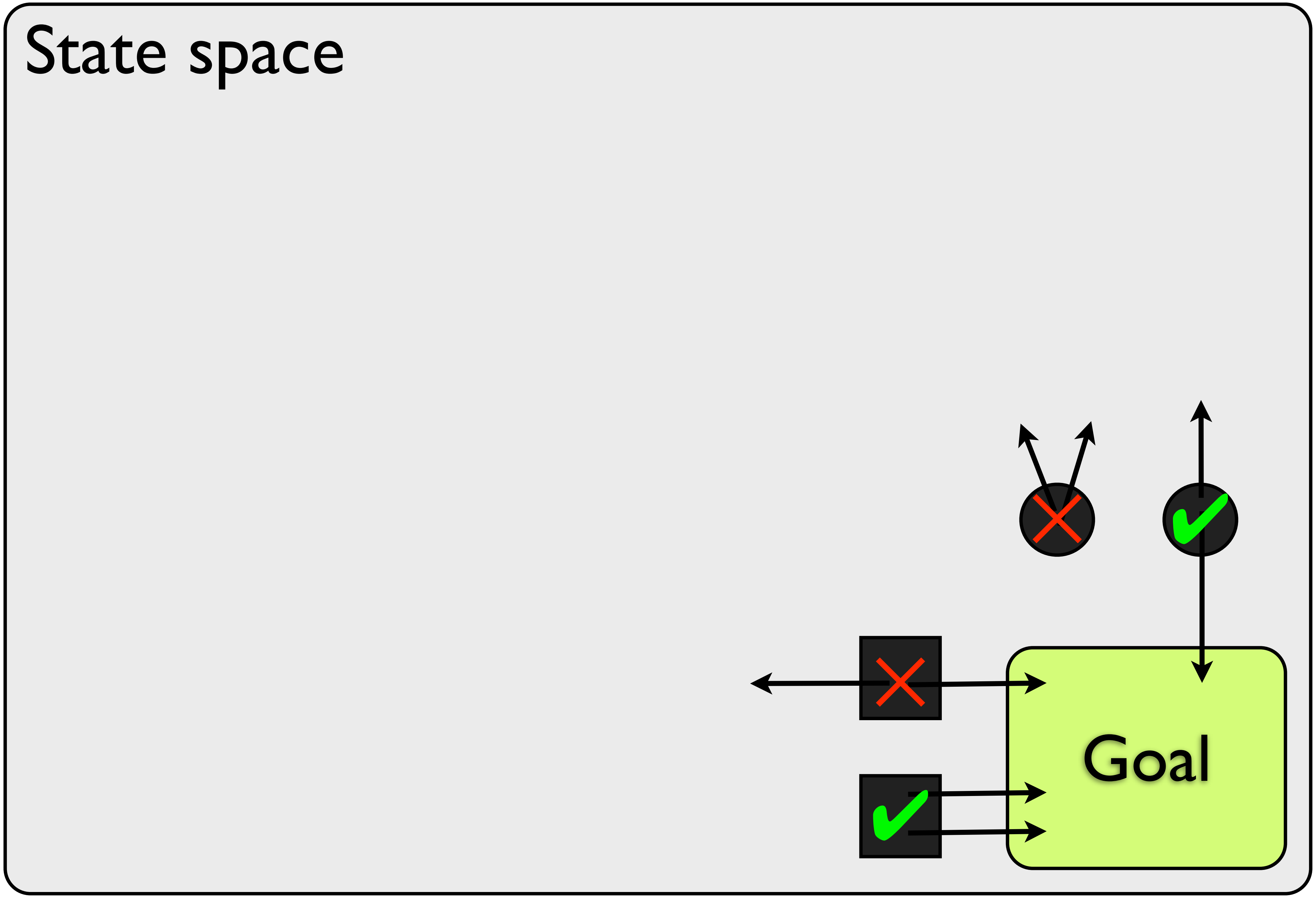
Algorithms for reachability in Two-Player Games

Reachability objective: Goal
What are the winning states for player 1 ?

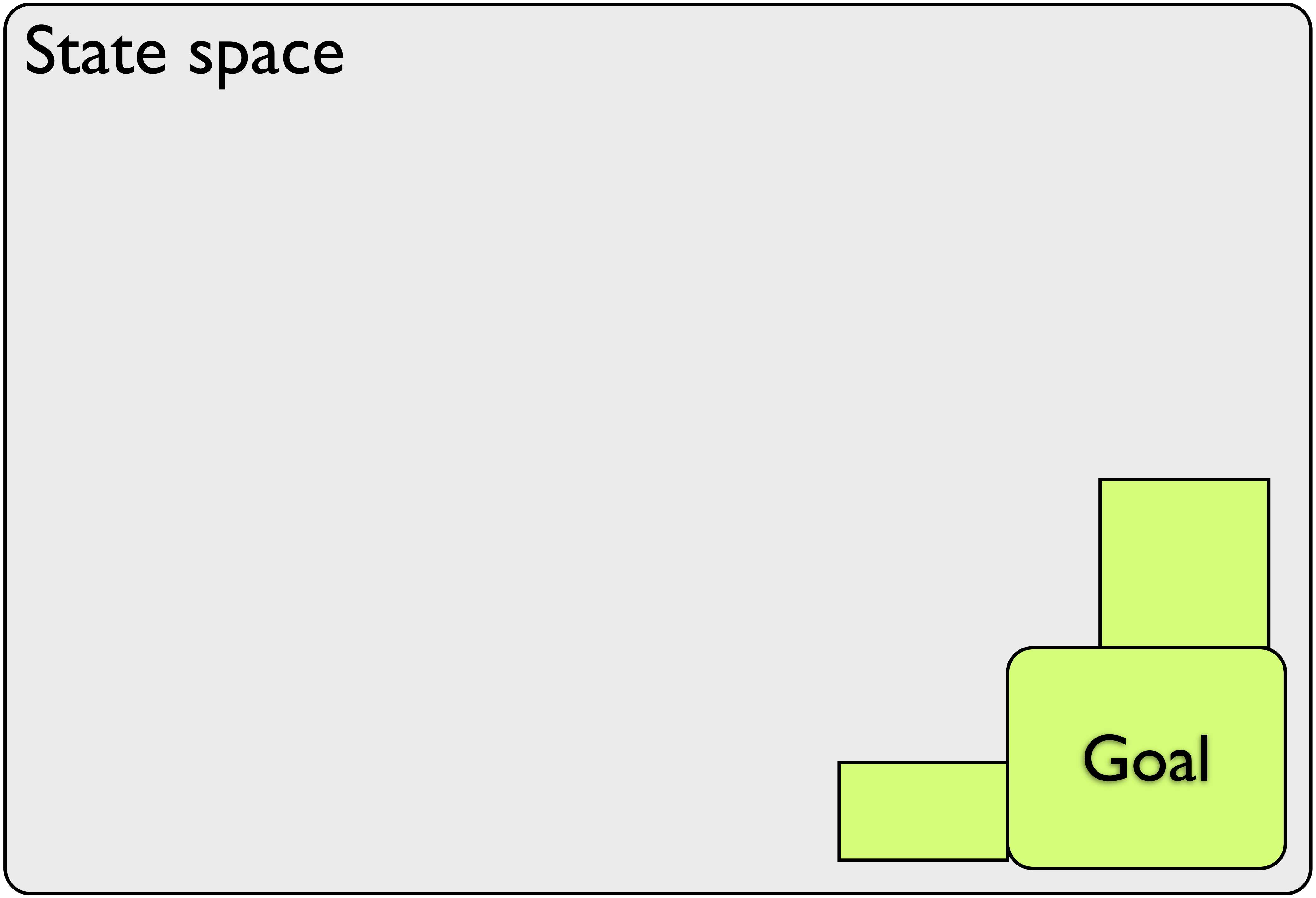
Reachability objective: Goal
What are the winning states for player I ?



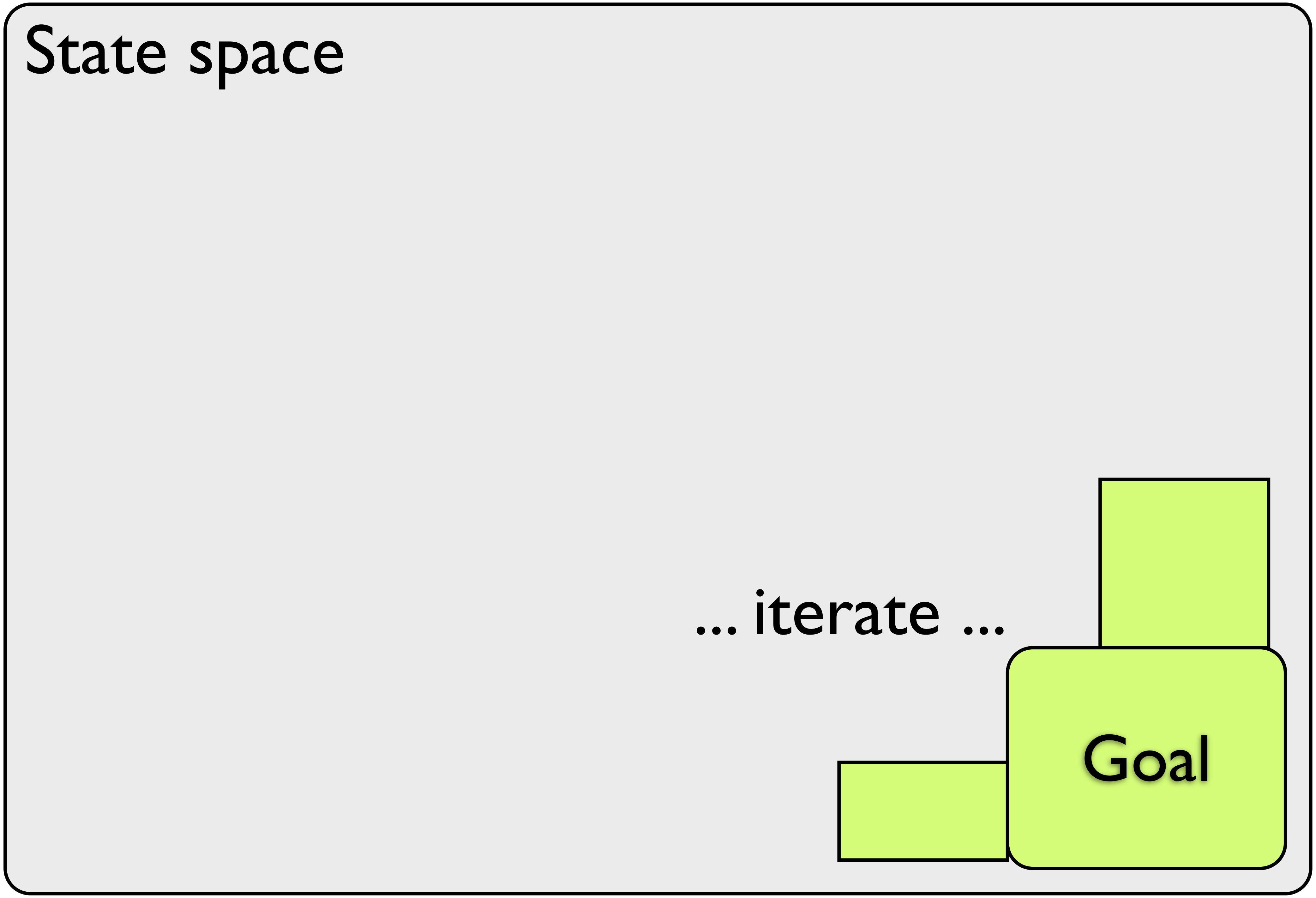
Reachability objective: Goal
What are the winning states for player I ?



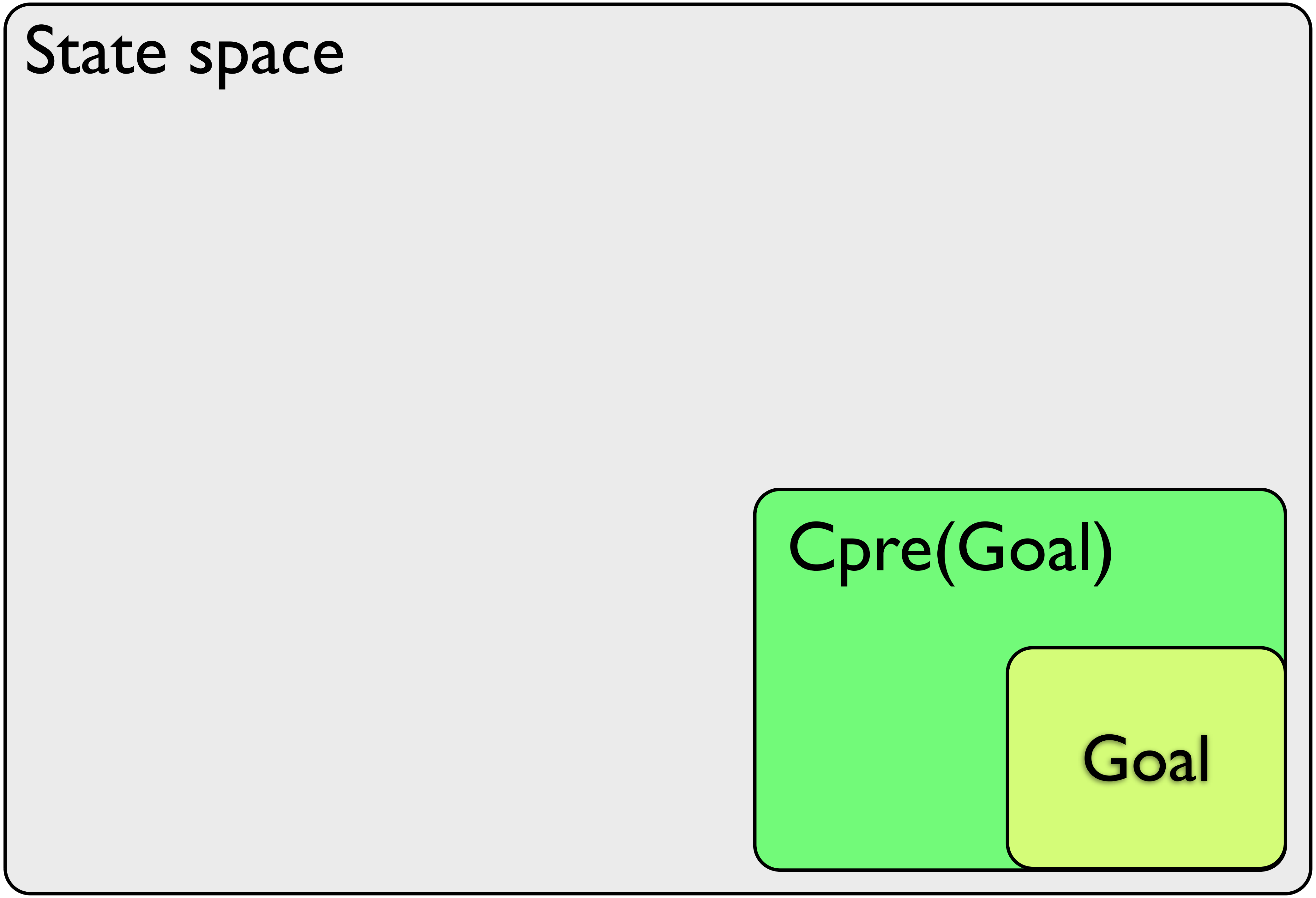
Reachability objective: Goal
What are the winning states for player I ?



Reachability objective: Goal
What are the winning states for player I ?



Reachability objective: Goal
What are the winning states for player I ?



Player I Controllable Predecessors

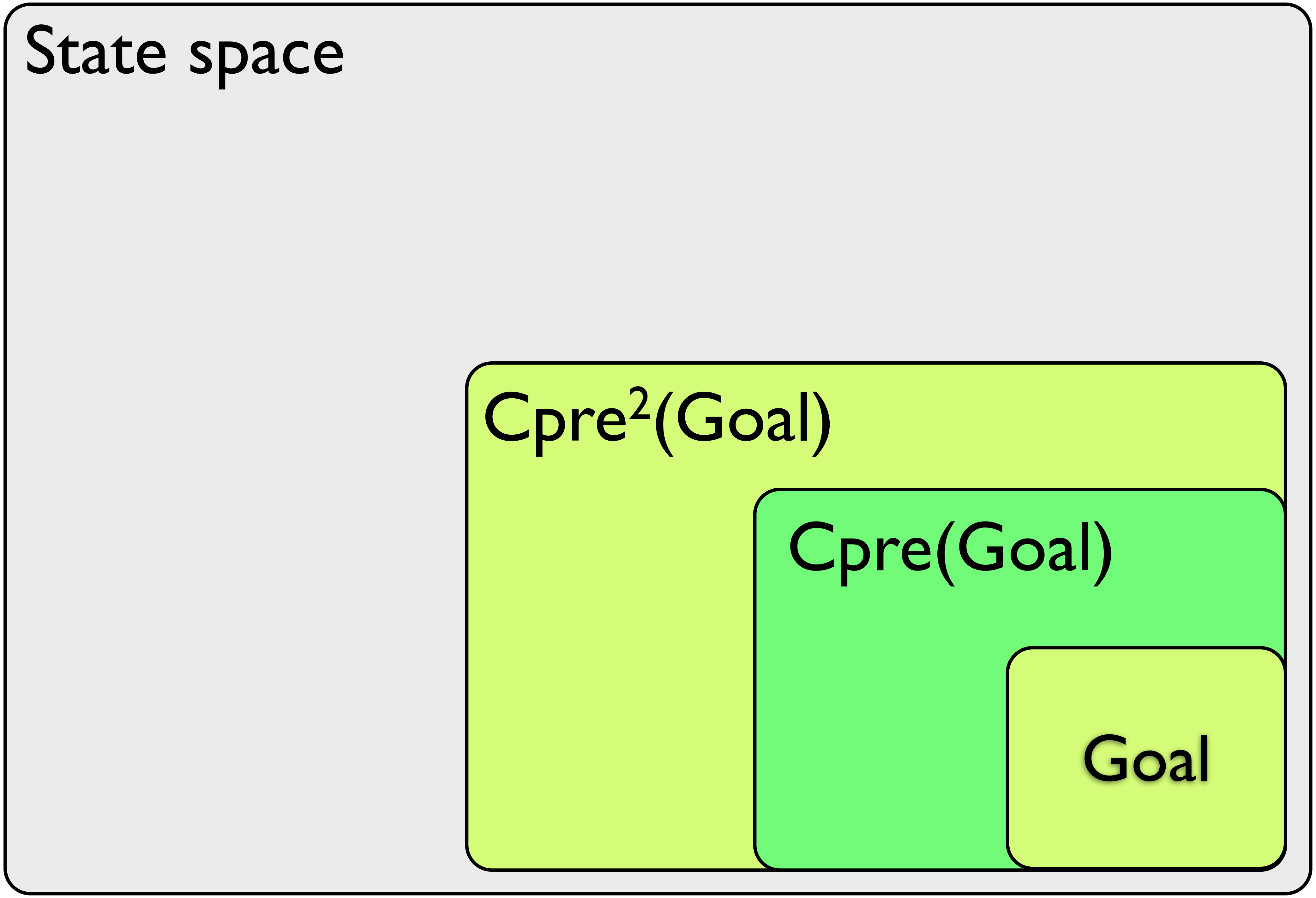
X is a set of positions

$$\mathsf{1CPre}_G(X) = \{q \in Q_1 \mid \exists q' : \delta(q,q') \wedge q' \in X\} \cup \{q \in Q_2 \mid \forall q' : \delta(q,q') : q' \in X\}$$

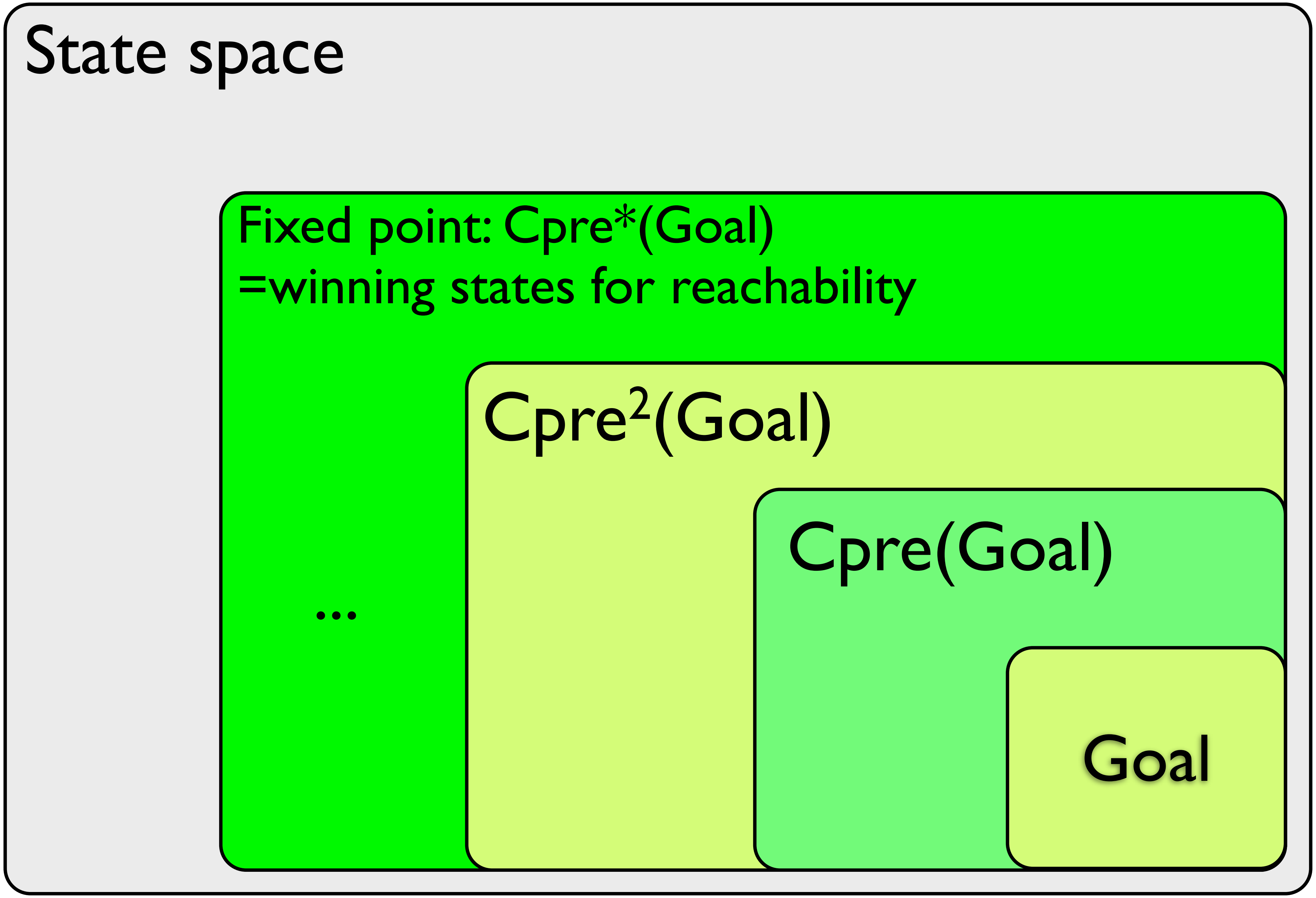
Set of Player I positions where she has a choice of successor that lies in *X*

Set of Player II positions where all her choices for successors lie in *X*

Reachability objective: Goal
What are the winning states for player I ?



Reachability objective: Goal
What are the winning states for player I ?

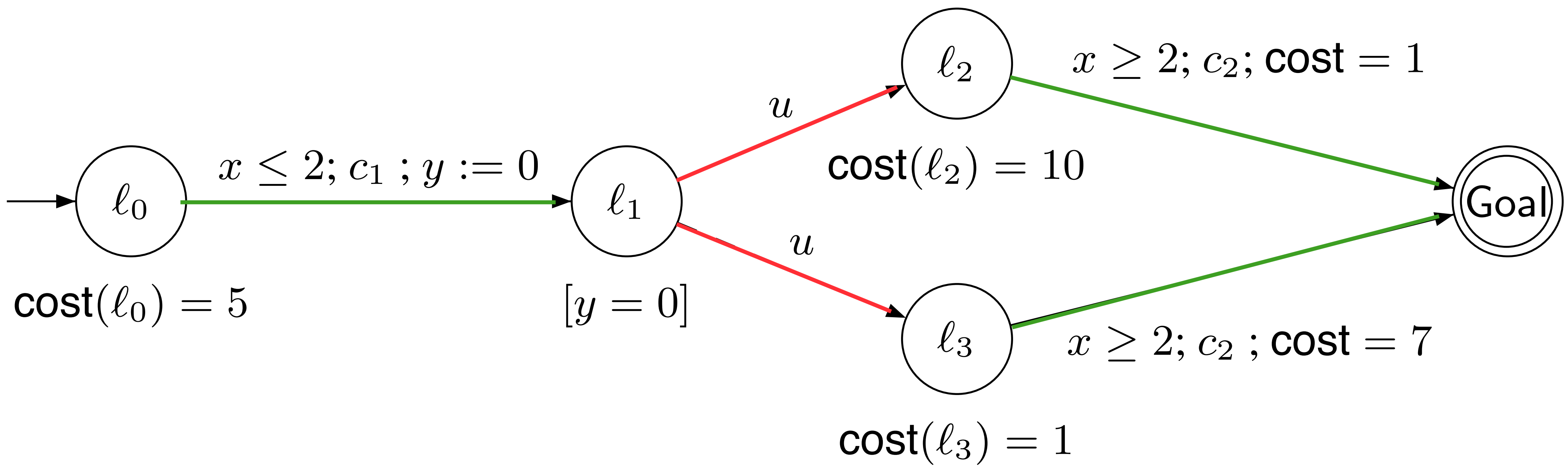


What is on the menu ?

- Premilinaries and motivations
- Two-player games - Fixed point algorithms
- Games on Weighted Timed Automata
- Symbolic semi-algorithm
- Undecidability
- Decidable subcases
- Conclusion - Bibliography

Games on WTA

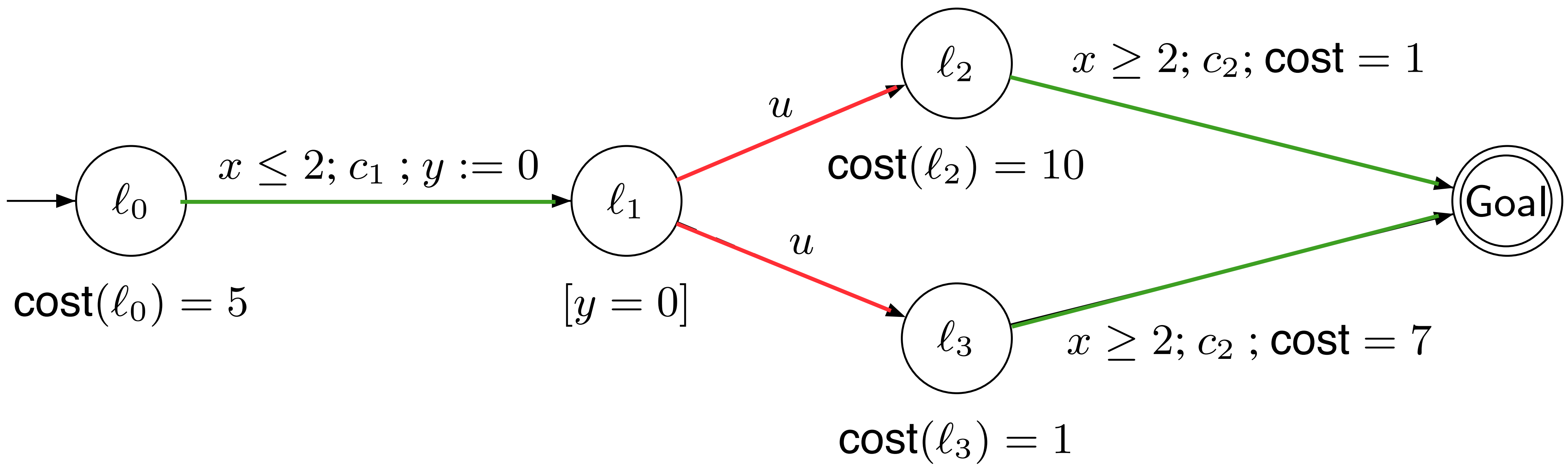
[Alur et al, 2004]
[Boyer et al, 2004]



To play games on WTA, we partition the transitions into:

controllable and uncontrollable

Games on WTA



State of the game =
location + clock values + **accumulated cost**

Games on WTA

A game on a WTA is played as follows: at any state q=(*l*,*v*,*c*)

Player 1 chooses a controllable action c and time *t* such that

$$q \xrightarrow{t} q' \xrightarrow{c} q_1$$

then Player 2 chooses :

- ➢ either to wait for *t* and to play *c*, and the game proceeds to state q_1,
- ➢ or to play at $t' \leq t$ an uncontrollable action *u* such that

$$q \xrightarrow{t'} q'' \xrightarrow{u} q_2$$

and the game proceeds to q_2.

Cost based strategies

Reachability objectives: A run (a play) is winning if it reaches a location labelled by "Goal".

A Player I (cost based) strategy is a function

$$\lambda : Q \times R^{+} \rightarrow \Sigma_{c} \times R^{+}$$

Given a state q and a strategy λ, we define $Outcome(q,\lambda)$ as the set of runs that can be obtained when Player I plays according to λ.

The strategy λ is winning from a state q if all runs of $Outcome(q,\lambda)$ are winning.

Cost (weight) associated to a strategy
Optimal cost

The cost of a run $\rho = q_1 \xrightarrow{t_1} \xrightarrow{e_1} \ldots q_n \xrightarrow{t_n} \xrightarrow{e_n} q_{n+1}$ is

$$W(\rho) = \sum_{i=1}^{n} W_L(l_i) \cdot t_i + \sum_{i=1}^{n} W_\delta(e_i)$$

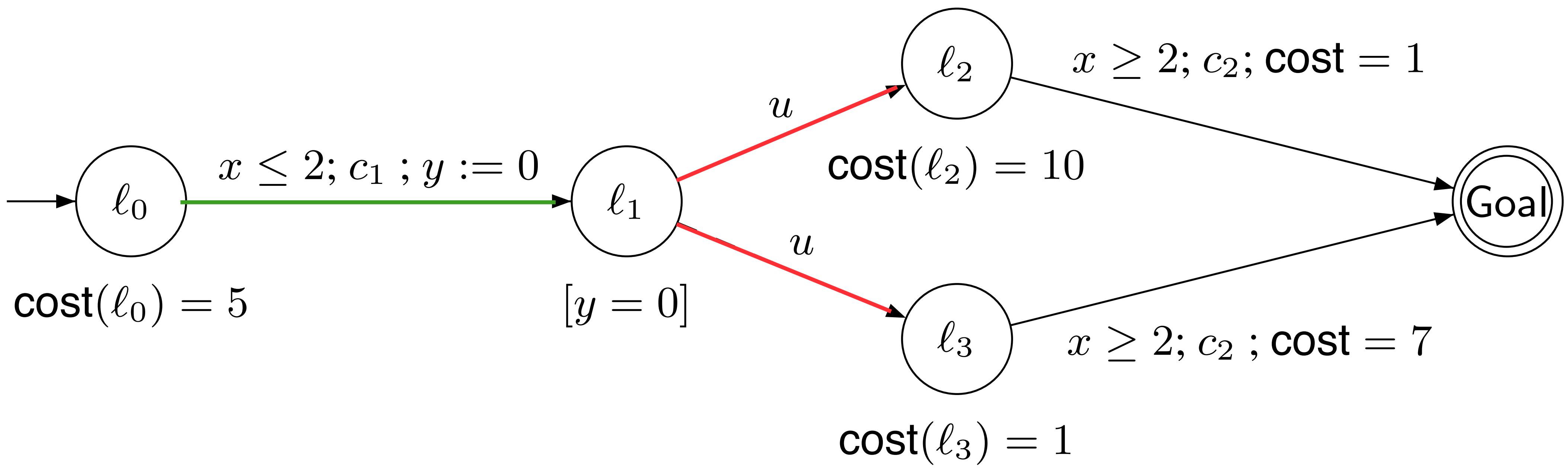
The cost associated with a winning strategy λ and a state *q* is defined by

$$\mathsf{Cost}(q, \lambda) = \sup\{W(\rho) \mid \rho \in \mathsf{Outcome}(q, \lambda)\}.$$

Given a state *q*, the optimal cost is given by

$$\mathsf{OptCost}(q) = \inf\{\mathsf{Cost}(q, \lambda) \mid \lambda \text{ is a winning strategy}\}.$$

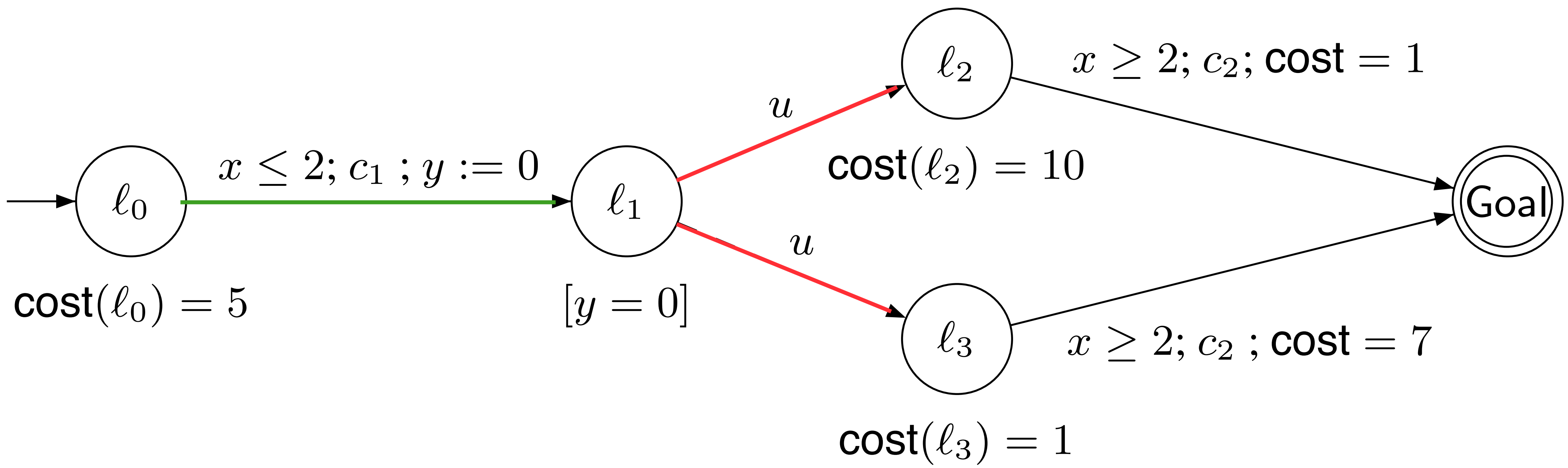
Optimal cost problem



Optimal Game Reachability Problem :

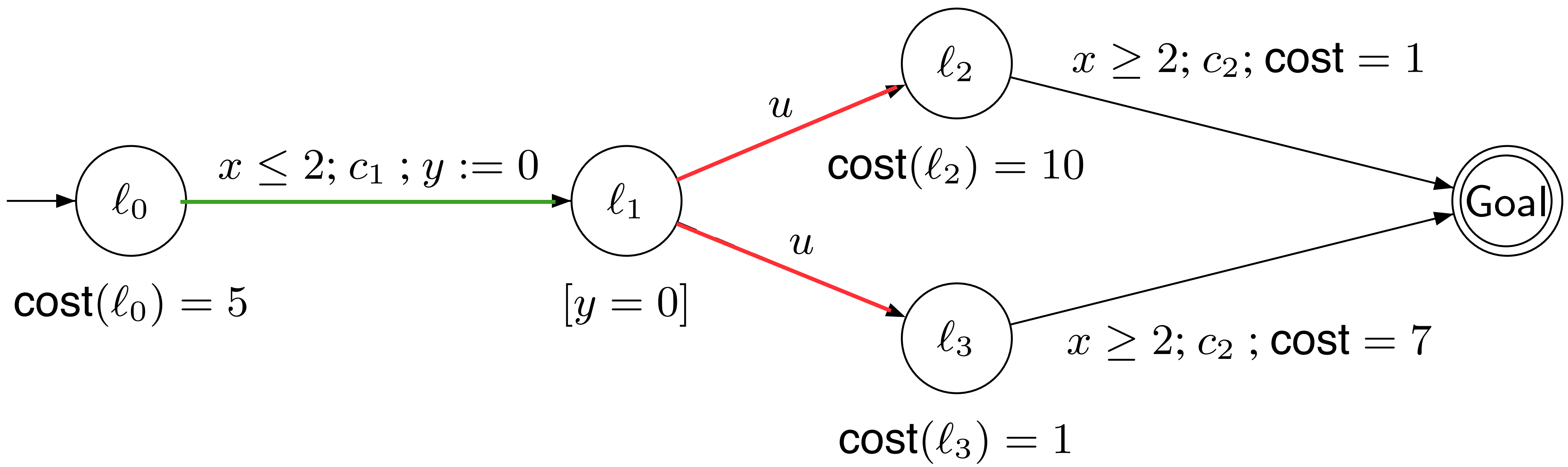
Given a WTA A, a state ***(l,v)*** and an positive integer ***c***, decide if there exists a winning Player I strategy **λ** from ***(l,v)*** such that ***Cost(q,λ)*** ≤ ***c***.

Optimal cost problem



What is the optimal cost that Player I can ensure ?

Optimal cost problem

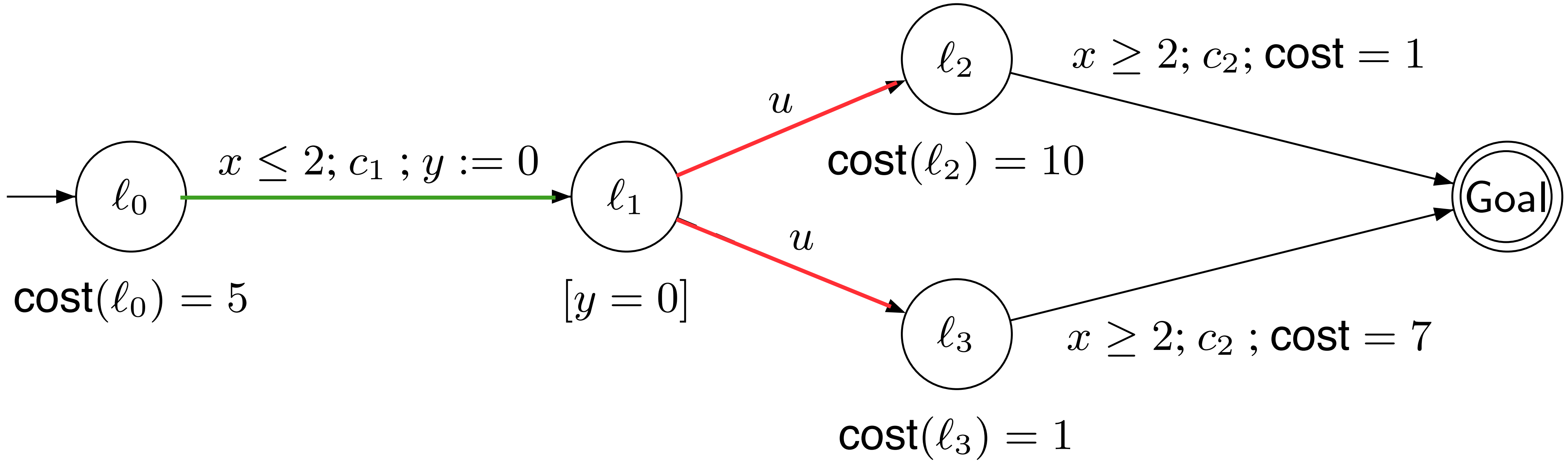


Player I's choice

$$\text{Min}_t(\text{Max}(5t + 10(2 - t) + 1, 5t + (2 - t) + 7))$$

Player II's choice

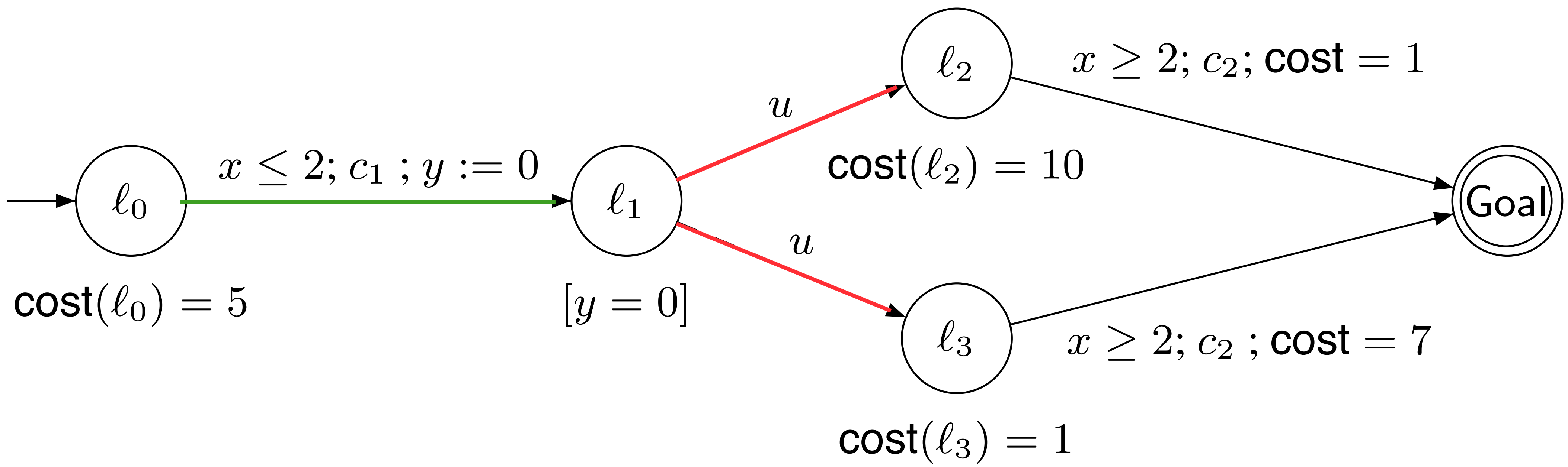
Optimal cost problem



$$\mathsf{Min}_t(\mathsf{Max}(5t + 10(2 - t) + 1, 5t + (2 - t) + 7))$$

$$\text{Which is when } t = \frac{3}{4} \text{ and the cost is } 14\frac{3}{4}$$

Optimal cost problem



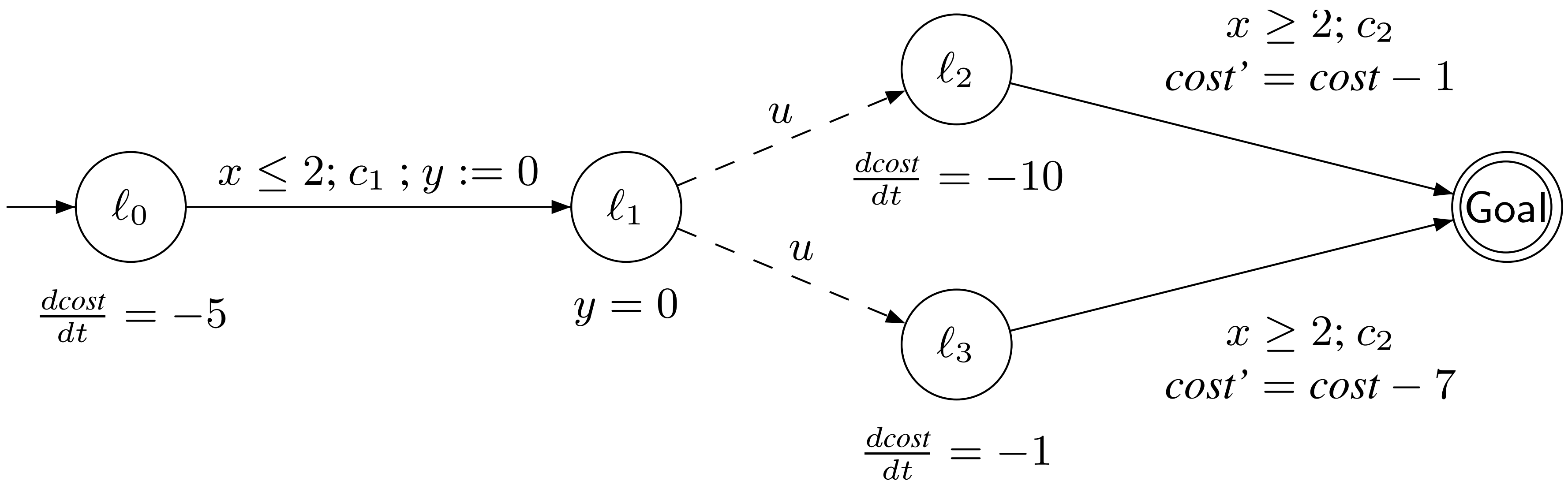
So, optimal moves are taken on rational points (and not only on integer points).

What is on the menu ?

- Premilinaries and motivations
- Two-player games - Fixed point algorithms
- Games on Weighted Timed Automata
- Symbolic semi-algorithm
- Undecidability
- Decidable subcases
- Conclusion - Bibliography

Symbolic Analysis

- Solving such a game can be seen as solving a hybrid automata game (add a variable w which models a "credit"), so states are of the form (*q*,*v*,*w*)



Symbolic Analysis

- Solving such a game can be seen as **solving a hybrid automata game** (add a variable *w* which models a **“credit”**), so states are of the form (*q,v,w*)

- We can define (as usual) a ***CPre* operator**, see [BCFL04], then we can try compute ***CPre*(Goal,w≥0)***

- ***(q,v,w) ∈ CPre*(Goal,w≥0)***

  iff

  Player I has a winning strategy of cost bounded by *w* in (*q,v*)

CPre operator

Let S be a set of triples (l,v,c). The ***controllable predecessor*** of S, CPre(S), is the set of triples (l',v',c') such that:

$\exists t \geq 0$: $\exists (l',v') -\sigma_c\rightarrow (l,v)$:

- $c = c' - W(\sigma_c) - t \times W(l') \wedge (l,v,c) \in S$
- $\forall t' \leq t$, $\forall (l,v)$:

  $(l',v') -\sigma_u\rightarrow (l,v) \wedge c = c' - W(\sigma_u) - t' \times W(l')$:

  $(l,v,c) \in S$

This operator transforms polyhedral sets into polyhedral sets.

Symbolic Analysis

- Solving such a game can be seen as **solving a hybrid automata game** (add a variable *w* which models a **“credit”**), so states are of the form (q,v,w)
- We can define (as usual) a ***CPre* operator**, see [BCFL04], then we can try compute ***CPre*(Goal,w≥0)***
- $$(q,v,w) \in CPre^*(Goal, w \geq 0)$$

  iff

  Player I has a winning strategy of cost bounded by *w* in (q,v)
- This fixpoint computation is guaranteed to terminate when “***every cycles in the region graph of the automaton has a cost bounded away from zero***” see [BCFL04]. The authors conjectured that this property was **not** necessary for terminaison.

What is on the menu ?

- Premilinaries and motivations
- Two-player games - Fixed point algorithms
- Games on Weighted Timed Automata
- Symbolic semi-algorithm
- Undecidability
- Decidable subcases
- Conclusion - Bibliography

CPre(Goal,w≥0)* is not Computable

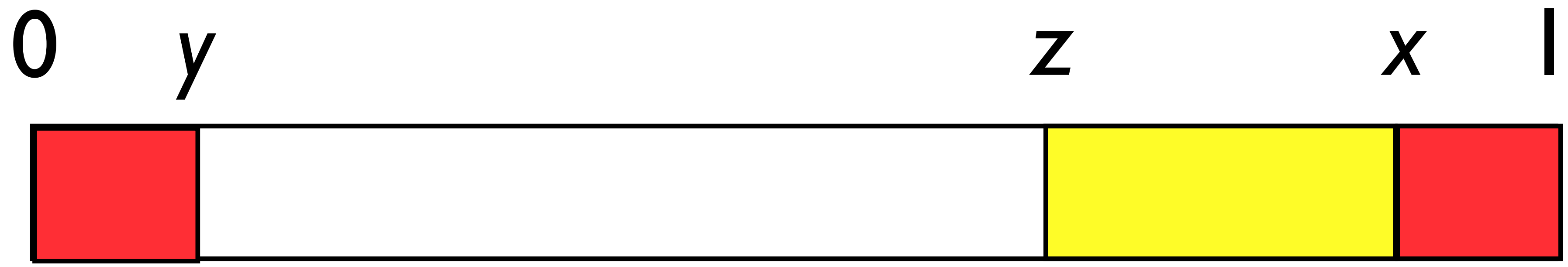
- Given a 2CM machine *M*, we can construct a WTA A such that

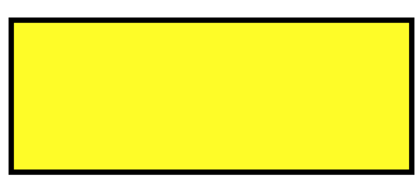
  Player I has a strategy to reach Goal at a cost bounded by 1
  iff
  M is halting

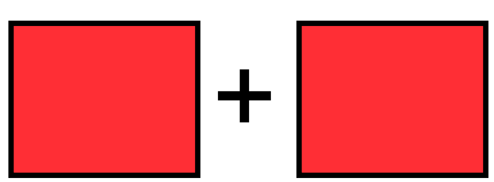
- Proof idea:
  - Player I simulates the 2CM computation
    - if *M* halts then the game ends in *Goal* with a cost $w \leq 1$
  - if he does not : Player II can force the game to *Goal* at a cost $w > 1$

Encoding the values of counters

We use three clocks *x*,*y*,*z* to encode *C*=*n*

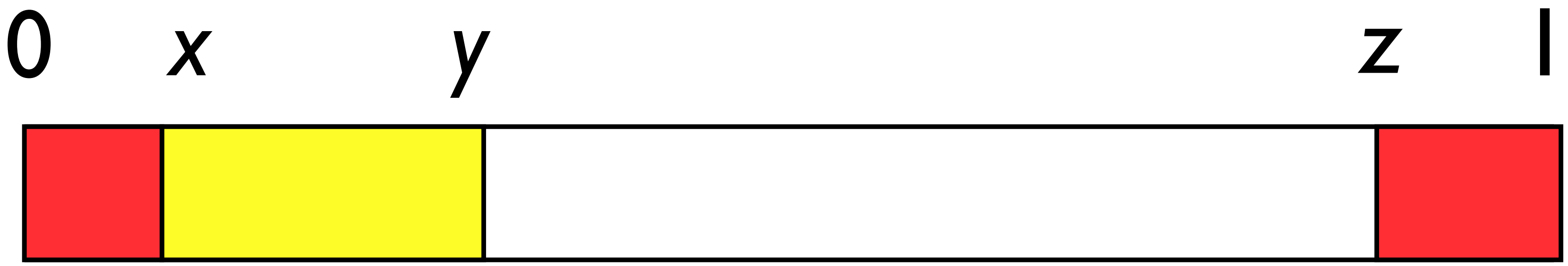


 $= x - z = \frac{1}{2^{n+1}}$

 $= 1 - x + y = \frac{1}{2^{n+1}}$

Encoding the values of counters

When time evolves ...



$$+ \;= 1 - z + x = \frac{1}{2^{n+1}}$$

$$= y - x = \frac{1}{2^{n+1}}$$

Leaving the values of counter unchanged

Widget I

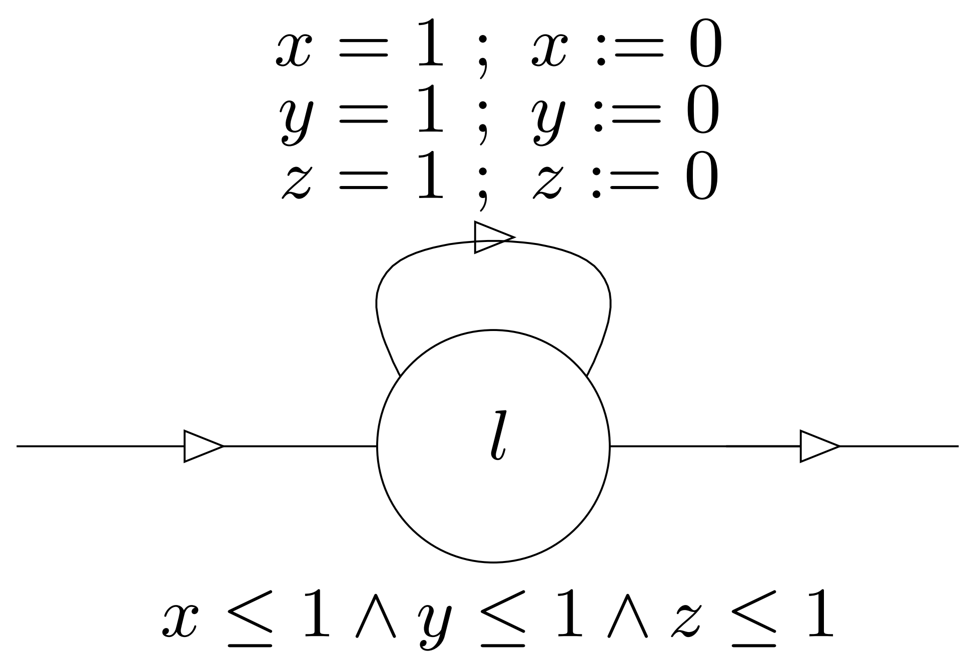
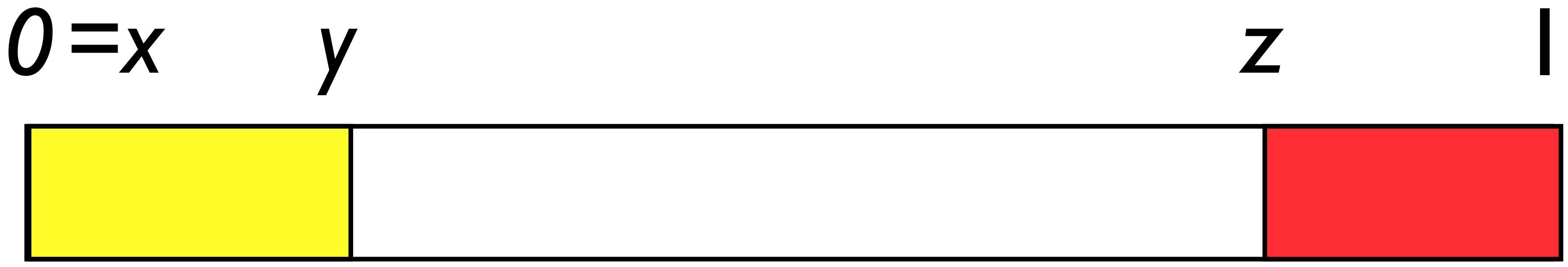
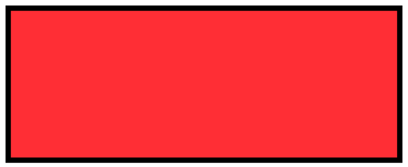


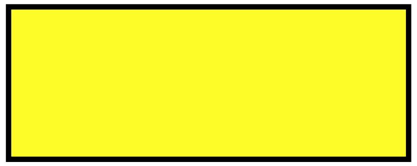
Fig. 4. Widget to let the value of a counter unchanged.

Encoding the values of counters

Normal form... when $x=0$



 $= 1 - z = \dfrac{1}{2^{n+1}}$

 $= y = \dfrac{1}{2^{n+1}}$

Encoding of a counter in normal form

Widget II

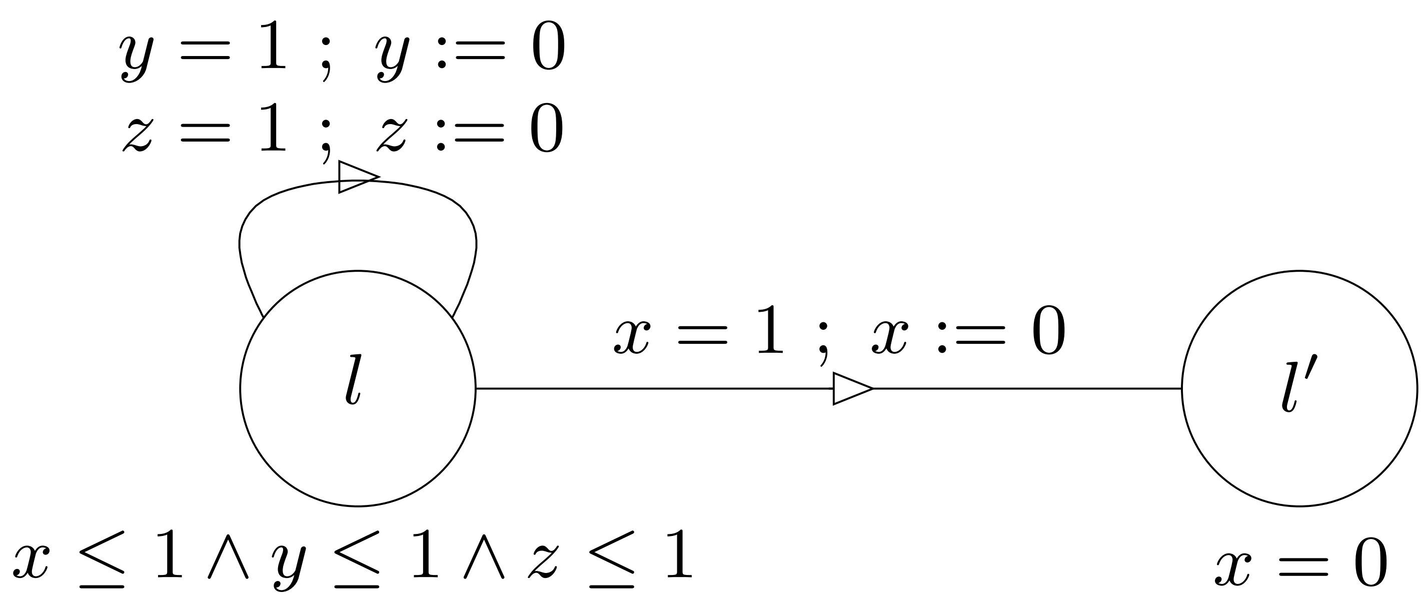


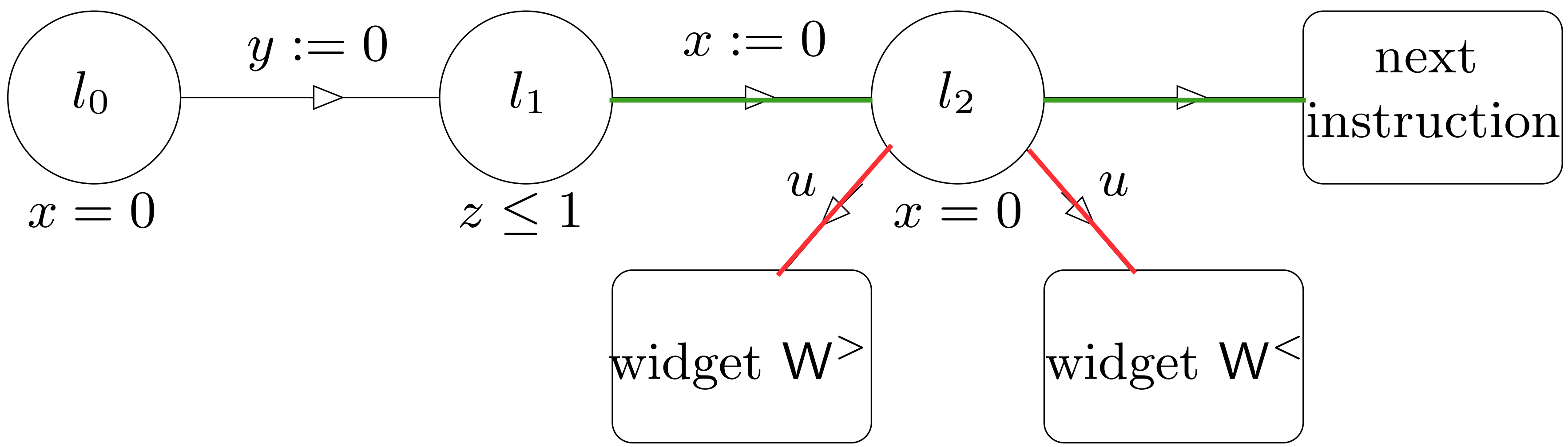
Fig. 5. Widget to put a counter encoding in normal form.

Simplifications

- when modifying the value of counter C_1, the value of counter C_2 is maintained by the widget I (and vice versa);
- before modifying the value of a counter, it is first put in normal form using widget II

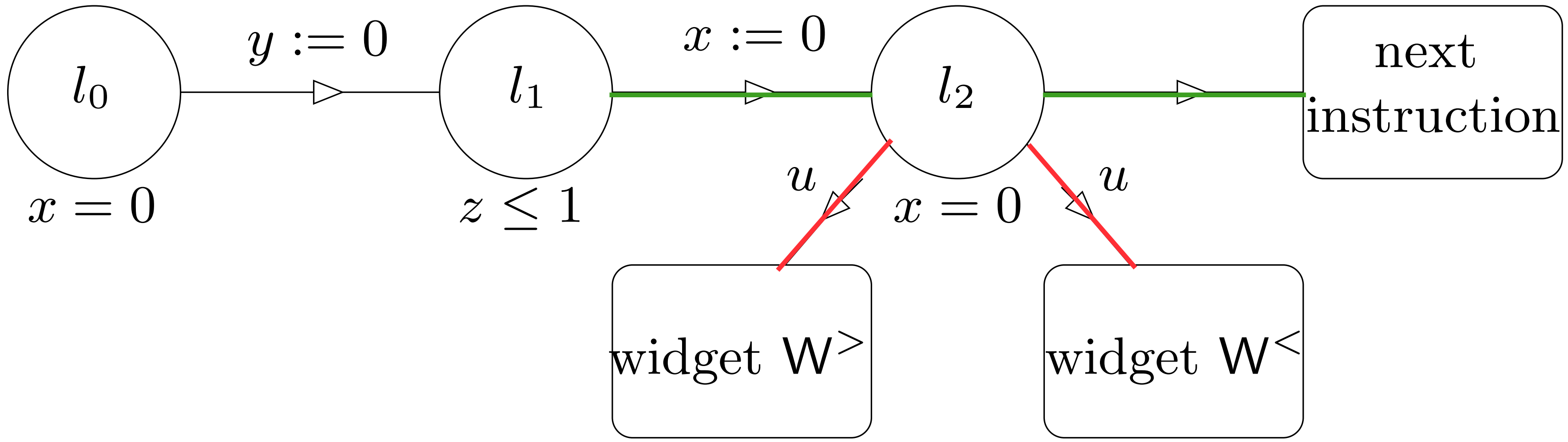
Simulating an increment

$$k : c_i := c_i + 1$$



Simulating an increment

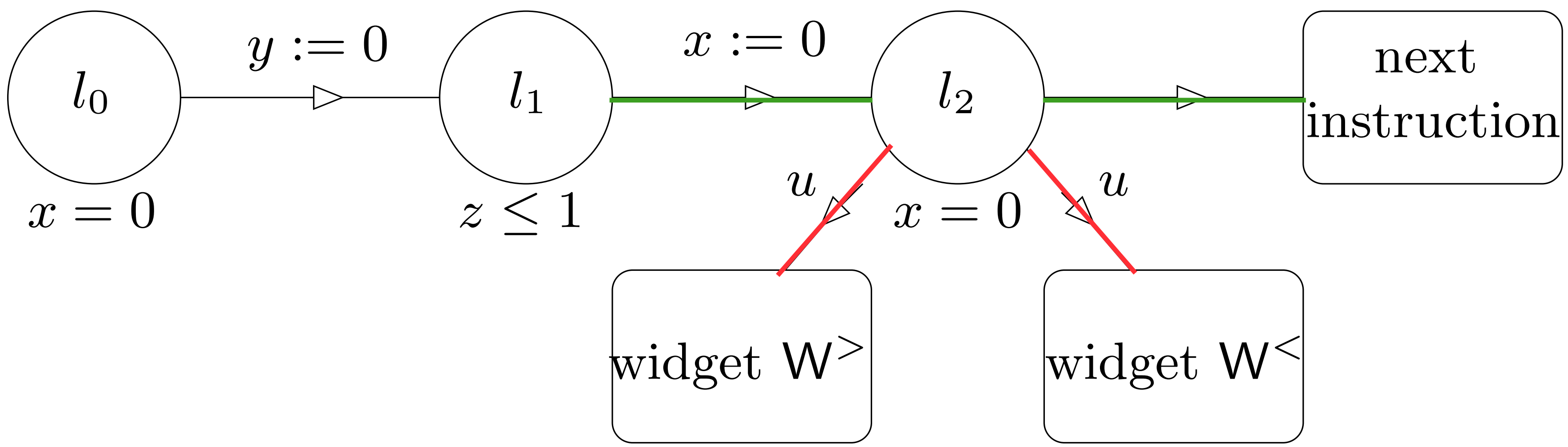
$$k : c_i := c_i + 1$$



Player 1 should reset *x* at the right moment in order to obtain the encoding for $C_i = n+1$ when entering l_2

Simulating an increment

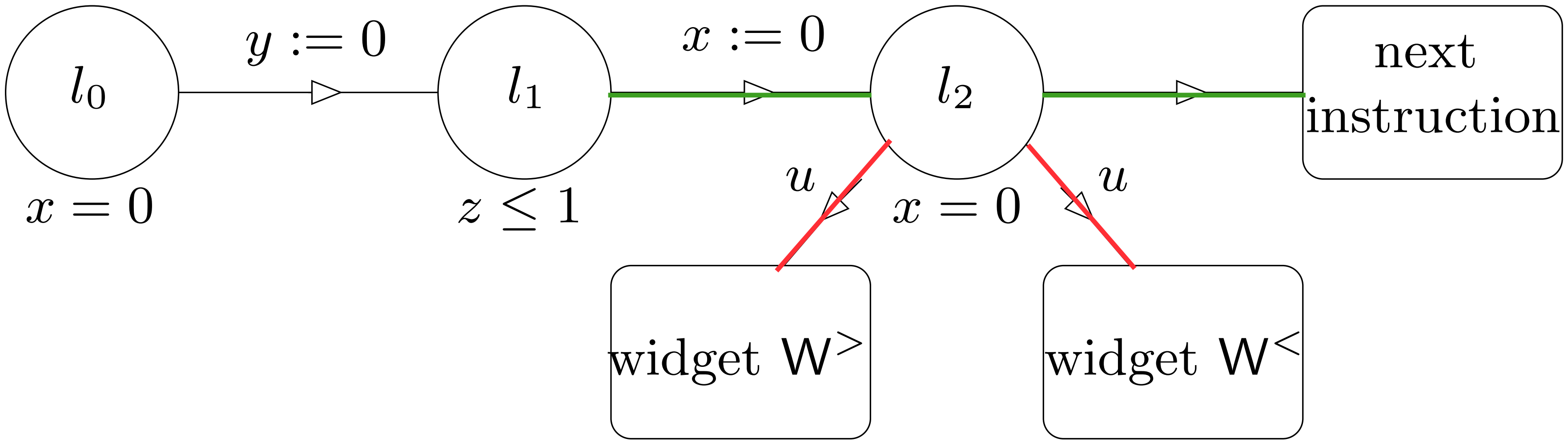
$$k : c_i := c_i + 1$$



Player II will verify that Player I has reset *x* at the right moment

Simulating an increment

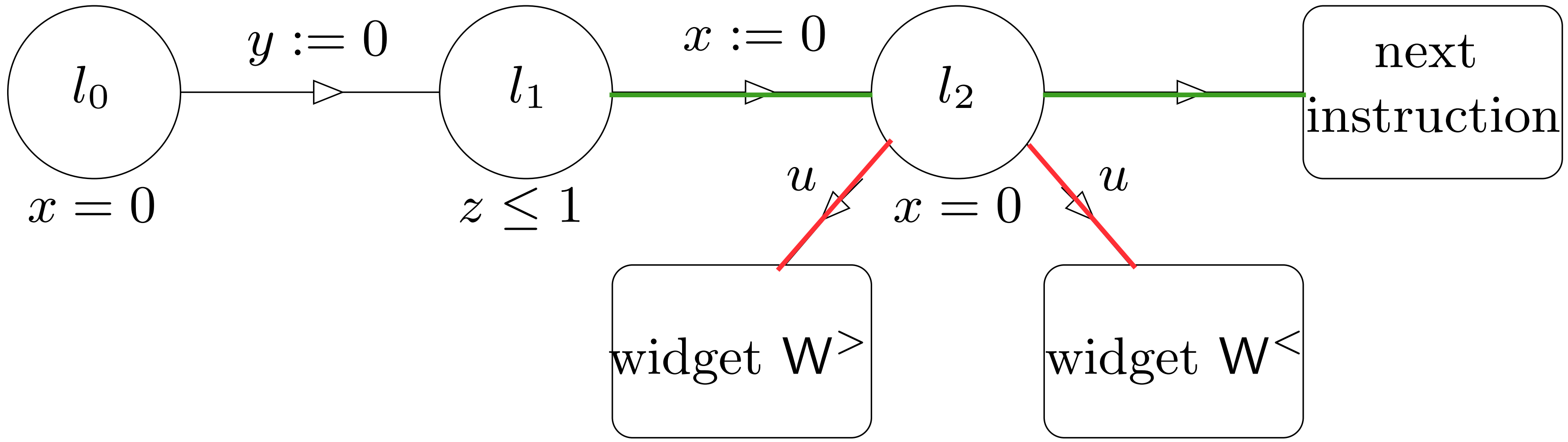
$$k : c_i := c_i + 1$$



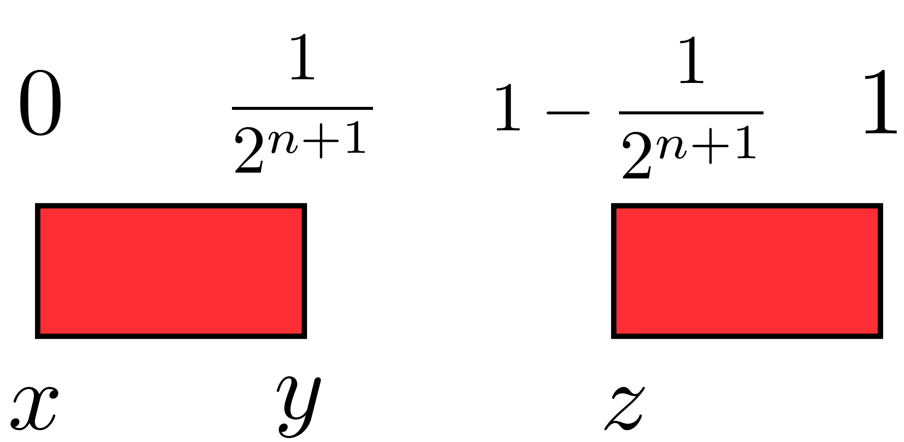
At what time should Player I reset x ?

Simulating an increment

$$k : c_i := c_i + 1$$

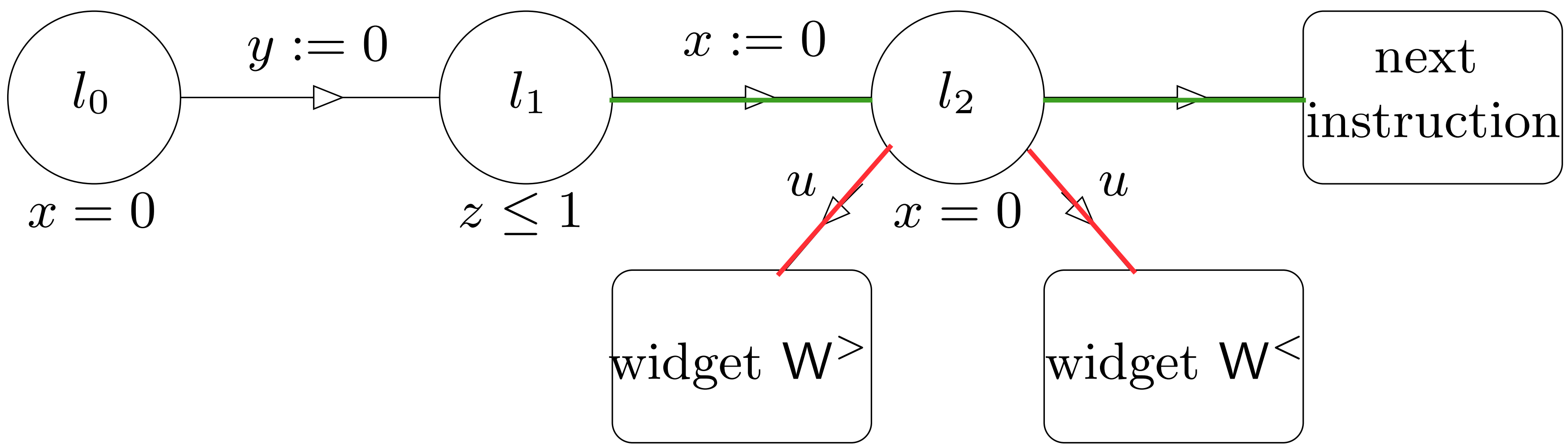


in l_0

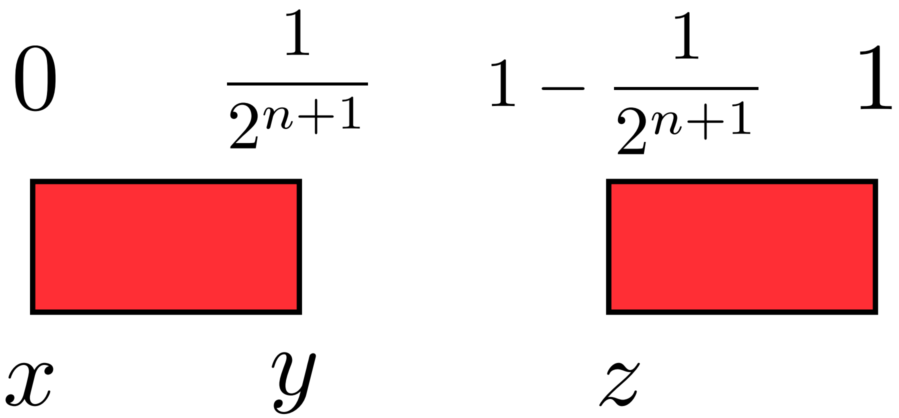


Simulating an increment

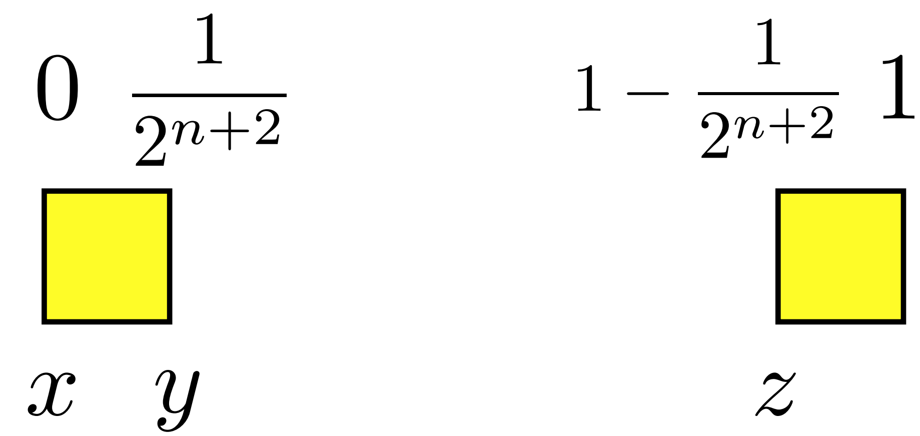
$$k : c_i := c_i + 1$$



in l_0

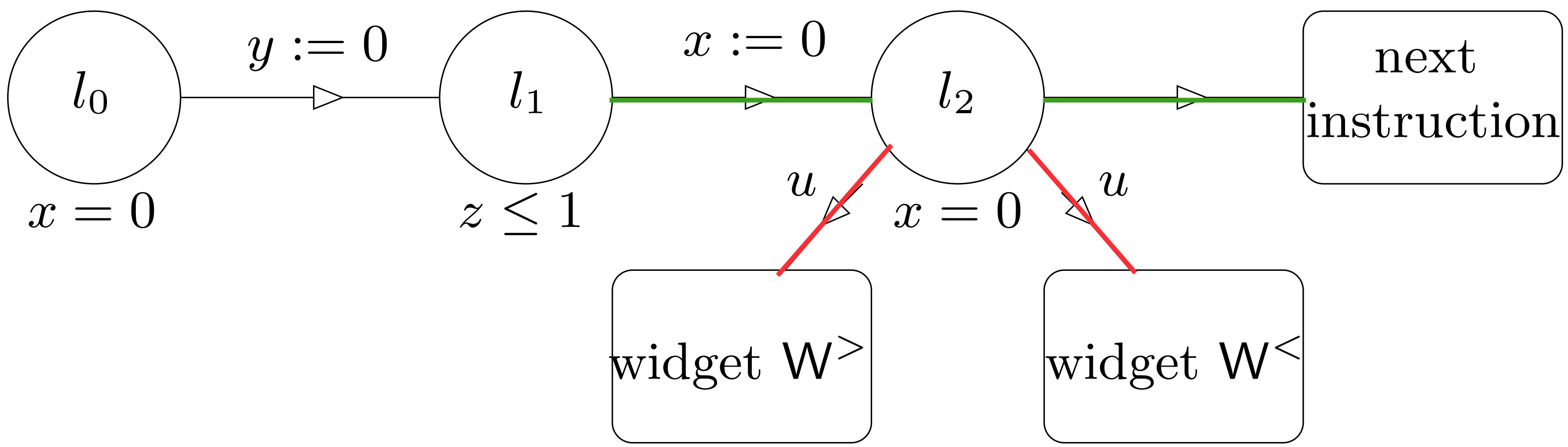


in l_2

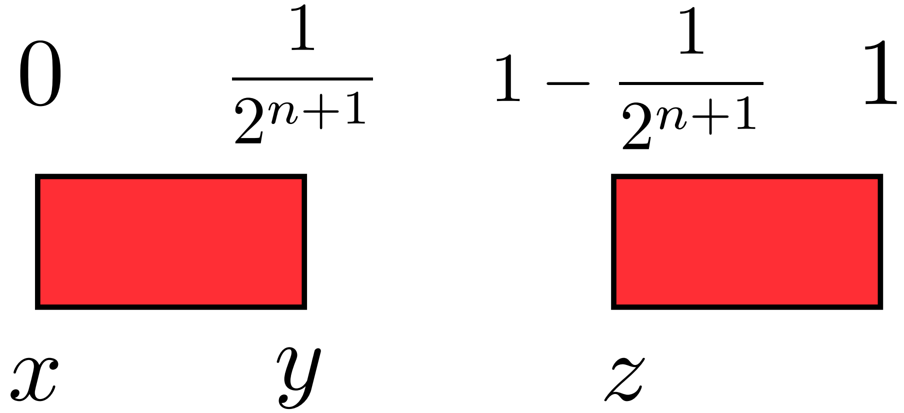


Simulating an increment

$$k : c_i := c_i + 1$$

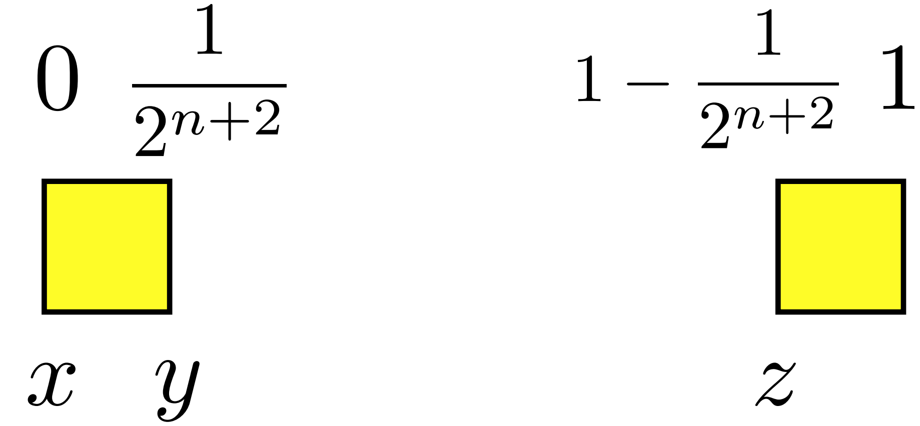


in l_0



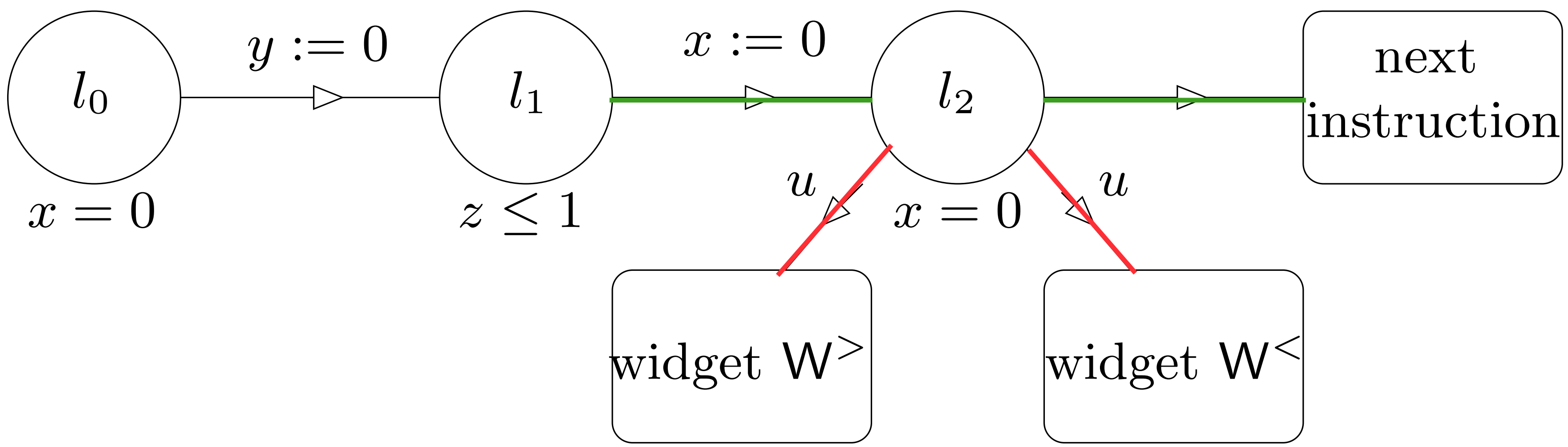
reset x after $\frac{1}{2^{n+2}}$ in l_1

in l_2



Simulating an increment

$$k : c_i := c_i + 1$$



in l_0

$0 \quad \frac{1}{2^{n+1}} \quad 1 - \frac{1}{2^{n+1}} \quad 1$

$x \quad y \quad z$

reset x after $\frac{1}{2^{n+2}}$ in l_1

We note *t* the time spent in l_1

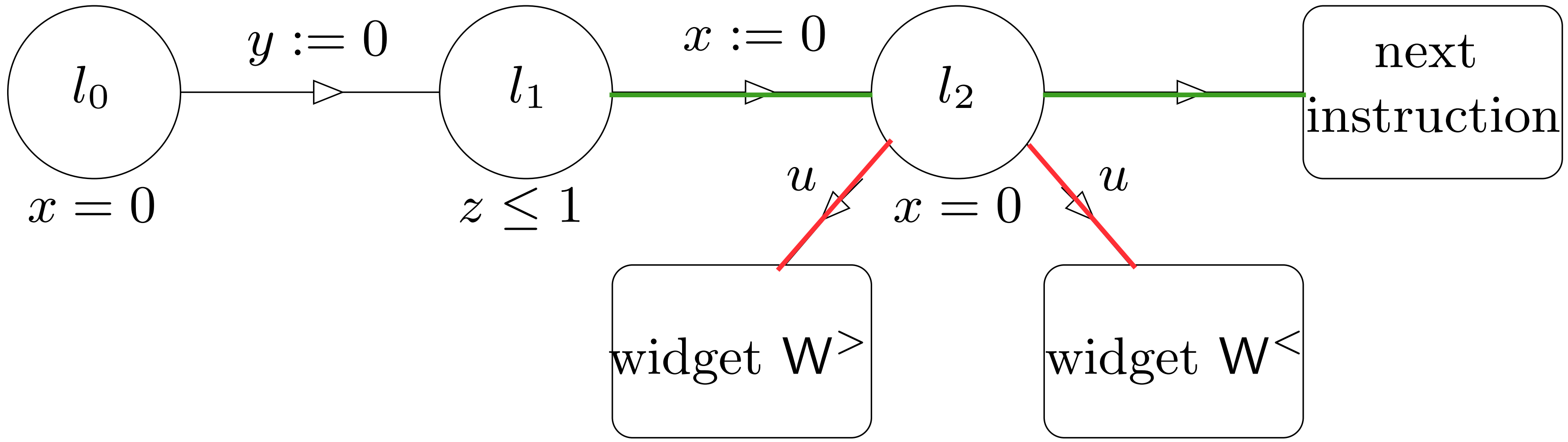
in l_2

$0 \quad \frac{1}{2^{n+2}} \quad 1 - \frac{1}{2^{n+2}} \quad 1$

$x \quad y \quad z$

Simulating an increment

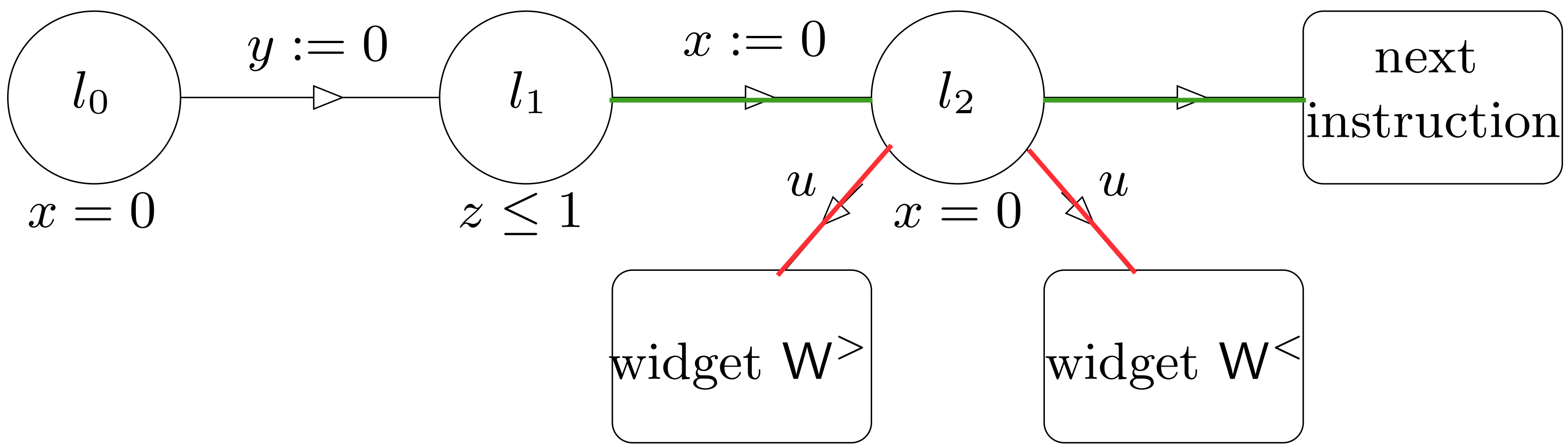
$$k : c_i := c_i + 1$$



How can Player II verify that
Player I has faithfully simulated the increment ?

Simulating an increment

$$k : c_i := c_i + 1$$

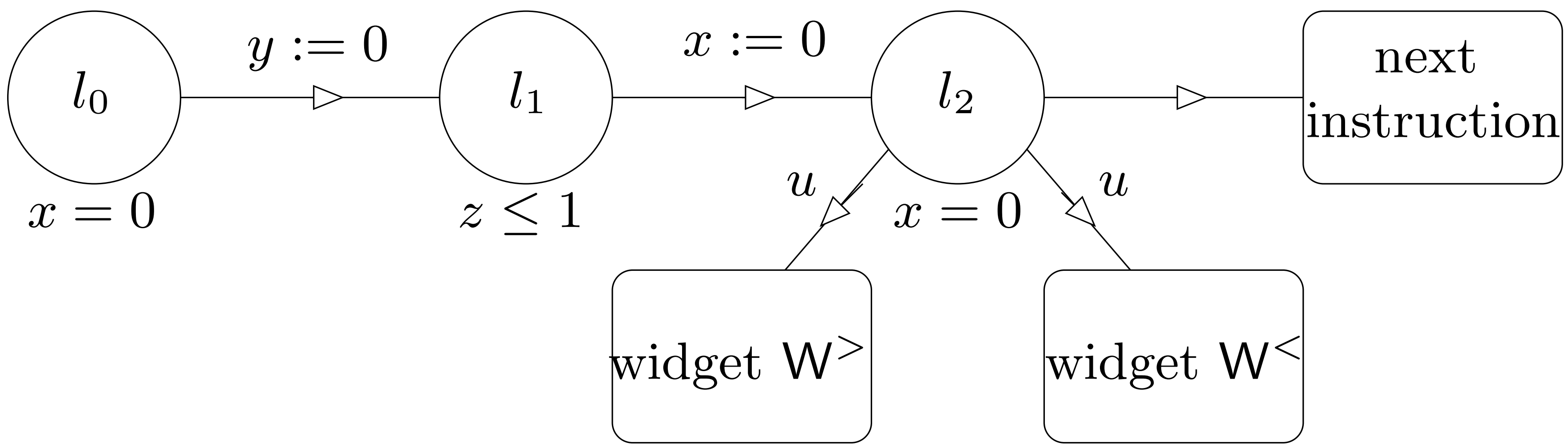


In l_2, we have that

$$t = \frac{1}{2^{n+2}} \Leftrightarrow y + z = 1.$$

Simulating an increment

$$k : c_i := c_i + 1$$



If *x*+*y*>*l*, Player II moves the game to Widget W>
if *x*+*y*<*l*, Player II moves the game to Widget W<

Widget W$^>$

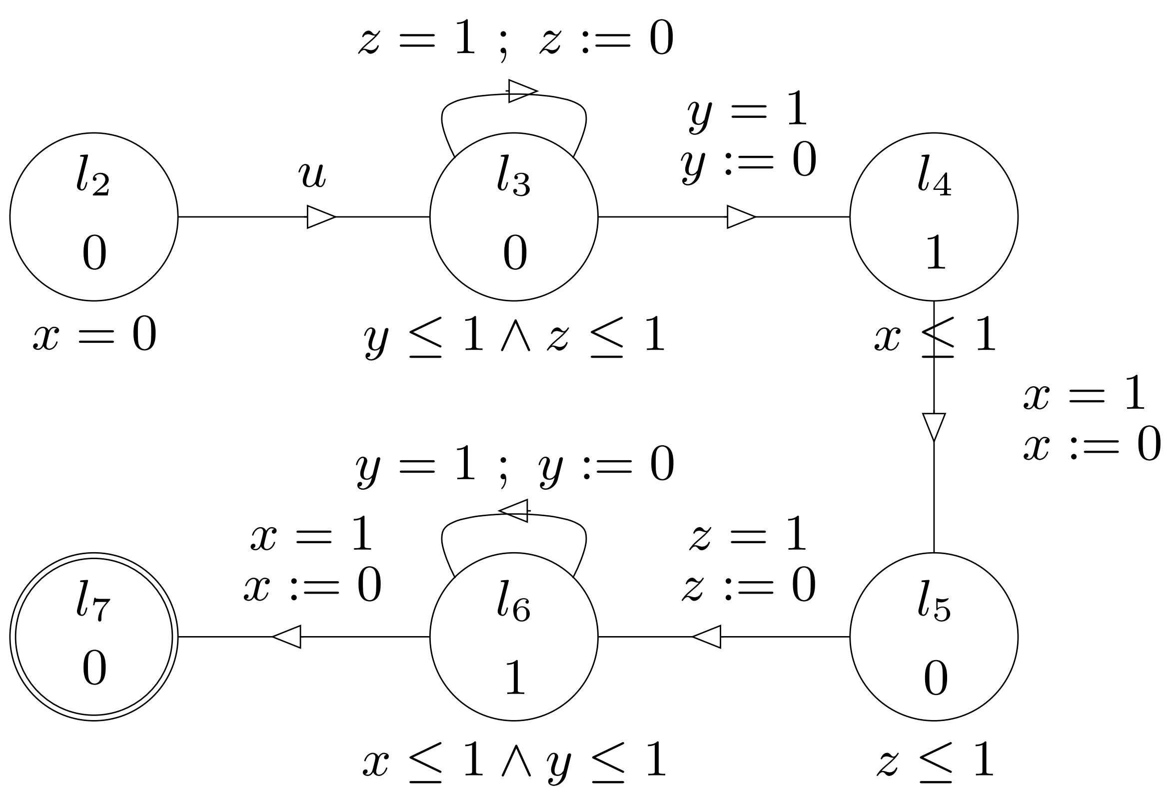


Fig. 8. Widget W$^>$.

$$y + z > 1 \Leftrightarrow W(\rho) > 1.$$

Widget $W^{>}$

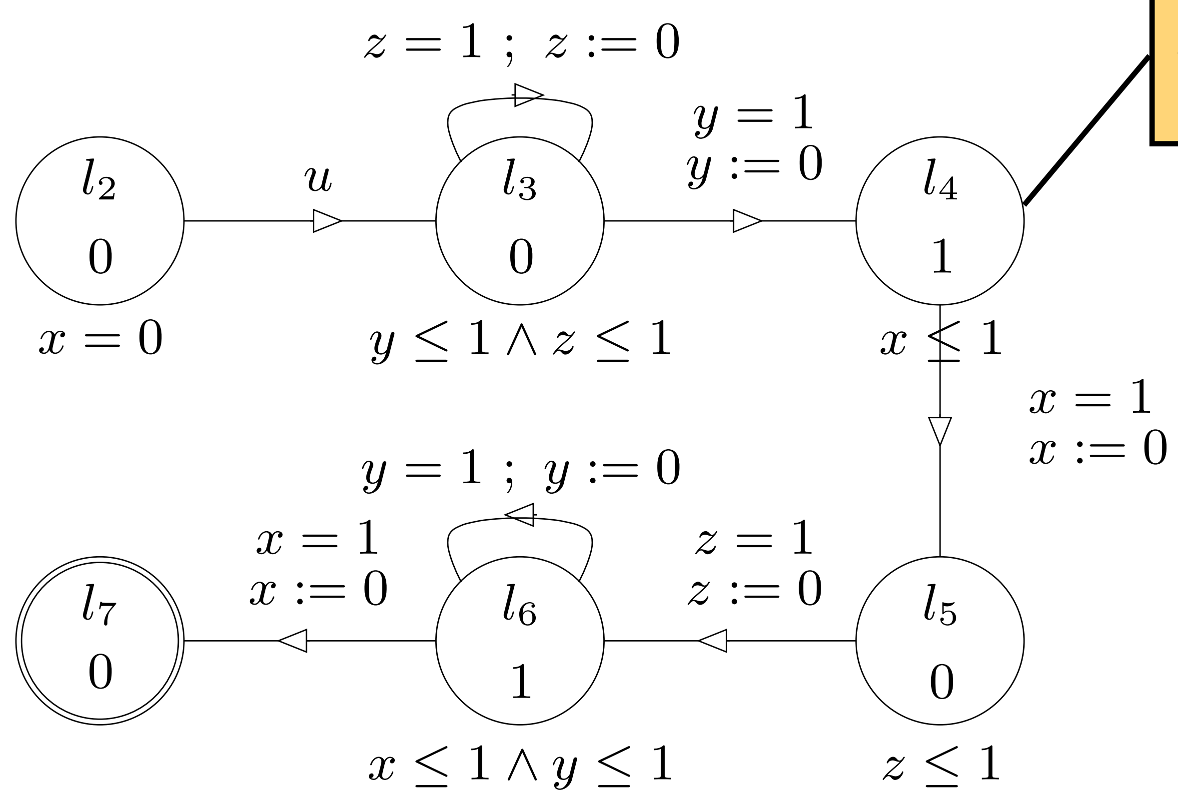


Fig. 8. Widget $W^{>}$.

Compute the value of *y* at *l*2

$$y + z > 1 \Leftrightarrow W(\rho) > 1.$$

Widget W$^>$

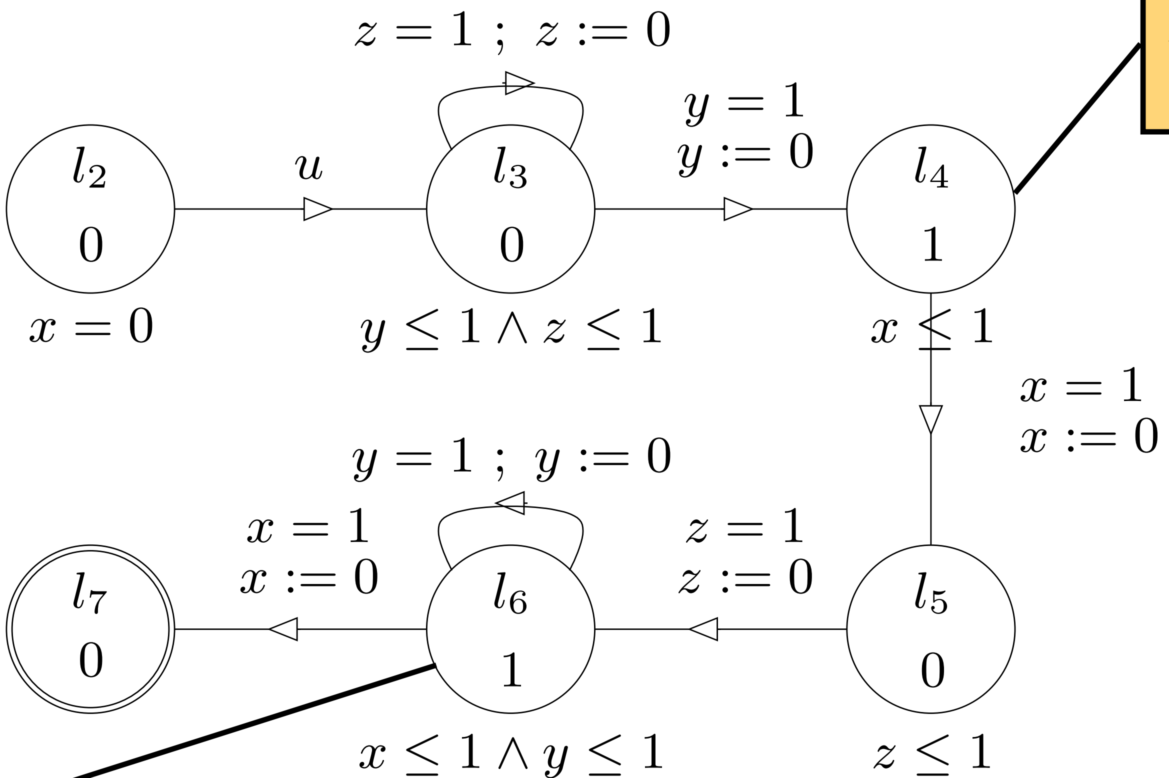


Fig. 8. Widget W$^>$.

Compute the value of *y* at *l*2

Compute the value of *z* at *l*2

$$y + z > 1 \Leftrightarrow W(\rho) > 1.$$

Widget W$^<$

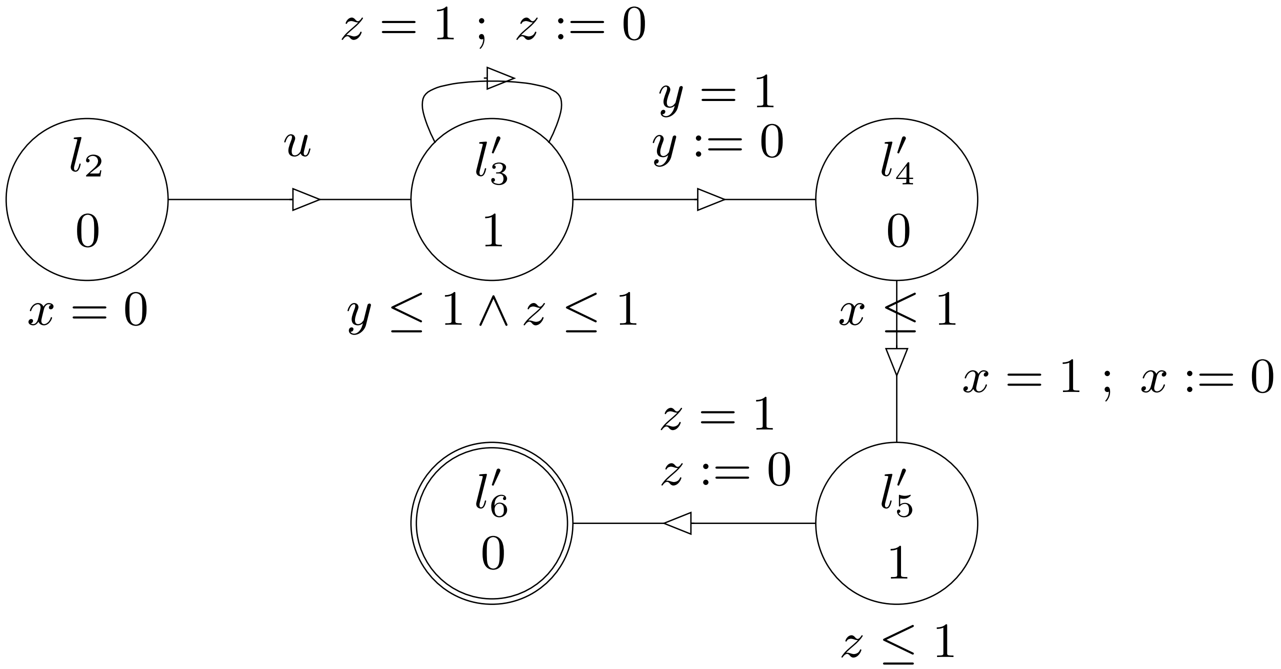
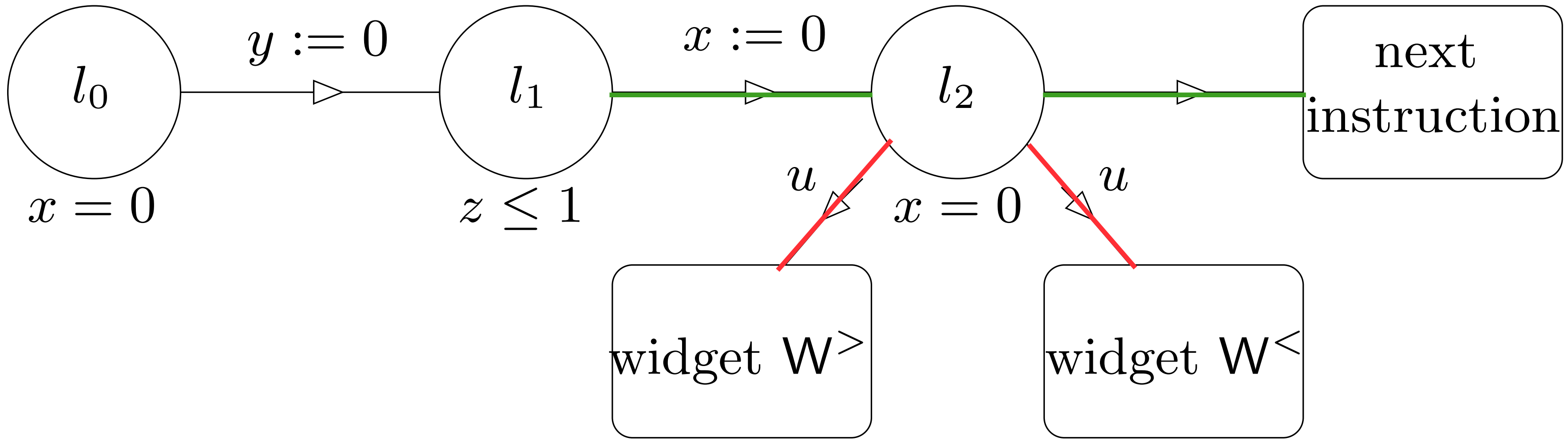


Fig. 13. Widget $\mathsf{W}^<$.

$$y + z < 1 \Leftrightarrow W(\rho') > 1.$$

Increment summary



Player I reset *x* to simulate the increment of C
If it does not do it faithfully, Player II force the game either to widget W> or W< and the game end with weight *w*>*1*, otherwise, the game proceeds to the next instruction.

Summary of the construction

- We construct from the widgets a game where :
  - if the 2CM is halting then Player I simulates faithfully the computation, then
    - either Player II let Player I play and the game end in an goal state with weight w=0
    - or Player II stops the game after an increment or a decrement and the game ends in a goal state with weight w=1
  - if the 2CM is not halting then Player I does not have a winning strategy:
    - either Player I simulates the 2CM faithfully, in that case, the game never reach a goal state
    - or Player I makes an error and Player II stops the game using W> or W< and the game reach a goal state with weight w>1

What is on the menu ?

- Premilinaries and motivations
- Two-player games - Fixed point algorithms
- Games on Weighted Timed Automata
- Symbolic semi-algorithm
- Undecidability
- Decidable subcases
- Conclusion - Bibliography

Decidables subcases

- **Time optimal reachability**: the strategy that minimizes time to target (all costs on locations are equal to 1) is computable (see [AM99,BHPR07]);
- One clock and costs 0 and d is decidable (see [BBR05] for details);

Termination for one clock, costs 0 and d

- There exists an infinite game-bisimulation quotient P

- Let P_1 and P_2 be two regions,

$$P_1 < P_2$$
$$\text{iff}$$
$$P_2 \downarrow x = P_2 \downarrow x \text{ and}$$
$$Min(P_1 \downarrow w) < Min(P_2 \downarrow w)$$

- $<$ is wqo : this ensures the termination

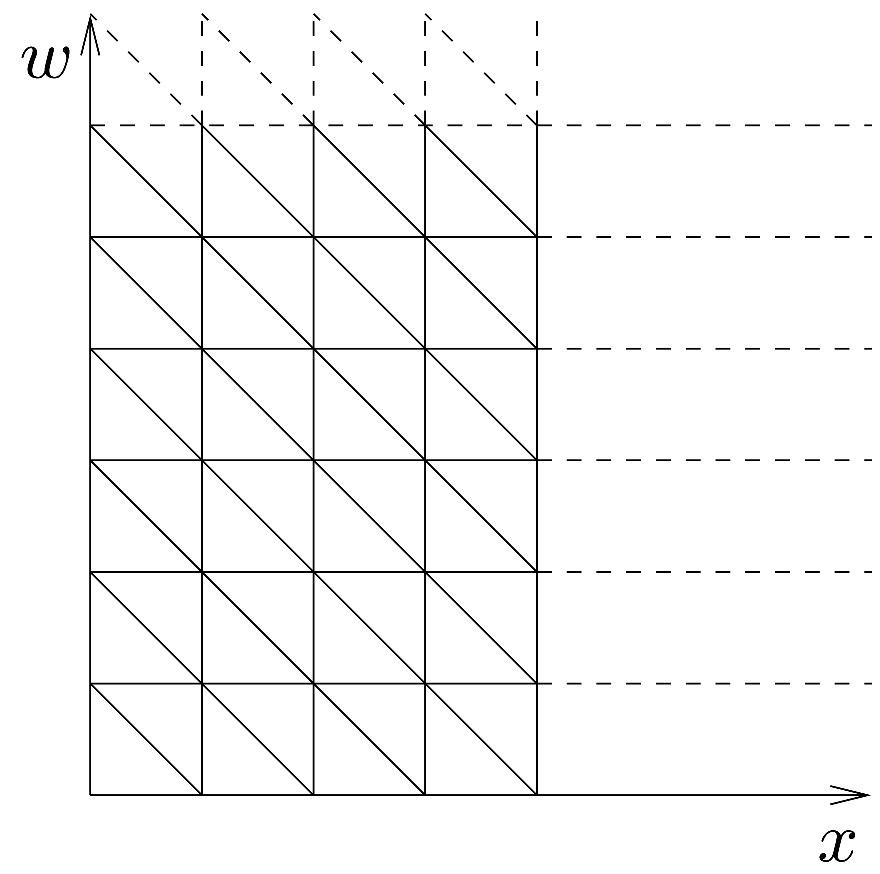


Fig. 11. The relation $\sim$ with $C = 4$.

Decidables subcases

- **Time optimal reachability**: the strategy that minimizes time to target (all costs on locations are equal to 1) is computable (see [AM99,BHPR07]);
- One clock and costs 0 and d is decidable (see [BBR05] for details);
- One clock and any costs, ε-optimality is decidable [BLMR06] ;

Conclusions

- Games on WTA are natural models for open embedded systems with resource constraints
- Positive results:
  - Optimal Reachability Game is decidable under the hypthesis of "strong non-zenoness of costs" [Bouyer et al, 2004]
  - Bounded case (play for *k* times) is decidable [Alur et al, 2004]
  - Optimal time reachability problem is decidable [Asarin et al, 1999, Brihaye et al, 2007]
  - One clock two costs [BBR05], One clock any costs [BLM07]
- Negative results:
  - The general problem is undecidable [BBR05, BLMR06]
- Open problem:
  - can we approximate optimal cost ?
  - can we develop a useful theory with discounting ?