,

One-clock synthesis problems

Sławomir Lasota[1], Mathieu Lehaut[1],
Julie Parreaux[2] and Radosław Piórkowski[3]

[1]University of Warsaw, Poland
[2]Université de Rennes, France
[3]Oxford University, UK

STACS'26

Reactive syntheses for timed systems

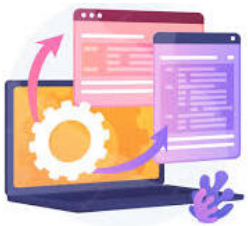
Build a real-time system correct by construction

Reactive syntheses for timed systems

Build a real-time system correct by construction

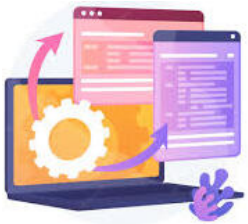
Reactive syntheses for timed systems

Build a real-time system correct by construction



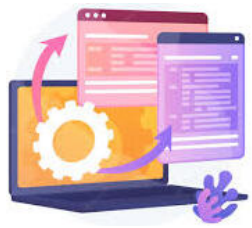
Reactive syntheses for timed systems

Build a real-time system correct by construction



Reactive syntheses for timed systems

Build a real-time system correct by construction



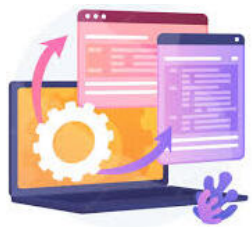
System

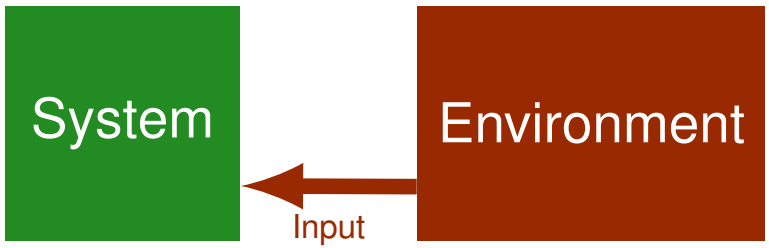
Environment

Reactive syntheses for timed systems

Build a real-time system correct by construction

Input: I^{ω}

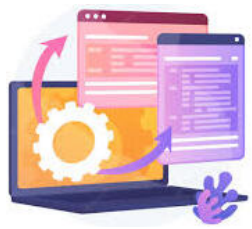


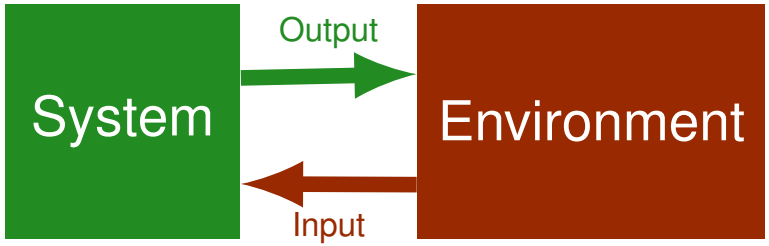


Reactive syntheses for timed systems

Build a real-time system correct by construction

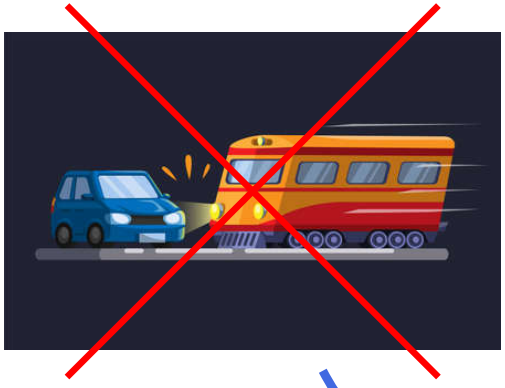
Output: O^ω Input: I^ω





Reactive syntheses for timed systems

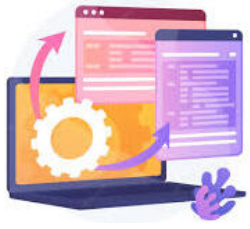
Build a real-time system correct by construction

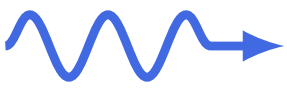


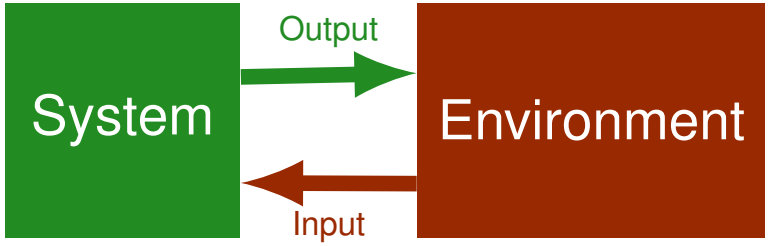
Interactions $\rightsquigarrow i_0 o_0 i_1 o_1 i_2 o_2 \cdots$

member ?

Output: O^ω Input: I^ω





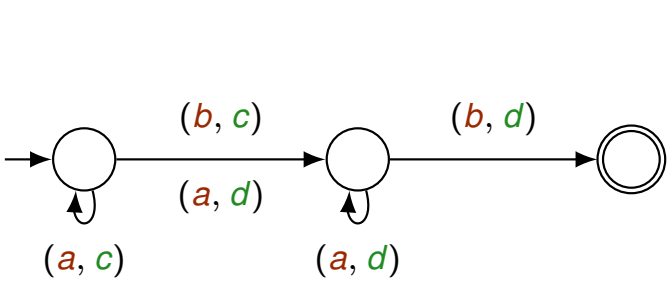


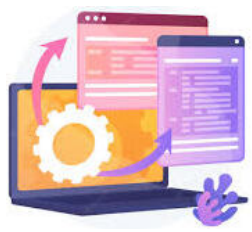
Reactive syntheses for timed systems

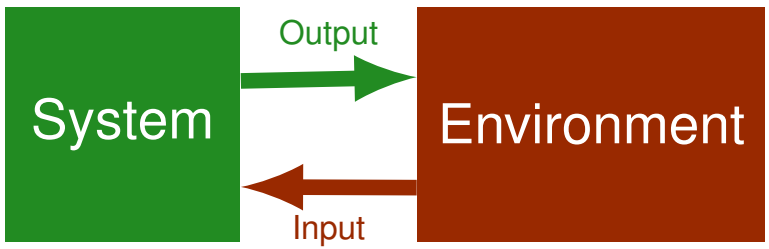
Build a real-time system correct by construction

Interactions $\rightsquigarrow i_0 o_0 i_1 o_1 i_2 o_2 \cdots$

Output: O^ω Input: I^ω





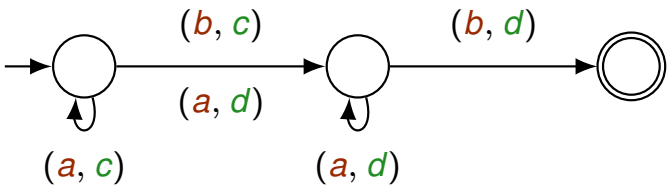


Reactive syntheses for timed systems

Build a real-time system correct by construction

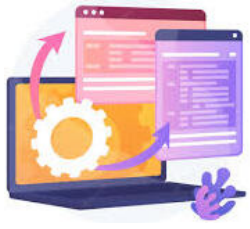
Interactions $\rightsquigarrow i_0 o_0 i_1 o_1 i_2 o_2 \cdots$

Output: O^ω Input: I^ω



member ?

Reactive synthesis

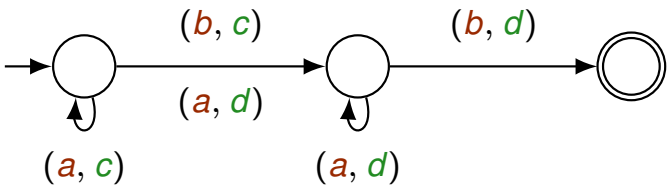


Reactive syntheses for timed systems

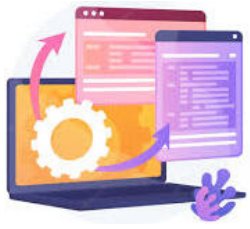
Build a real-time system correct by construction

Interactions $\rightsquigarrow i_0 o_0 i_1 o_1 i_2 o_2 \cdots$

Output: O^ω Input: I^ω

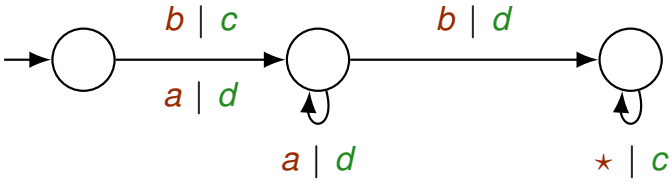


member ?



Reactive synthesis

Church synthesis

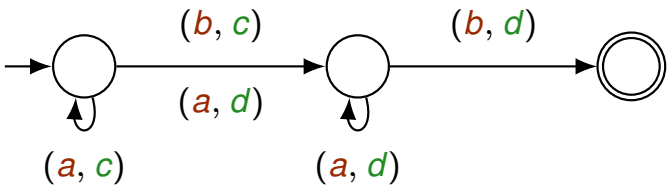


Reactive syntheses for timed systems

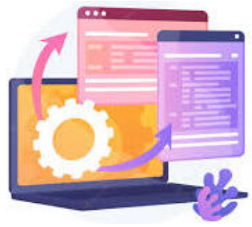
Build a real-time system correct by construction

Interactions $\rightsquigarrow i_0 o_0 i_1 o_1 i_2 o_2 \cdots$

Output: O^ω Input: I^ω

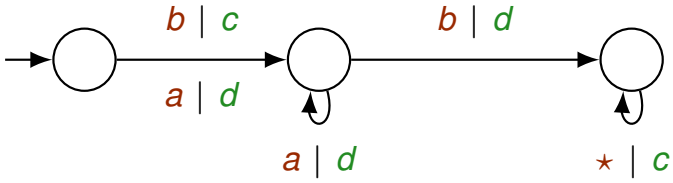


member ?



Reactive synthesis

Church synthesis



Reactive syntheses problems

EXPTIME-complete for non-deterministic parity automata.

Solving sequential conditions finite-state strategies, J.R. Büchi and L.H. Landweber, 1969, AMS

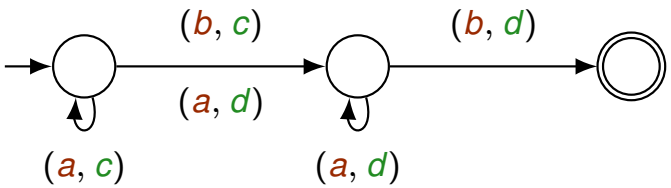
On the Synthesis of a Reactive Module, A. Pnueli and R. Rosner, 1989, POPL

Reactive syntheses for timed systems

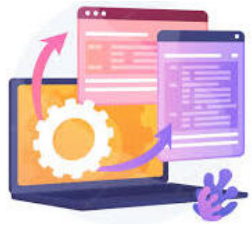
Build a real-time system correct by construction

Interactions $\rightsquigarrow i_0 o_0 i_1 o_1 i_2 o_2 \cdots$

Output: O^ω Input: I^ω

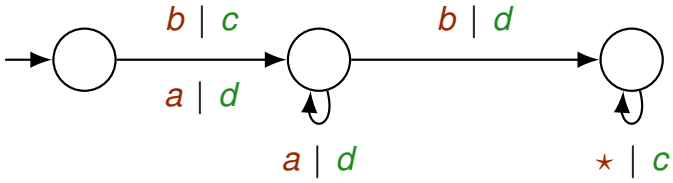


member ?



Reactive synthesis

Church synthesis



Reactive syntheses problems

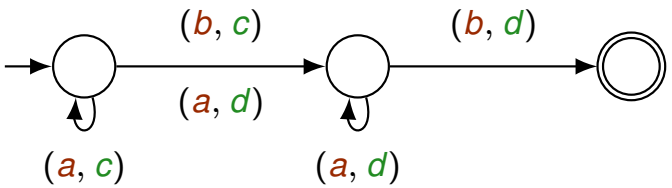
EXPTIME-complete for non-deterministic parity automata.

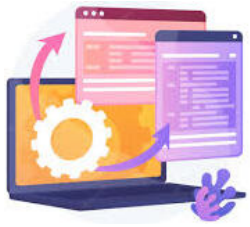
Reactive syntheses for timed systems

Build a real-time system correct by construction

Interactions $\rightsquigarrow i_0 o_0 t_0 i_1 o_1 t_1 i_2 o_2 t_2 \cdots$

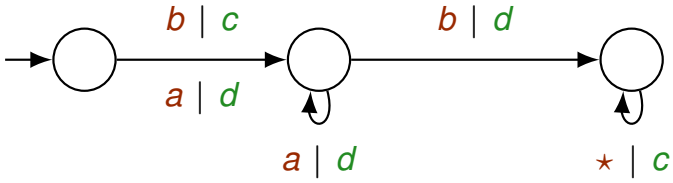
Output: O^ω  Input: I^ω





Reactive synthesis

Church synthesis



Reactive syntheses problems

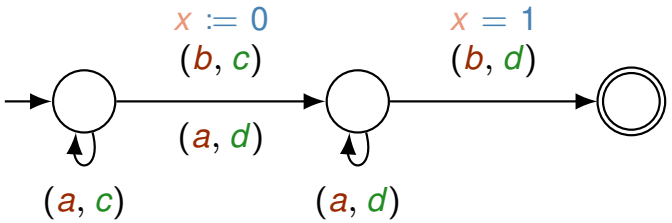
EXPTIME-complete for non-deterministic parity automata.

Reactive syntheses for timed systems

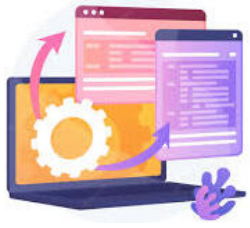
Build a real-time system correct by construction

Interactions $\rightsquigarrow i_0 o_0 t_0 i_1 o_1 t_1 i_2 o_2 t_2 \cdots$

Output: O^ω  Input: I^ω

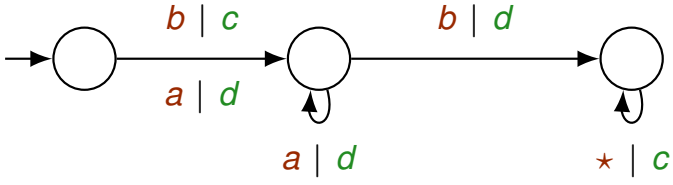


member ?



Reactive synthesis

Church synthesis



Reactive syntheses problems

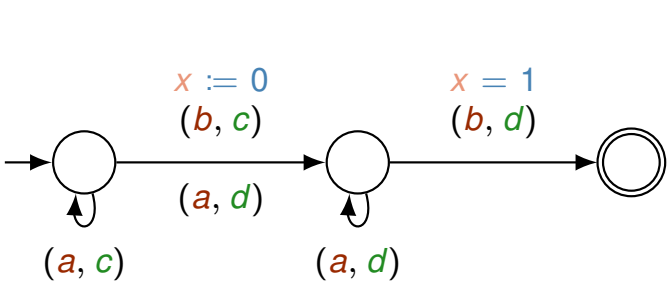
EXPTIME-complete for non-deterministic parity automata.

A theory of timed automata, R. Alur D. Dill, TCS 1994

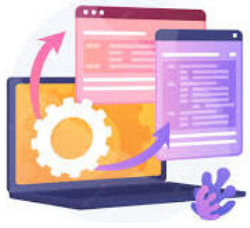
Reactive syntheses for timed systems

Build a real-time system correct by construction

Interactions $\rightsquigarrow i_0 o_0 t_0 i_1 o_1 t_1 i_2 o_2 t_2 \cdots$

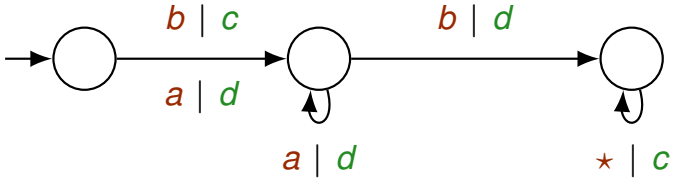


Output: O^ω  Input: I^ω



Reactive synthesis

Church synthesis



Reactive syntheses problems

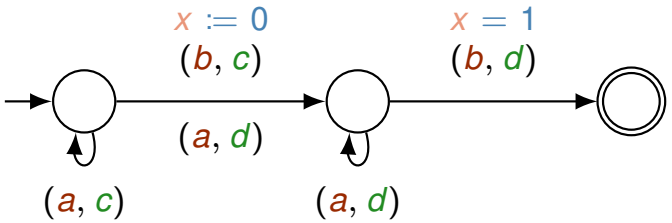
EXPTIME-complete for non-deterministic parity automata.

Reactive syntheses for timed systems

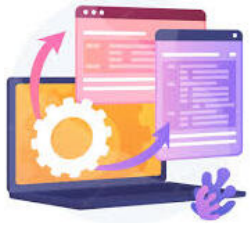
Build a real-time system correct by construction

Interactions $\rightsquigarrow i_0 o_0 t_0 i_1 o_1 t_1 i_2 o_2 t_2 \cdots$

Output: O^ω  Input: I^ω

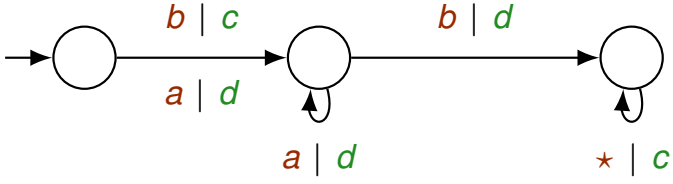


member ?



Reactive synthesis

Church synthesis 



Reactive syntheses problems

EXPTIME-complete for non-deterministic parity automata.

Reactive synthesis seen as a two-player game

How to play in this game?

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

(a_1, t_1)

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

(a_1, t_1)

b_1

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

(a_1, t_1)

(a_2, t_2)

b_1

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

(a_1, t_1)

(a_2, t_2)

b_1

b_2

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

(a_1, t_1)

b_1

(a_2, t_2)

b_2

$\vdots$

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

(a_1, t_1)

(a_2, t_2)

$\vdots$

b_1

b_2

$\vdots$

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

(a_1, t_1)

(a_2, t_2)

$\vdots$

b_1

b_2

$\vdots$

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

(a_1, t_1)

(a_2, t_2)

$\vdots$

b_1

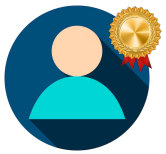
b_2

$\vdots$

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

Winning condition: $\mathcal{A}$

a non-deterministic timed automata over $A \times B$



wins $\Leftrightarrow w \in \mathcal{L}(\mathcal{A})$

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

(a_1, t_1)

(a_2, t_2)

$\vdots$

b_1

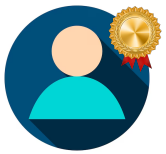
b_2

$\vdots$

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

Winning condition: $\mathcal{A}$

a non-deterministic timed automata over $A \times B$



wins $\Leftrightarrow w \in \mathcal{L}(\mathcal{A})$

Deterministic TA defined a timed game on an arena

Reactive synthesis seen as a two-player game

How to play in this game?

$A \times \mathbb{Q}_{\geqslant 0}$

B

(a_1, t_1)

(a_2, t_2)

$\vdots$

b_1

b_2

$\vdots$

Produce $w \in (A \times B \times \mathbb{Q}_{\geqslant 0})^\omega$

Winning condition: $\mathcal{A}$

a non-deterministic timed automata over $A \times B$

wins $\Leftrightarrow w \in \mathcal{L}(\mathcal{A})$

Deterministic TA defined a timed game on an arena

Reactive syntheses on DTA

Problems are decidable over deterministic timed automata.

The element of surprise in timed games, L. de Alfaro, M. Faella, T. Henzinger, R. Majumdar, and M. Stoelinga, CONCUR 2003

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Church synthesis

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

Church synthesis

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

Church synthesis

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

- NTA are not complementable

Church synthesis

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

- NTA are not complementable

$\neq$

Church synthesis

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

- NTA are not complementable
- Timed language are not Borel (a priori)

$\neq$

Church synthesis

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

- NTA are not complementable
- Timed language are not Borel (a priori)

$\neq$

Church synthesis

Output : if the player wins the game

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

- NTA are not complementable
- Timed language are not Borel (a priori)

$\neq$

Church synthesis

Output : if the player wins the game with a controller

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

- NTA are not complementable
- Timed language are not Borel (a priori)

$\neq$

Church synthesis

Output : if the player wins the game with a controller

What is a controller ?

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

- NTA are not complementable
- Timed language are not Borel (a priori)

$\neq$

Church synthesis

Output : if the player wins the game with a controller

What is a controller ?

Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

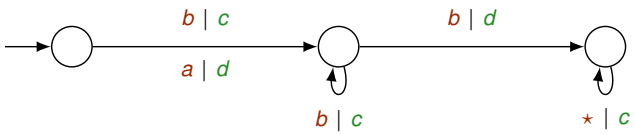
- NTA are not complementable
- Timed language are not Borel (a priori)

$\neq$

Church synthesis

Output : if the player wins the game with a controller

What is a controller ?



Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

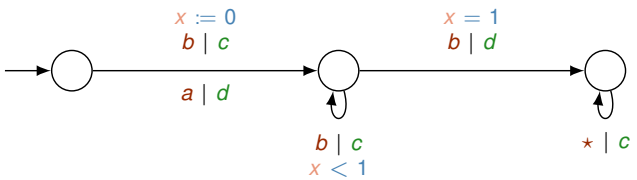
- NTA are not complementable
- Timed language are not Borel (a priori)

$\neq$

Church synthesis

Output : if the player wins the game with a controller

What is a controller ?



Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

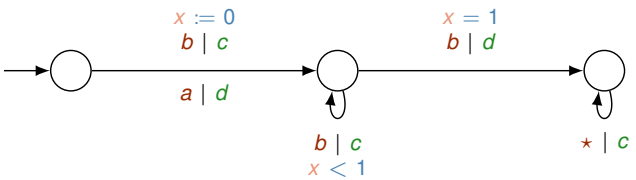
- NTA are not complementable
- Timed language are not Borel (a priori)

$\neq$

Church synthesis

Output : if the player wins the game with a controller

What is a controller ?



Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

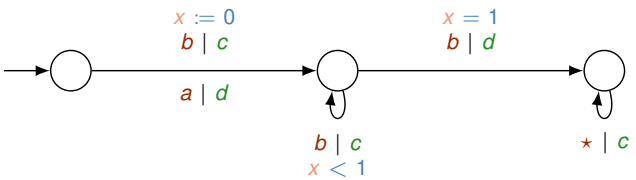
- NTA are not complementable
- Timed language are not Borel (a priori)

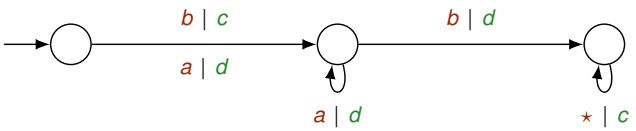
$\neq$

Church synthesis

Output : if the player wins the game with a controller

What is a controller ?





Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

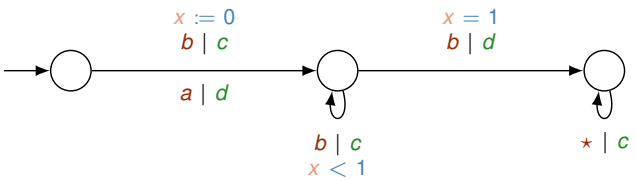
- NTA are not complementable
- Timed language are not Borel (a priori)

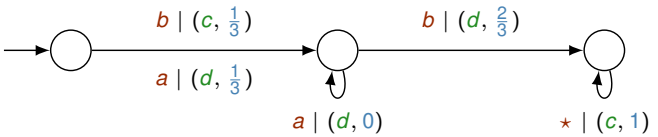
$\neq$

Church synthesis

Output : if the player wins the game with a controller

What is a controller ?





Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

Output : if the player wins the game

4 distincts problems

- NTA are not complementable
- Timed language are not Borel (a priori)

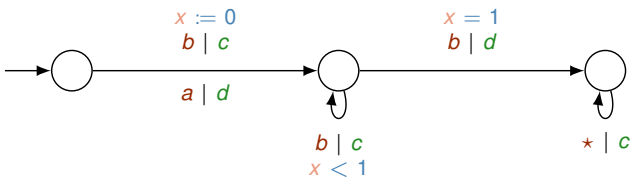
$\neq$

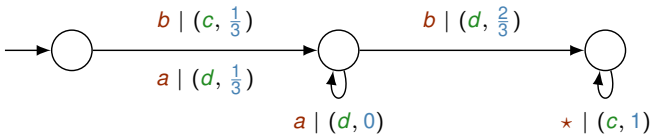
Church synthesis

Output : if the player wins the game with a controller

$\neq$

What is a controller ?





Two distinct classes of reactive syntheses

Input : a NTA, the owner of the winning condition, and one player

Reactive synthesis

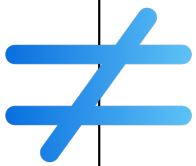
Output : if the player wins the game

Church synthesis

Output : if the player wins the game with a controller

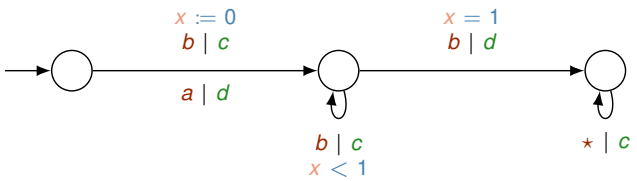
4 distincts problems

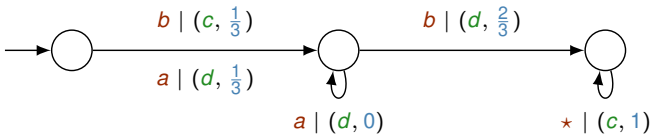
- NTA are not complementable
- Timed language are not Borel (a priori)



What is a controller ?

$\neq$





Study the decidability of these 8 problems.

Undecidability of syntheses problems for Büchi NTA_1

			
Reactive synthesis			
Church synthesis			
			

Undecidability of syntheses problems for Büchi NTA_1

			
Reactive synthesis		Universality	
Church synthesis		Universality sampled langages	
		Universality	

Undecidability of syntheses problems for Büchi NTA_1

			
Reactive synthesis		Universality	
Church synthesis		Universality sampled langages	
		Universality	

For NTA_2
Reduction from finitness in lossy counters machines

Timed Games and Deterministic Separability, L. Clemente, S. Lasota, R. Piórkowski, ICALP, 2020

Undecidability of syntheses problems for Büchi NTA_1

			
Reactive synthesis		Universality	
Church synthesis		Universality sampled langages	
		Universality	

For NTA_2
Reduction from finitness in lossy counters machines

 Can we adapt this reduction for one-clock setting ?

Timed Games and Deterministic Separability, L. Clemente, S. Lasota, R. Piórkowski, ICALP, 2020

Syntheses from lossy counters machines

Finitness in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine

Syntheses from lossy counters machines

Finitness in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Syntheses from lossy counters machines

Finitness in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Lossy counter machines decidability cheat sheet, P. Schnoebelen, RP, 2010.

Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Timed words encode runs of $\mathcal{M}$

Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Timed words encode runs of $\mathcal{M}$

$$\text{run: } (s_0,\nu_0) \xrightarrow{\mathsf{I}(c_1)} (s_1,\nu_1) \xrightarrow{\mathsf{I}(c_2)} (s_2,\nu_2) \xrightarrow{\mathsf{I}(c_1)} (s_3,\nu_3) \xrightarrow{\mathsf{I}(c_1)} (s_4,\nu_4) - \cdots$$

Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Timed words encode runs of $\mathcal{M}$

run: $(s_0, \nu_0) \xrightarrow{\mathsf{I}(c_1)} (s_1, \nu_1) \xrightarrow{\mathsf{I}(c_2)} (s_2, \nu_2) \xrightarrow{\mathsf{I}(c_1)} (s_3, \nu_3) \xrightarrow{\mathsf{I}(c_1)} (s_4, \nu_4) - \cdots$

encoding:

Syntheses from lossy counters machines

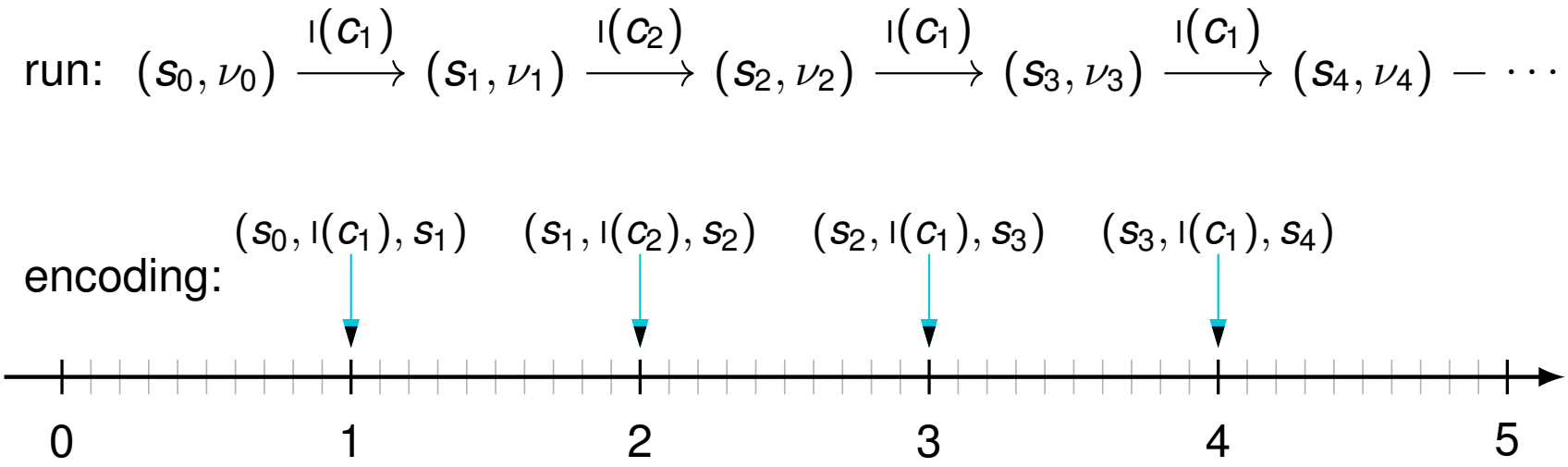
Finitness in lossy counter machine

Undecidable

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Timed words encode runs of $\mathcal{M}$



Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Timed words encode runs of $\mathcal{M}$

run: $(s_0,\nu_0) \xrightarrow{\mathrm{I}(c_1)} (s_1,\nu_1) \xrightarrow{\mathrm{I}(c_2)} (s_2,\nu_2) \xrightarrow{\mathrm{I}(c_1)} (s_3,\nu_3) \xrightarrow{\mathrm{I}(c_1)} (s_4,\nu_4) - \cdots$

encoding: $(s_0,\mathrm{I}(c_1),s_1)$ $(s_1,\mathrm{I}(c_2),s_2)$ $(s_2,\mathrm{I}(c_1),s_3)$ $(s_3,\mathrm{I}(c_1),s_4)$

ν_i: $\nu_0=(0,0,0,0)$ $\nu_1=(1,0,0,0)$ $\nu_2=(1,1,0,0)$ $\nu_3=(2,1,0,0)$ $\nu_4=(0,1,0,0)$

Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

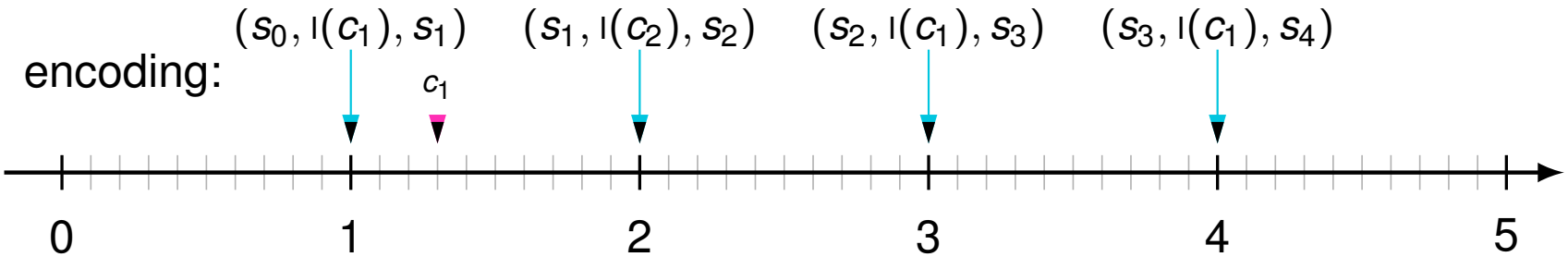
Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Timed words encode runs of $\mathcal{M}$

run: $(s_0,\nu_0) \xrightarrow{\mathrm{I}(c_1)} (s_1,\nu_1) \xrightarrow{\mathrm{I}(c_2)} (s_2,\nu_2) \xrightarrow{\mathrm{I}(c_1)} (s_3,\nu_3) \xrightarrow{\mathrm{I}(c_1)} (s_4,\nu_4) - \cdots$

encoding:



ν_i: $\nu_0 = (0,0,0,0)$ $\nu_1 = (1,0,0,0)$ $\nu_2 = (1,1,0,0)$ $\nu_3 = (2,1,0,0)$ $\nu_4 = (0,1,0,0)$

Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

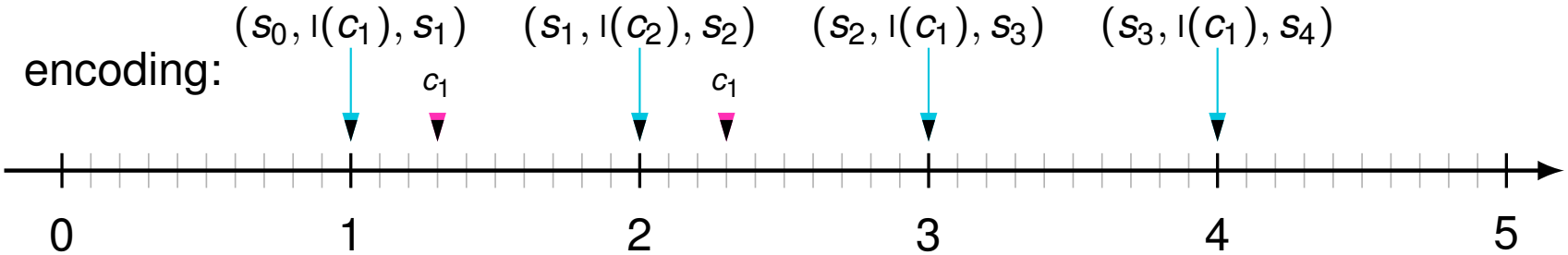
Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Timed words encode runs of $\mathcal{M}$

run: $(s_0,\nu_0) \xrightarrow{\mathrm{I}(c_1)} (s_1,\nu_1) \xrightarrow{\mathrm{I}(c_2)} (s_2,\nu_2) \xrightarrow{\mathrm{I}(c_1)} (s_3,\nu_3) \xrightarrow{\mathrm{I}(c_1)} (s_4,\nu_4) - \cdots$

encoding:



ν_i: $\nu_0 = (0,0,0,0)$ $\nu_1 = (1,0,0,0)$ $\nu_2 = (1,1,0,0)$ $\nu_3 = (2,1,0,0)$ $\nu_4 = (0,1,0,0)$

Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

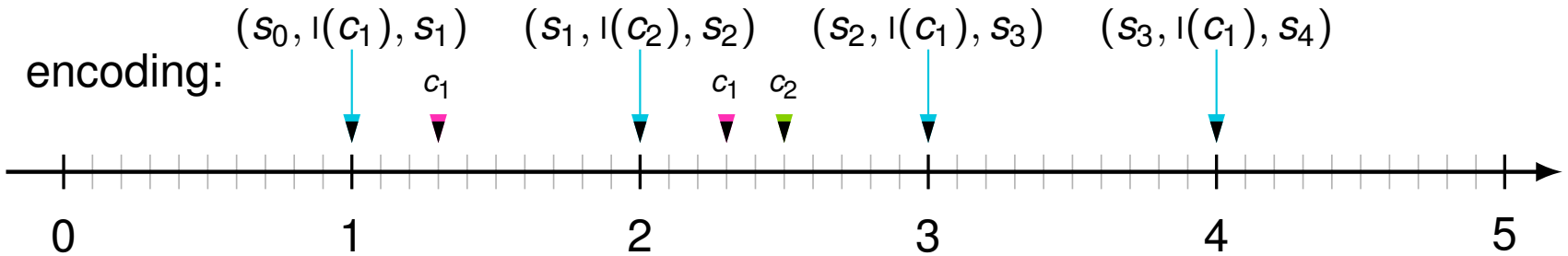
Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Timed words encode runs of $\mathcal{M}$

run: $(s_0,\nu_0) \xrightarrow{\mathrm{I}(c_1)} (s_1,\nu_1) \xrightarrow{\mathrm{I}(c_2)} (s_2,\nu_2) \xrightarrow{\mathrm{I}(c_1)} (s_3,\nu_3) \xrightarrow{\mathrm{I}(c_1)} (s_4,\nu_4) - \cdots$

encoding:



ν_i: $\nu_0 = (0,0,0,0)$ $\nu_1 = (1,0,0,0)$ $\nu_2 = (1,1,0,0)$ $\nu_3 = (2,1,0,0)$ $\nu_4 = (0,1,0,0)$

Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

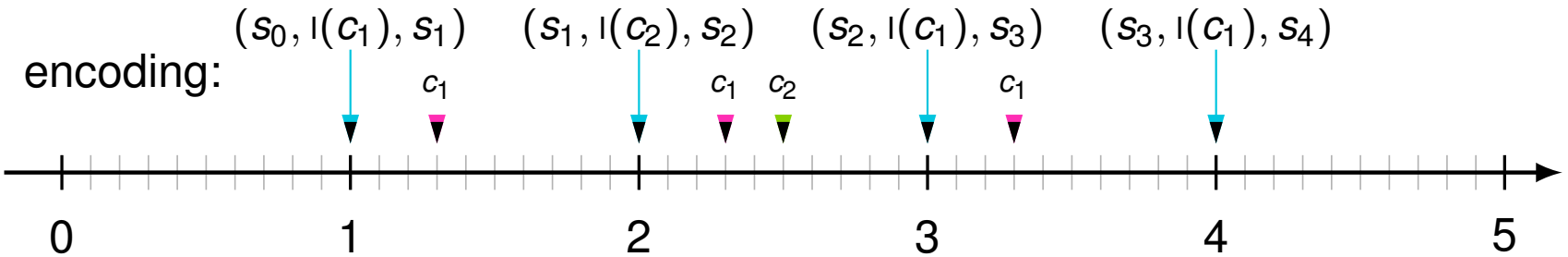
Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Timed words encode runs of $\mathcal{M}$

run: $(s_0,\nu_0) \xrightarrow{\mathrm{I}(c_1)} (s_1,\nu_1) \xrightarrow{\mathrm{I}(c_2)} (s_2,\nu_2) \xrightarrow{\mathrm{I}(c_1)} (s_3,\nu_3) \xrightarrow{\mathrm{I}(c_1)} (s_4,\nu_4) - \cdots$

encoding:



ν_i: $\nu_0 = (0,0,0,0)$ $\nu_1 = (1,0,0,0)$ $\nu_2 = (1,1,0,0)$ $\nu_3 = (2,1,0,0)$ $\nu_4 = (0,1,0,0)$

Syntheses from lossy counters machines

Finitness in lossy counter machine

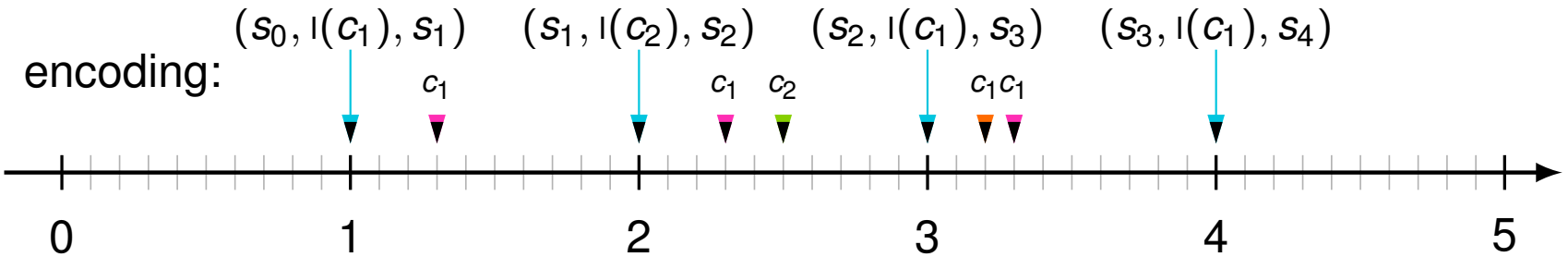
Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Undecidable

Timed words encode runs of $\mathcal{M}$

run: $(s_0,\nu_0) \xrightarrow{\mathrm{I}(c_1)} (s_1,\nu_1) \xrightarrow{\mathrm{I}(c_2)} (s_2,\nu_2) \xrightarrow{\mathrm{I}(c_1)} (s_3,\nu_3) \xrightarrow{\mathrm{I}(c_1)} (s_4,\nu_4) - \cdots$

encoding:



ν_i: $\nu_0 = (0,0,0,0)$ $\nu_1 = (1,0,0,0)$ $\nu_2 = (1,1,0,0)$ $\nu_3 = (2,1,0,0)$ $\nu_4 = (0,1,0,0)$

Syntheses from lossy counters machines

Finitness in lossy counter machine

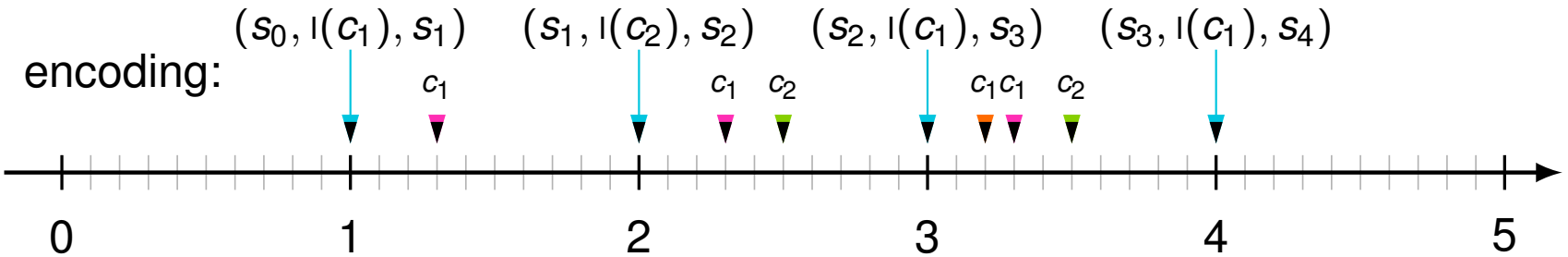
Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Undecidable

Timed words encode runs of $\mathcal{M}$

run: $(s_0,\nu_0) \xrightarrow{\mathrm{I}(c_1)} (s_1,\nu_1) \xrightarrow{\mathrm{I}(c_2)} (s_2,\nu_2) \xrightarrow{\mathrm{I}(c_1)} (s_3,\nu_3) \xrightarrow{\mathrm{I}(c_1)} (s_4,\nu_4) - \cdots$

encoding:



ν_i: $\nu_0 = (0,0,0,0)$ $\nu_1 = (1,0,0,0)$ $\nu_2 = (1,1,0,0)$ $\nu_3 = (2,1,0,0)$ $\nu_4 = (0,1,0,0)$

Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

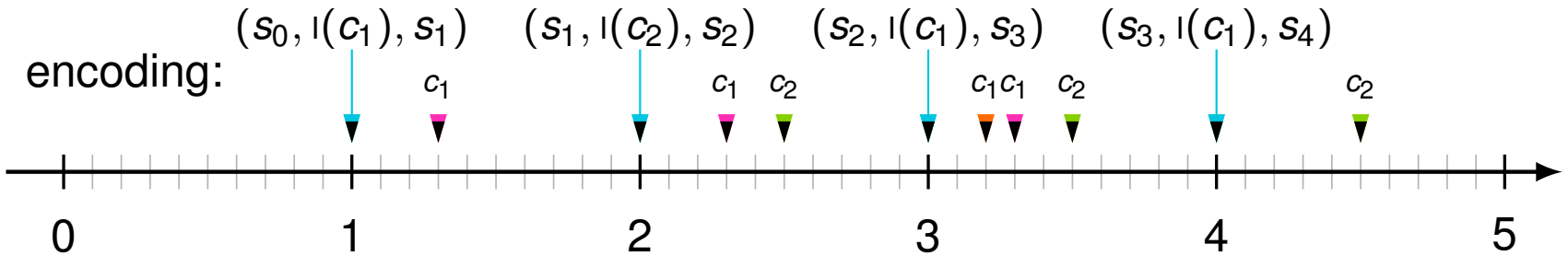
Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Timed words encode runs of $\mathcal{M}$

run: $(s_0,\nu_0) \xrightarrow{\mathrm{I}(c_1)} (s_1,\nu_1) \xrightarrow{\mathrm{I}(c_2)} (s_2,\nu_2) \xrightarrow{\mathrm{I}(c_1)} (s_3,\nu_3) \xrightarrow{\mathrm{I}(c_1)} (s_4,\nu_4) - \cdots$

encoding:



ν_i: $\nu_0 = (0,0,0,0)$ $\nu_1 = (1,0,0,0)$ $\nu_2 = (1,1,0,0)$ $\nu_3 = (2,1,0,0)$ $\nu_4 = (0,1,0,0)$

Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)

Output : if the set of reachable configurations is finite.

Timed words encode
runs of $\mathcal{M}$

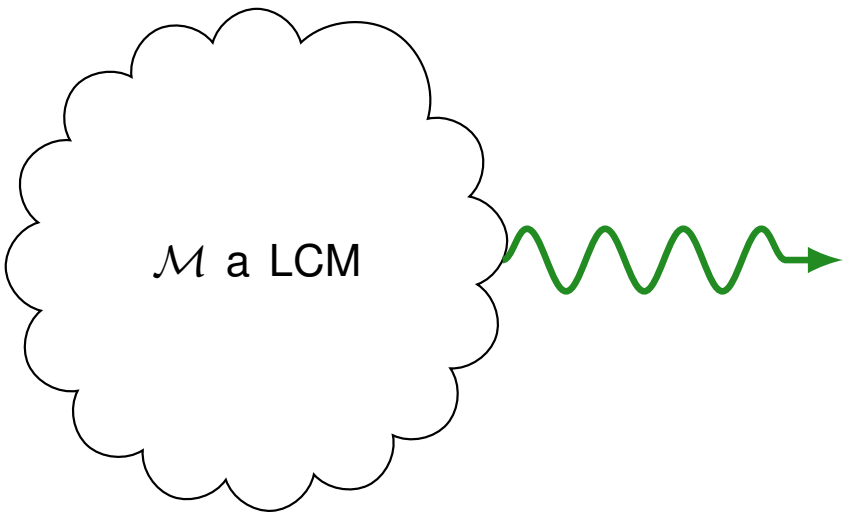
Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Timed words encode
runs of $\mathcal{M}$



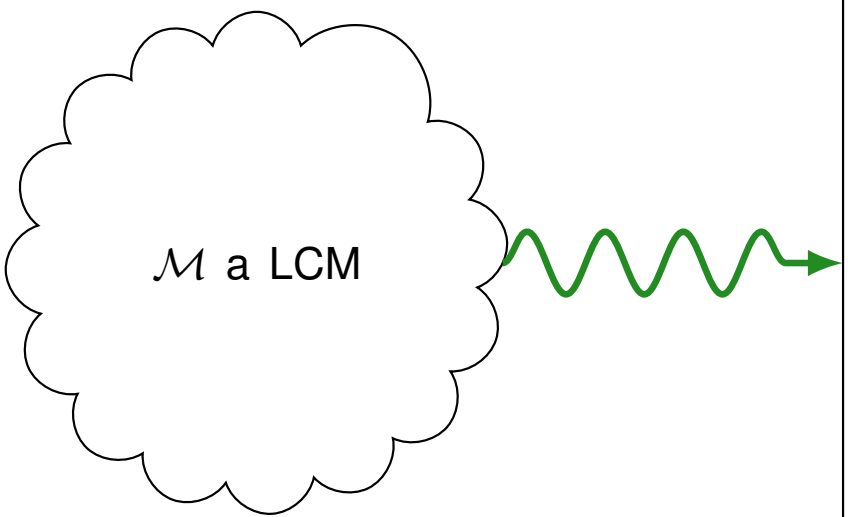
Syntheses from lossy counters machines

Finitness in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Undecidable

Timed words encode
runs of $\mathcal{M}$



Reactive synthesis game $\mathcal{G}$

actions

encoding a run or
almost a run

Syntheses from lossy counters machines

Finitness in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Undecidable

Timed words encode
runs of $\mathcal{M}$

$\mathcal{M}$ a LCM

Reactive synthesis game $\mathcal{G}$

actions

encoding a run or
almost a run

monitor the run to
detect errors

Syntheses from lossy counters machines

Finitness in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Undecidable

Timed words encode
runs of $\mathcal{M}$

$\mathcal{M}$ a LCM

Reactive synthesis game $\mathcal{G}$

actions

encoding a run or
almost a run

monitor the run to
detect errors

winning condition : NTA_2
if  fails in monitoring

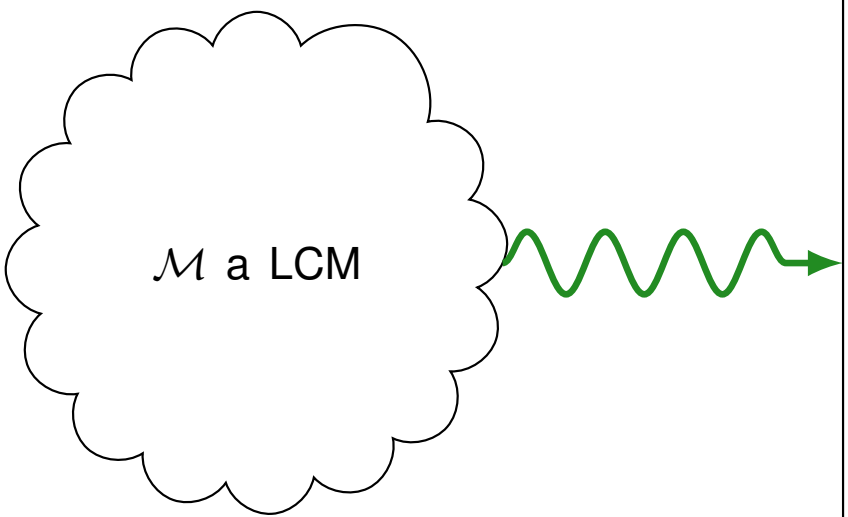
Syntheses from lossy counters machines

Finitness in lossy counter machine

Undecidable

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Timed words encode
runs of $\mathcal{M}$



Reactive synthesis game $\mathcal{G}$

actions

encoding a run or
almost a run

monitor the run to
detect errors

winning condition : NTA_2
if  fails in monitoring

Syntheses from lossy counters machines

Finitness in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Undecidable

Timed words encode
runs of $\mathcal{M}$

$\mathcal{M}$ a LCM

Reactive synthesis game $\mathcal{G}$

actions

encoding a run or
almost a run

monitor the run to
detect errors

winning condition : NTA_2
if  fails in monitoring

always wins

Syntheses from lossy counters machines

Finitness in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Undecidable

Timed words encode
runs of $\mathcal{M}$

$\mathcal{M}$ a LCM

Reactive synthesis game $\mathcal{G}$

actions

encoding a run or
almost a run

monitor the run to
detect errors

winning condition : NTA_2
if  fails in monitoring

always wins
wins with a controller $\Leftrightarrow$ $\mathcal{M}$ is finite

Syntheses from lossy counters machines

Finitness in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine
(counters can always decrease by lossing values)
Output : if the set of reachable configurations is finite.

Undecidable

Timed words encode runs of $\mathcal{M}$

$\mathcal{M}$ a LCM

Reactive synthesis game $\mathcal{G}$

actions

encoding a run or almost a run

monitor the run to detect errors

winning condition : NTA_2
if  fails in monitoring

always wins
wins with a controller $\Leftrightarrow$ $\mathcal{M}$ is finite

Why two clocks are needed ?

Removing one clock in the winning condition

  can win by encoding a good run of $\mathcal{M}$

Removing one clock in the winning condition

  can win by encoding a good run of $\mathcal{M}$

Büchi in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine and q_f
Output : if q_f is repeatly reachable.

Lossy counter machines decidability cheat sheet, P. Schnoebelen, RP, 2010.

Removing one clock in the winning condition

  can win by encoding a good run of $\mathcal{M}$

Büchi in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine and q_f
Output : if q_f is repeatly reachable.

 wins with a controller

 needs infinite memory to win

Removing one clock in the winning condition

  can win by encoding a good run of $\mathcal{M}$

Büchi in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine and q_f
Output : if q_f is repeatly reachable.

 wins with a controller

 needs infinite memory to win

 Use a new encoding of runs into timed words

Removing one clock in the winning condition


can win by encoding a good run of $\mathcal{M}$

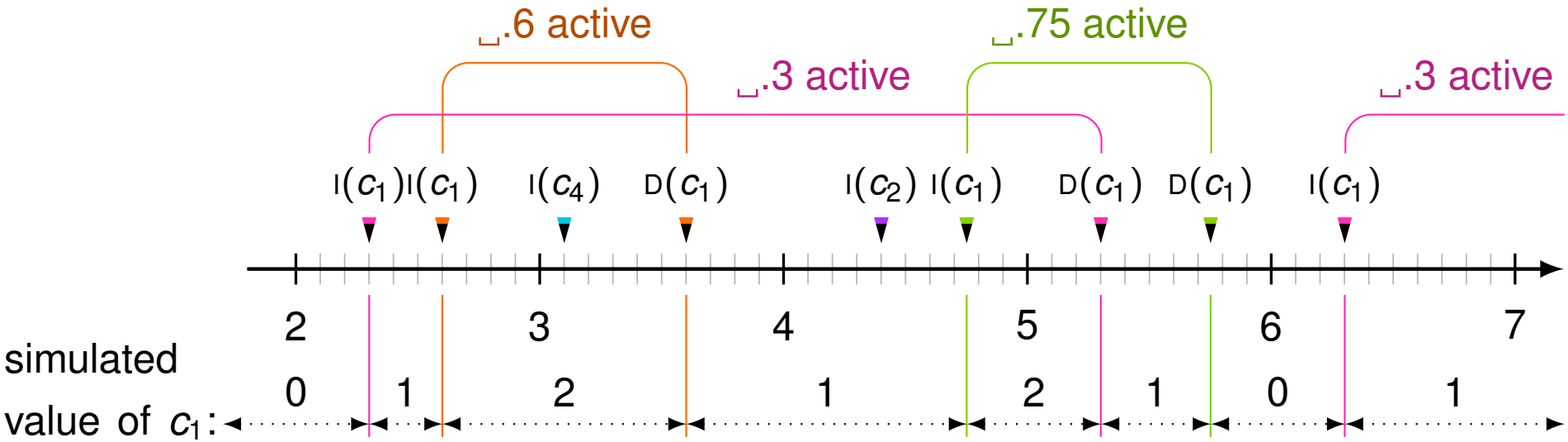
Büchi in lossy counter machine

Input : $\mathcal{M}$ a lossy counter machine and q_f
Output : if q_f is repeatly reachable.

wins with a controller

needs infinite memory to win

Use a new encoding of runs into timed words



Undecidability of syntheses problems for Büchi NTA_1

			
Reactive synthesis		Universality	
Church synthesis		Universality sampled langages	
		Universality	

For NTA_2
Reduction from finitness in lossy counters machines

Undecidability of syntheses problems for Büchi NTA_1

			
Reactive synthesis		Universality	Büchi in lossy counters machines
Church synthesis		Universality sampled langages	Büchi in lossy counters machines
		Universality	Finitness in lossy counters machines

For NTA_2
Reduction from finitness in lossy counters machines

Undecidability of syntheses problems for Büchi NTA_1

			
Reactive synthesis		Universality	Büchi in lossy counters machines
Church synthesis		Universality sampled langages	Büchi in lossy counters machines
		Universality	Finitness in lossy counters machines

For NTA_2
Reduction from finitness in lossy counters machines

Synthese of register automata

Challenging questions

Existence of decidable subclass of game ?

Challenging questions

 Existence of decidable subclass of game ?

 restriction of the winning condition

Challenging questions

 Existence of decidable subclass of game ?

 restriction of the winning condition

- 1-clock reachability NTA

Challenging questions

 Existence of decidable subclass of game ?

 restriction of the winning condition

- 1-clock reachability NTA
- history-deterministic Büchi TA

Challenging questions

 Existence of decidable subclass of game ?

- restriction of the winning condition
  - 1-clock reachability NTA
  - history-deterministic Büchi TA
- restriction of players' powers

Challenging questions

Existence of decidable subclass of game ?

- restriction of the winning condition
  - 1-clock reachability NTA
  - history-deterministic Büchi TA
- restriction of players' powers
  - (finite) alphabet used

Challenging questions

 Existence of decidable subclass of game ?

- restriction of the winning condition
  - 1-clock reachability NTA
  - history-deterministic Büchi TA
- restriction of players' powers
  - (finite) alphabet used
  - familly of strategies used

Challenging questions

Existence of decidable subclass of game ?

- restriction of the winning condition
  - 1-clock reachability NTA
  - history-deterministic Büchi TA
- restriction of players' powers
  - (finite) alphabet used
  - familly of strategies used

Decidability for resource-bounded synthesis

Timed Games and Deterministic Separability, L. Clemente, S. Lasota, R. Piórkowski, ICALP, 2020

Challenging questions

 Existence of decidable subclass of game ?

- restriction of the winning condition
  - 1-clock reachability NTA
  - history-deterministic Büchi TA
- restriction of players' powers
  - (finite) alphabet used
  - familly of strategies used

 Decidability for resource-bounded synthesis

- change the semantics of timed automata : robustness

Challenging questions

 Existence of decidable subclass of game ?

 restriction of the winning condition

- 1-clock reachability NTA
- history-deterministic Büchi TA

 restriction of players' powers

- (finite) alphabet used
- familly of strategies used

 Decidability for resource-bounded synthesis

 change the semantics of timed automata : robustness

 Are theses games determined ?

Challenging questions

 Existence of decidable subclass of game ?

- restriction of the winning condition
  - 1-clock reachability NTA
  - history-deterministic Büchi TA
- restriction of players' powers
  - (finite) alphabet used
  - familly of strategies used

 Decidability for resource-bounded synthesis

- change the semantics of timed automata : robustness

 Are theses games determined ?

 not Borel

Challenging questions

 Existence of decidable subclass of game ?

- restriction of the winning condition
  - 1-clock reachability NTA
  - history-deterministic Büchi TA
- restriction of players' powers
  - (finite) alphabet used
  - familly of strategies used

 Decidability for resource-bounded synthesis

- change the semantics of timed automata : robustness

 Are theses games determined ?

 not Borel

- study the topology of timed langages

Challenging questions

 Existence of decidable subclass of game ?

 restriction of the winning condition

- 1-clock reachability NTA
- history-deterministic Büchi TA

 restriction of players' powers

- (finite) alphabet used
- familly of strategies used

 Decidability for resource-bounded synthesis

 change the semantics of timed automata : robustness

 Are theses games determined ?

 not Borel

 study the topology of timed langages

Thank you! Questions ?

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable

- reachability NTA_2

A theory of timed automata, R. Alur and D. L. Dill, TCS, 1994

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable

- reachability NTA_2
- Büchi NTA_1

Alternating timed automata, S. Lasota and I. Walukiewicz, ACM Transactions on Computational Logic, 2008

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable
- reachability NTA_2
- Büchi NTA_1

Ackerman-complete.
- reachability NTA_1

Alternating timed automata, S. Lasota and I. Walukiewicz, ACM Transactions on Computational Logic, 2008

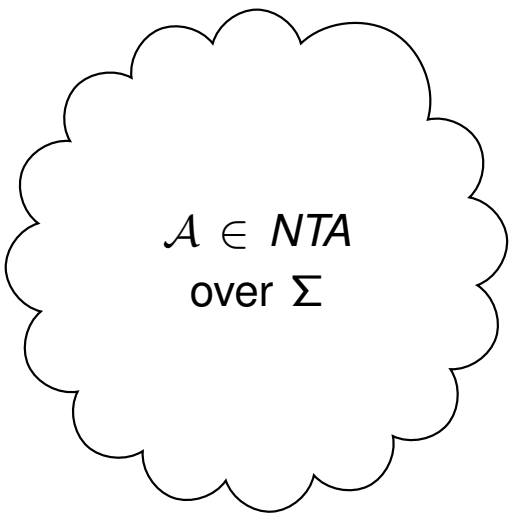
Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable
- reachability NTA_2
- Büchi NTA_1

Ackerman-complete.
- reachability NTA_1



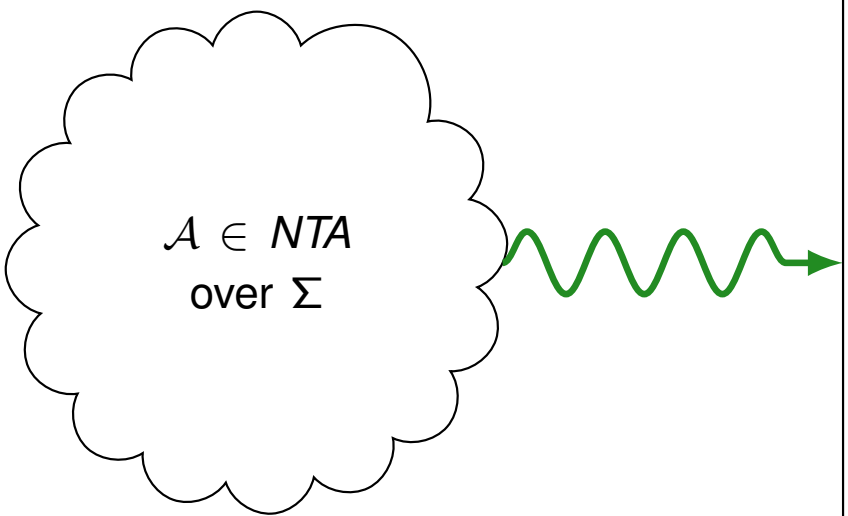
Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable
- reachability NTA_2
- Büchi NTA_1

Ackerman-complete.
- reachability NTA_1



Reactive synthesis game $\mathcal{G}$

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable
- reachability NTA_2
- Büchi NTA_1

Ackerman-complete.
- reachability NTA_1

$\mathcal{A} \in$ *NTA* over Σ

Reactive synthesis game $\mathcal{G}$

actions → Σ

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable
- reachability NTA_2
- Büchi NTA_1

Ackerman-complete.
- reachability NTA_1

$\mathcal{A} \in$ *NTA* over Σ

Reactive synthesis game $\mathcal{G}$

actions

Σ

nothing

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable
- reachability NTA_2
- Büchi NTA_1

Ackerman-complete.
- reachability NTA_1

$\mathcal{A} \in$ *NTA* over Σ

Reactive synthesis game $\mathcal{G}$

actions

Σ

nothing

winning condition : $\mathcal{A}$

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable
- reachability NTA_2
- Büchi NTA_1

Ackerman-complete.
- reachability NTA_1

$\mathcal{A} \in$ *NTA* over Σ

Reactive synthesis game $\mathcal{G}$

actions

Σ

nothing

winning condition : $\mathcal{A}$

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable
- reachability NTA_2
- Büchi NTA_1

Ackerman-complete.
- reachability NTA_1

$\mathcal{A} \in$ *NTA* over Σ

Reactive synthesis game $\mathcal{G}$

actions

Σ

nothing

winning condition : $\mathcal{A}$

wins $\Leftrightarrow \exists w \notin \mathcal{L}(\mathcal{A})$

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable
- reachability NTA_2
- Büchi NTA_1

Ackerman-complete.
- reachability NTA_1

$\mathcal{A} \in$ *NTA* over Σ

Reactive synthesis game $\mathcal{G}$

actions

Σ

nothing

winning condition : $\mathcal{A}$

wins $\Leftrightarrow \exists w \notin \mathcal{L}(\mathcal{A})$

wins with a controller $\Leftrightarrow \forall w \in \mathcal{L}(\mathcal{A})$

Syntheses from universality problem in *NTA*

Universality

Input : $\mathcal{A} \in$ *NTA* over Σ
Output : if $\mathcal{L}(\mathcal{A}) = \mathbb{T}(\Sigma^*)$
(or $\mathbb{T}(\Sigma^\omega)$)

Undecidable
- reachability NTA_2
- Büchi NTA_1

Ackerman-complete.
- reachability NTA_1

$\mathcal{A} \in$ *NTA* over Σ

Reactive synthesis game $\mathcal{G}$

actions

Σ

nothing

winning condition : $\mathcal{A}$

wins $\Leftrightarrow \exists w \notin \mathcal{L}(\mathcal{A})$

wins with a controller $\Leftrightarrow \forall w \in \mathcal{L}(\mathcal{A})$

How to guarantee a winning controller for

?

Controller for  from universality problems

Universal sampled timed language

Input : $\mathcal{A}$ a *NTA*
Output : if for all $\varepsilon > 0$, $\mathcal{L}_\varepsilon(\mathcal{A})$ is universal.

Controller for  from universality problems

Universal sampled timed language

Input : $\mathcal{A}$ a *NTA*
Output : if for all $\varepsilon > 0$, $\mathcal{L}_\varepsilon(\mathcal{A})$ is universal.

Undecidable

- Büchi NTA_1

Sampled universality of timed automata, P.Abdulla, P. Krcal, and W. Yi, FoSSaCS, 2007.

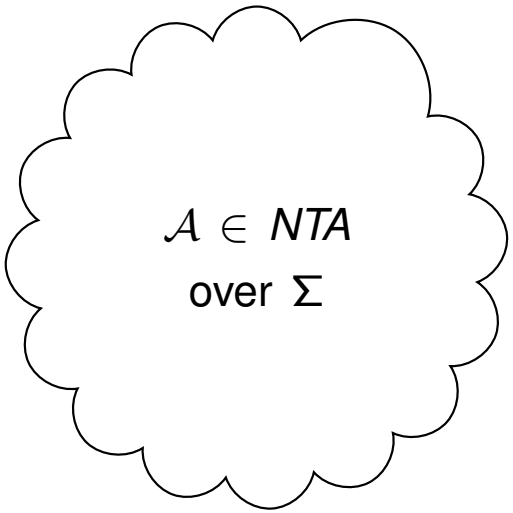
Controller for  from universality problems

Universal sampled timed language

Input : $\mathcal{A}$ a *NTA*
Output : if for all $\varepsilon > 0$, $\mathcal{L}_\varepsilon(\mathcal{A})$ is universal.

Undecidable

- Büchi NTA_1



Controller for from universality problems

Universal sampled timed language

Input : $\mathcal{A}$ a *NTA*
Output : if for all $\varepsilon > 0$, $\mathcal{L}_\varepsilon(\mathcal{A})$ is universal.

Undecidable
- Büchi NTA_1

$\mathcal{A} \in NTA$ over Σ

Reactive synthesis game $\mathcal{G}$

actions → Σ

actions → nothing

winning condition : $\mathcal{A}$

Controller for from universality problems

Universal sampled timed language

Input : $\mathcal{A}$ a *NTA*
Output : if for all $\varepsilon > 0$, $\mathcal{L}_\varepsilon(\mathcal{A})$ is universal.

Undecidable
- Büchi NTA_1

$\mathcal{A} \in NTA$ over Σ

Reactive synthesis game $\mathcal{G}$

actions → Σ

actions → nothing

winning condition : $\mathcal{A}$

wins $\Leftrightarrow \exists w \notin \mathcal{L}(\mathcal{A})$

Controller for from universality problems

Universal sampled timed language

Input : $\mathcal{A}$ a *NTA*
Output : if for all $\varepsilon > 0$, $\mathcal{L}_\varepsilon(\mathcal{A})$ is universal.

Undecidable
- Büchi NTA_1

$\mathcal{A} \in NTA$ over Σ

Reactive synthesis game $\mathcal{G}$

actions → Σ

actions → nothing

winning condition : $\mathcal{A}$

---

wins with a controller $\Leftrightarrow$
$\exists \varepsilon, w \mid w \in \mathcal{L}_\varepsilon(\Sigma^\omega) \setminus \mathcal{L}_\varepsilon(\mathcal{A})$

Controller for from universality problems

Universal sampled timed language

Input : $\mathcal{A}$ a *NTA*
Output : if for all $\varepsilon > 0$, $\mathcal{L}_\varepsilon(\mathcal{A})$ is universal.

Undecidable
- Büchi NTA_1

$\mathcal{A} \in NTA$ over Σ

Reactive synthesis game $\mathcal{G}$

actions

Σ

nothing

winning condition : $\mathcal{A}$

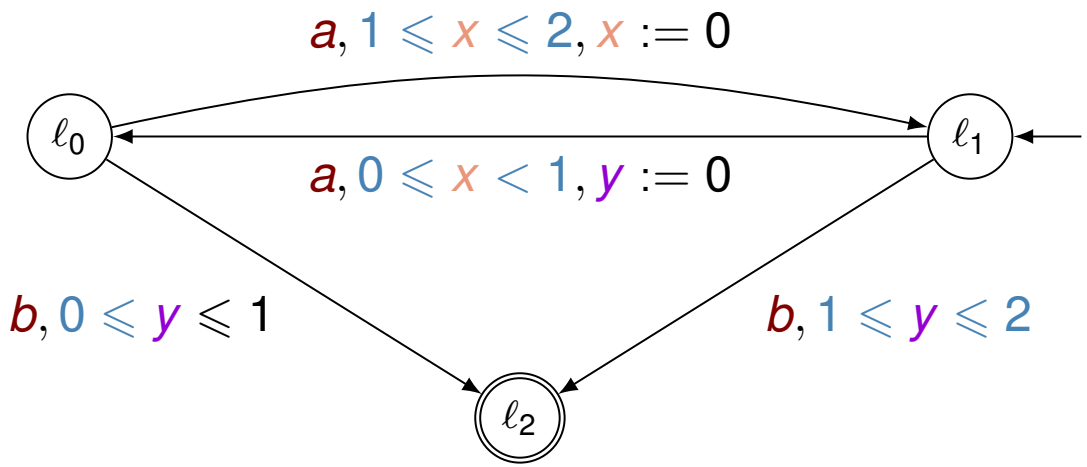
Tick automata

wins with a controller $\Leftrightarrow$
$\exists \varepsilon, w \mid w \in \mathcal{L}_\varepsilon(\Sigma^\omega) \setminus \mathcal{L}_\varepsilon(\mathcal{A})$

On timed automata with discrete time – structural and language theoretical characterization, H. Gruber, M. Holzer, A. Kiehn, and B. König, DLT, 2005.

Interlude : focus on timed automata

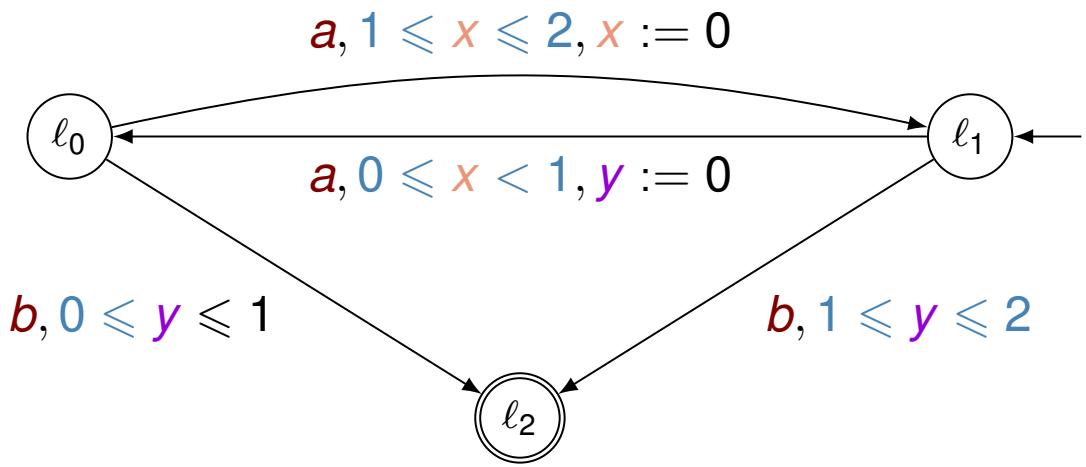
$\mathcal{A}$



A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

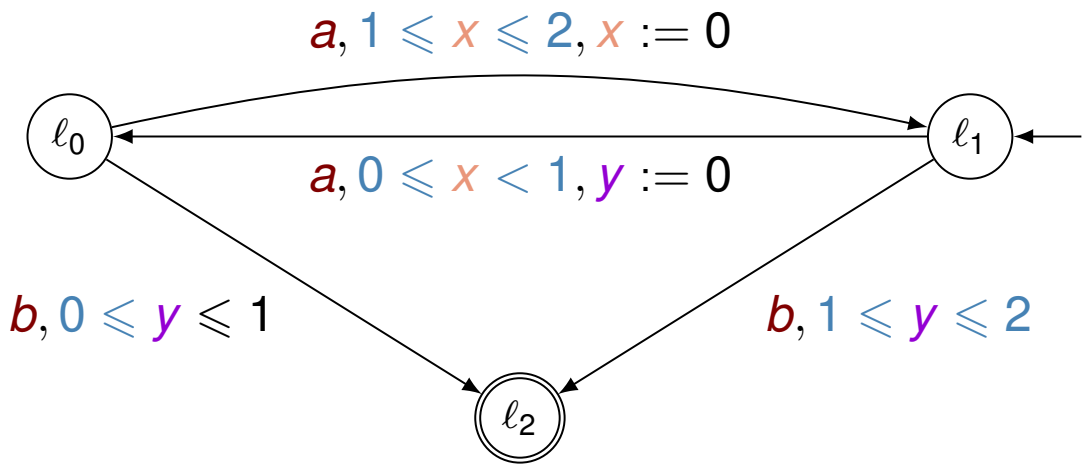
$\mathcal{A}$



A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

$\mathcal{A}$



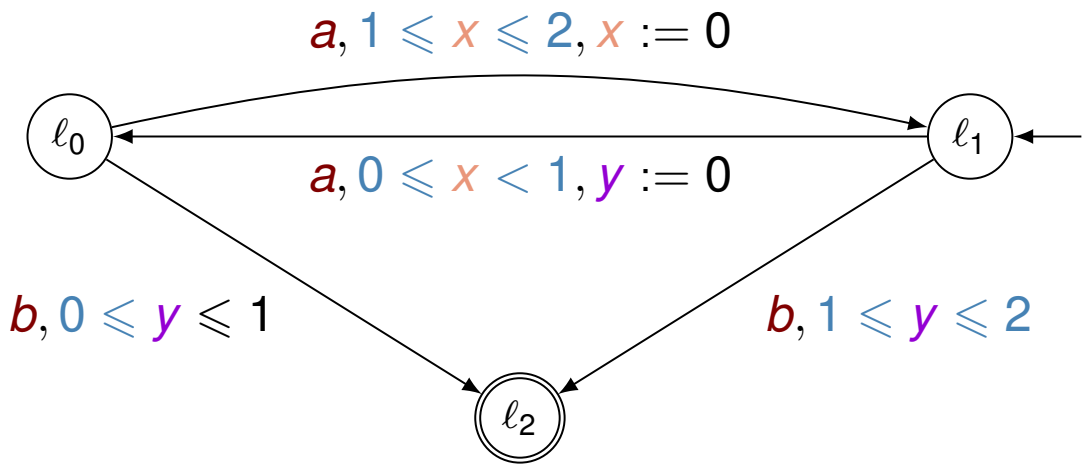
Language of $\mathcal{A}$

$w \in \mathcal{L}(\mathcal{A}) \Leftrightarrow \exists$ an accepting execution on w in $\mathcal{A}$

A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

$\mathcal{A}$



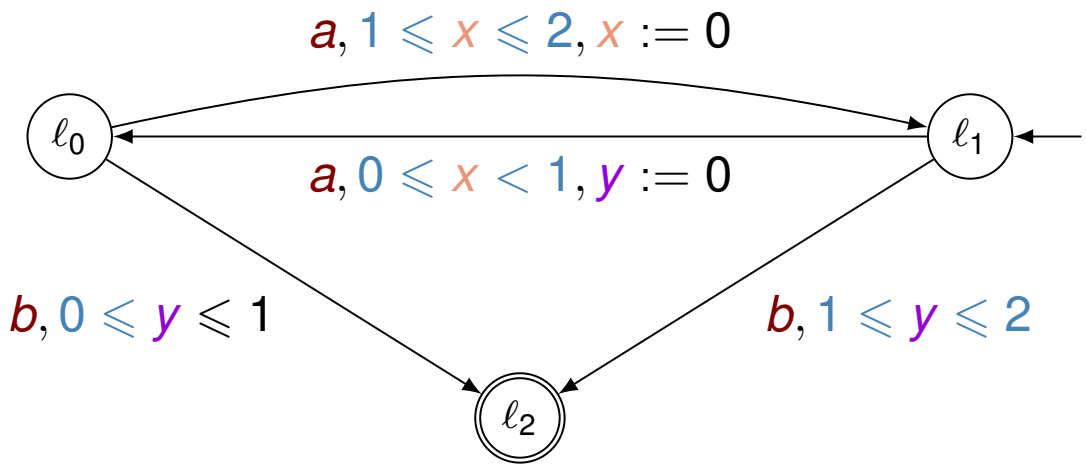
Language of $\mathcal{A}$

$(a, 0.5)(a, 1.25)(b, \frac{1}{3}) = w \in \mathcal{L}(\mathcal{A}) \Leftrightarrow \exists$ an accepting execution on w in $\mathcal{A}$

A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

$\mathcal{A}$



Language of $\mathcal{A}$

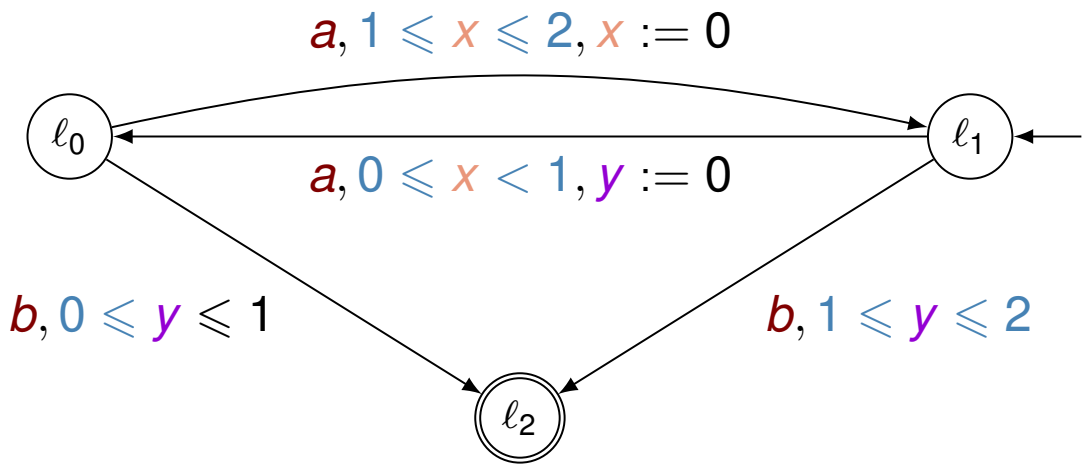
$(a, 0.5)(a, 1.25)(b, \frac{1}{3}) = w \in \mathcal{L}(\mathcal{A}) \Leftrightarrow \exists$ an $\underbrace{\text{accepting execution on } w}$ in $\mathcal{A}$

$$(\ell_1, \begin{bmatrix} x \mapsto 0 \\ y \mapsto 0 \end{bmatrix})$$

A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

$\mathcal{A}$



Language of $\mathcal{A}$

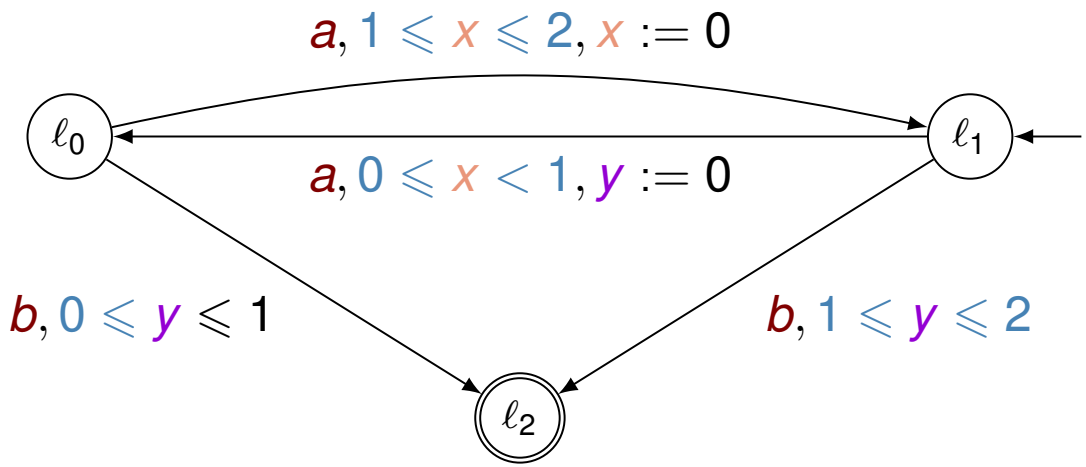
$(a, 0.5)(a, 1.25)(b, \frac{1}{3}) = w \in \mathcal{L}(\mathcal{A}) \Leftrightarrow \exists$ an $\underbrace{\text{accepting execution on } w}$ in $\mathcal{A}$

$$(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix})$$

A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

$\mathcal{A}$



Language of $\mathcal{A}$

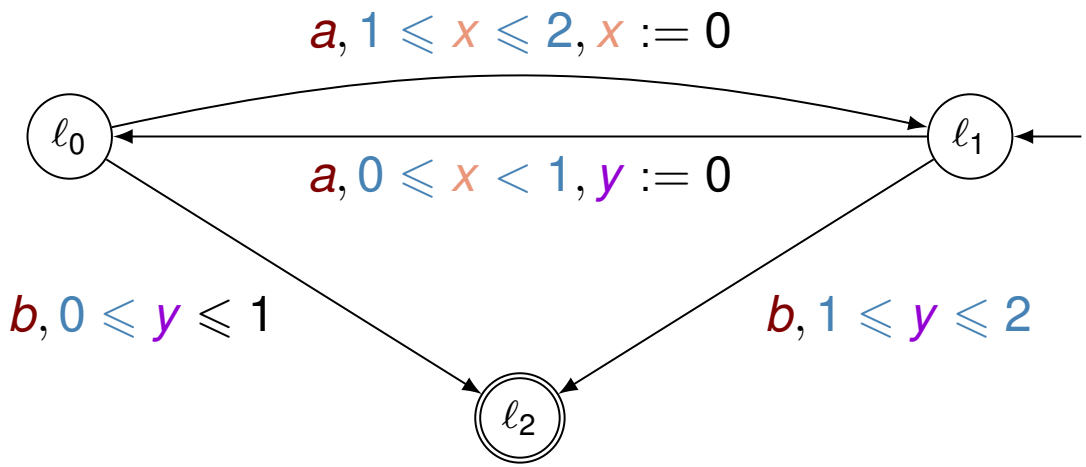
$(a, 0.5)(a, 1.25)(b, \frac{1}{3}) = w \in \mathcal{L}(\mathcal{A}) \Leftrightarrow \exists$ an accepting execution on w in $\mathcal{A}$

$$(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{a,\ 0.5}$$

A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

$\mathcal{A}$



Language of $\mathcal{A}$

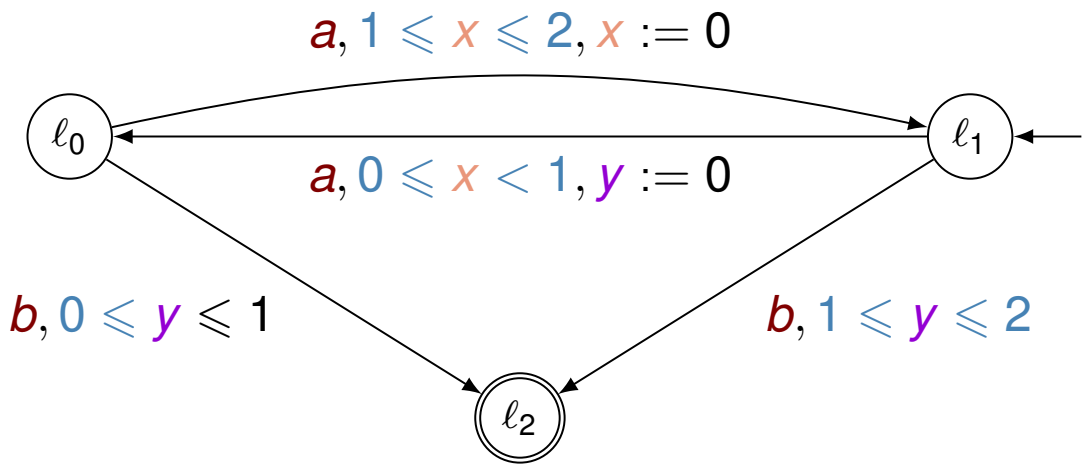
$(a, 0.5)(a, 1.25)(b, \frac{1}{3}) = w \in \mathcal{L}(\mathcal{A}) \Leftrightarrow \exists$ an accepting execution on w in $\mathcal{A}$

$$(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{a,\ 0.5} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix})$$

A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

$\mathcal{A}$



Language of $\mathcal{A}$

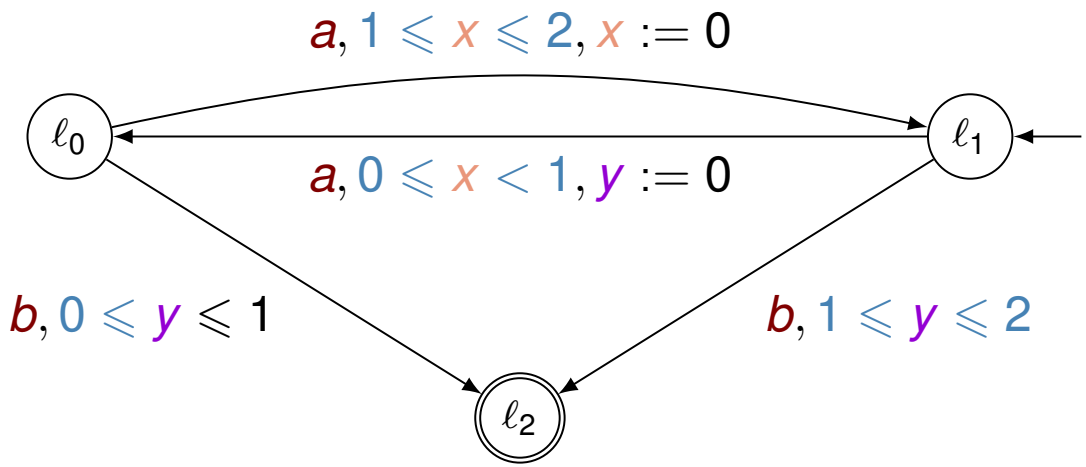
$(a, 0.5)(a, 1.25)(b, \frac{1}{3}) = w \in \mathcal{L}(\mathcal{A}) \Leftrightarrow \exists$ an accepting execution on w in $\mathcal{A}$

$$(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{a,\ 0.5} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{a,\ 1.25} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{b,\ 1/3} (\ell_2, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$$

A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

$\mathcal{A}$



Language of $\mathcal{A}$

$(a, 0.5)(a, 1.25)(b, \frac{1}{3}) = w \in \mathcal{L}(\mathcal{A}) \Leftrightarrow \exists$ an accepting execution on w in $\mathcal{A}$

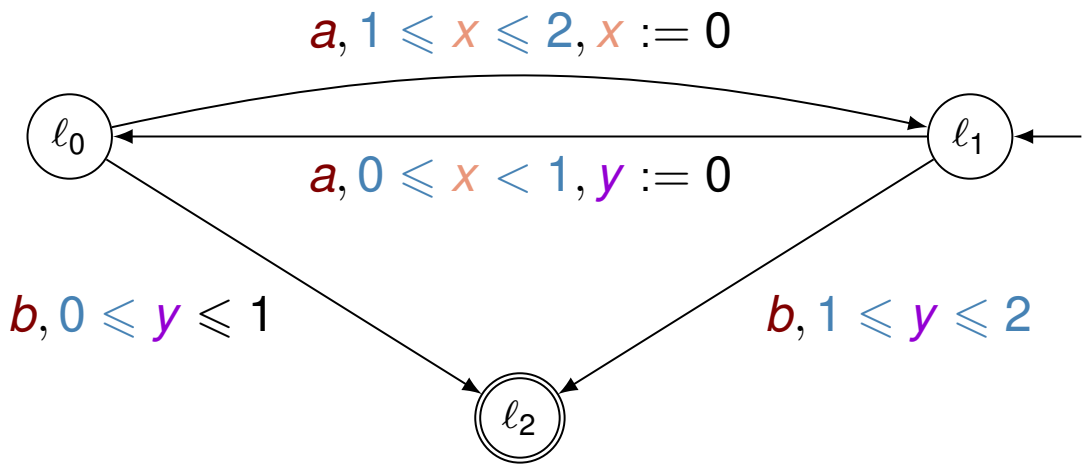
$$(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{a,\ 0.5} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{a,\ 1.25} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{b,\ 1/3} (\ell_2, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$$

Wanted properties for timed automata

A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

$\mathcal{A}$



Language of $\mathcal{A}$

$(a, 0.5)(a, 1.25)(b, \frac{1}{3}) = w \in \mathcal{L}(\mathcal{A}) \Leftrightarrow \exists$ an accepting execution on w in $\mathcal{A}$

$$(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{a,\ 0.5} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{a,\ 1.25} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{b,\ 1/3} (\ell_2, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$$

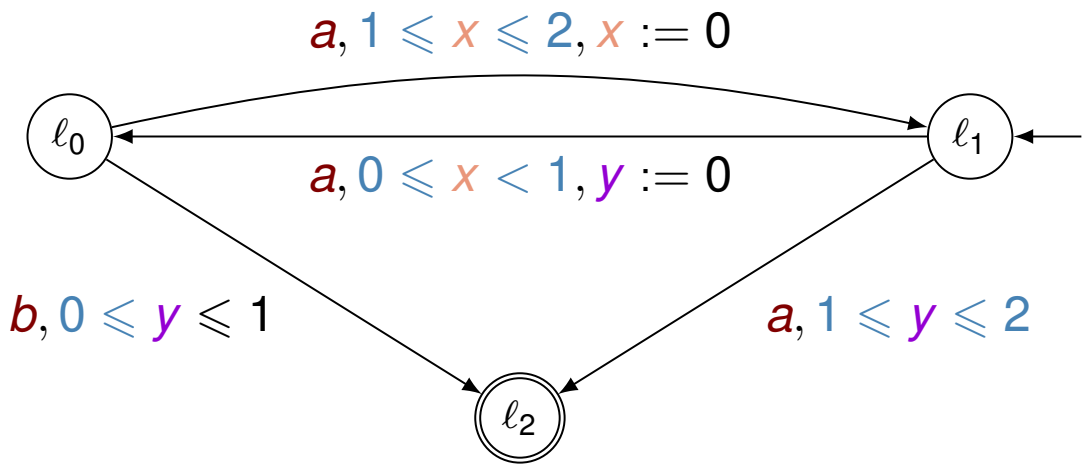
Wanted properties for timed automata

☹ Not determinizable

A theory of timed automata, R. Alur D. Dill, TCS 1994

Interlude : focus on timed automata

$\mathcal{A}$



Language of $\mathcal{A}$

$(a, 0.5)(a, 1.25)(b, \frac{1}{3}) = w \in \mathcal{L}(\mathcal{A}) \Leftrightarrow \exists$ an accepting execution on w in $\mathcal{A}$

$$(\ell_1, \begin{bmatrix} 0 \\ 0 \end{bmatrix}) \xrightarrow{a,\ 0.5} (\ell_0, \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}) \xrightarrow{a,\ 1.25} (\ell_1, \begin{bmatrix} 0 \\ 1.25 \end{bmatrix}) \xrightarrow{b,\ 1/3} (\ell_2, \begin{bmatrix} 1/3 \\ 19/12 \end{bmatrix})$$

Wanted properties for timed automata

☹ Not determinizable ☹ Not stable by complement

A theory of timed automata, R. Alur D. Dill, TCS 1994