Weighted Timed Automata: Optimization Problems

Patricia Bouyer

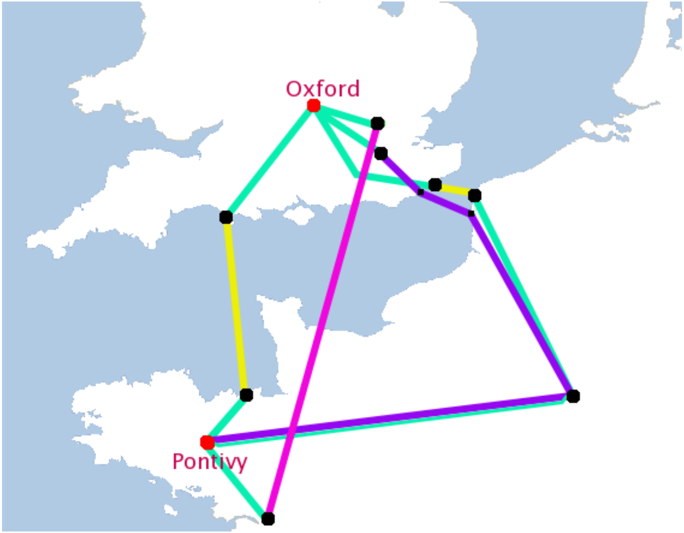
LSV – ENS Cachan & CNRS – France

Based on joint works with Thomas Brihaye (UMH, Belgium), Ed Brinksma (Twente University, The Netherlands), Véronique Bruyère (UMH, Belgium), Franck Cassez (IRCCyN, France), Emmanuel Fleury (LaBRI, France), Kim G. Larsen (Aalborg University, Denmark), Nicolas Markey (LSV, France), Jean-François Raskin (ULB, Belgium), and Jacob Illum Rasmussen (Aalborg University, Denmark)

Outline

1. Introduction
2. Timed automata with costs
3. Optimal timed games
4. Conclusion

A starting example



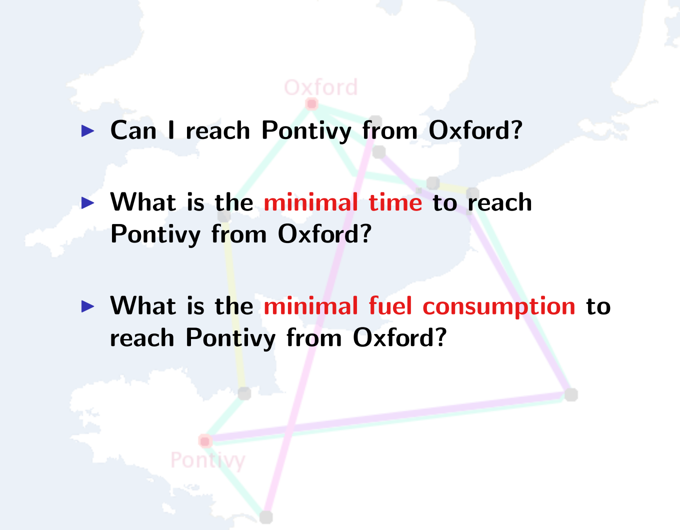
Natural questions

- **Can I reach Pontivy from Oxford?**

Natural questions

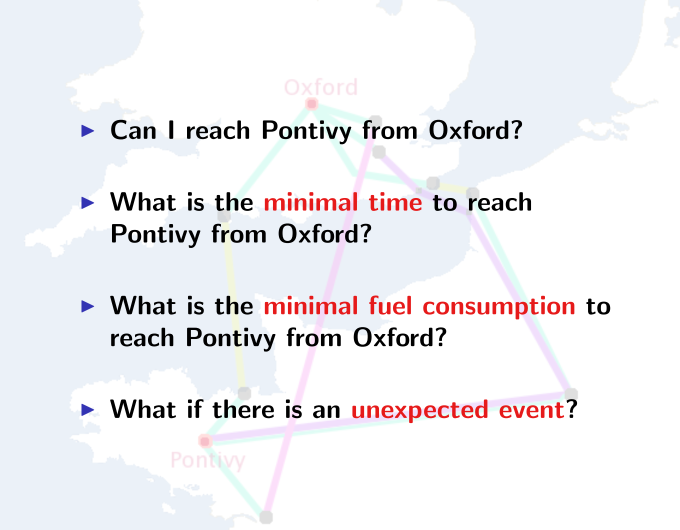
- **Can I reach Pontivy from Oxford?**
- **What is the minimal time to reach Pontivy from Oxford?**

Natural questions



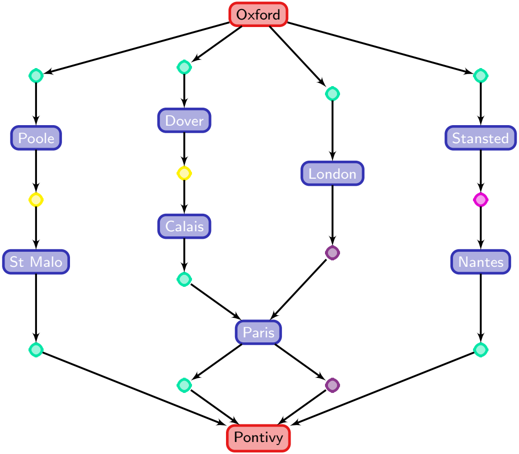
- **Can I reach Pontivy from Oxford?**
- **What is the minimal time to reach Pontivy from Oxford?**
- **What is the minimal fuel consumption to reach Pontivy from Oxford?**

Natural questions

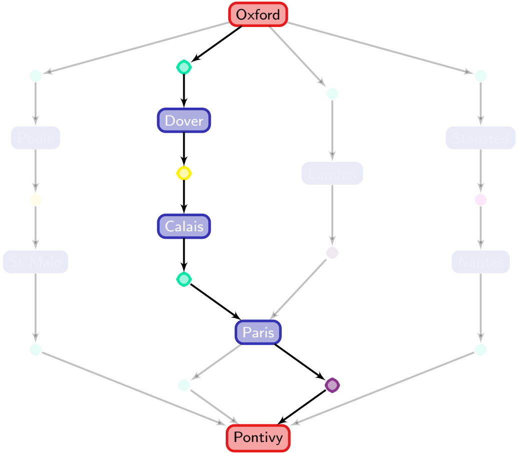


- **Can I reach Pontivy from Oxford?**
- **What is the minimal time to reach Pontivy from Oxford?**
- **What is the minimal fuel consumption to reach Pontivy from Oxford?**
- **What if there is an unexpected event?**

A first model of the system

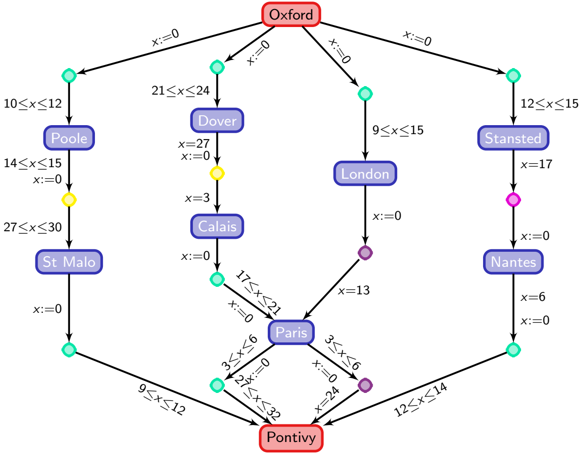


Can I reach Pontivy from Oxford?

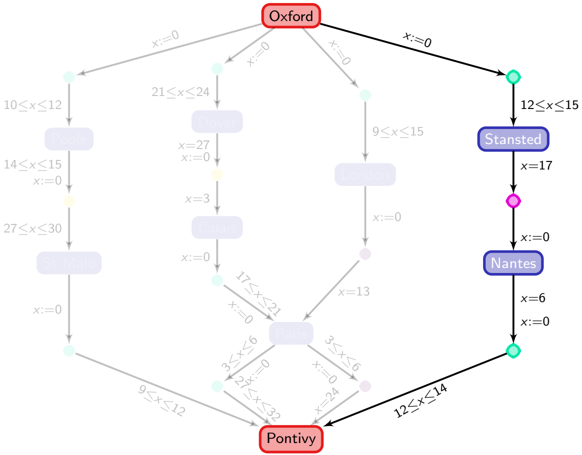


This is a reachability question in a finite graph: Yes, I can!

A second model of the system

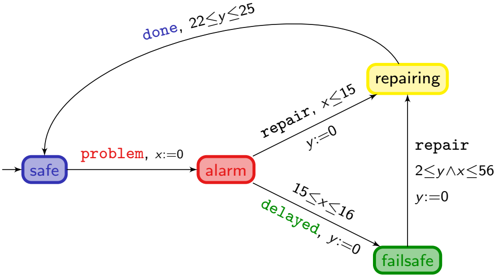


How long will that take?

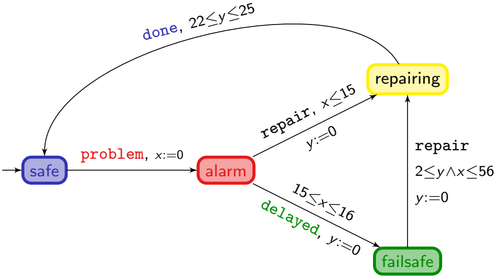


It is a reachability (and optimization) question in a timed automaton: at least 350mn = 5h50mn!

An example of a timed automaton

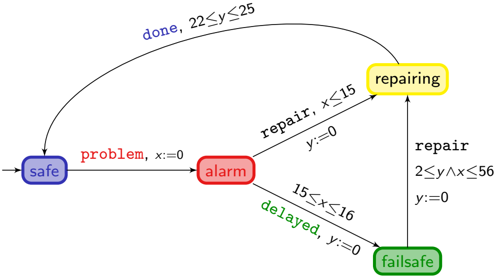


An example of a timed automaton



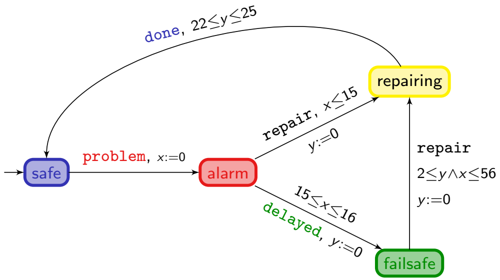
	safe
x	0
y	0

An example of a timed automaton



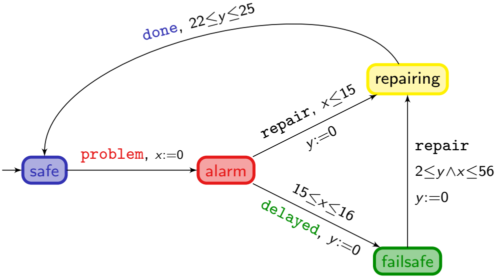
	safe	$\xrightarrow{23}$	safe
x	0		23
y	0		23

An example of a timed automaton



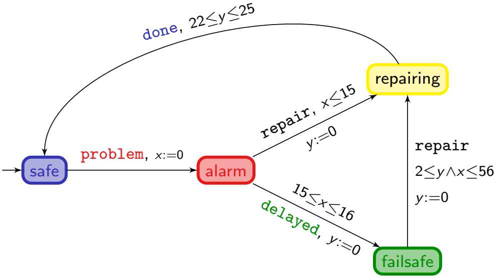
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm
x	0		23		0
y	0		23		23

An example of a timed automaton



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm
x	0		23		0		15.6
y	0		23		23		38.6

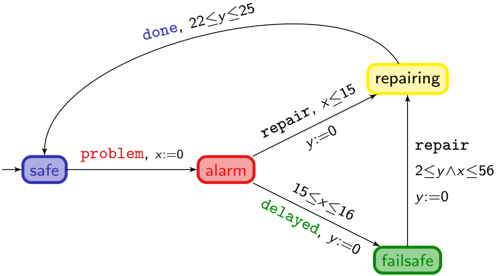
An example of a timed automaton



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe	
x	0		23		0		15.6		15.6	···
y	0		23		23		38.6		0	

	failsafe
···	15.6
	0

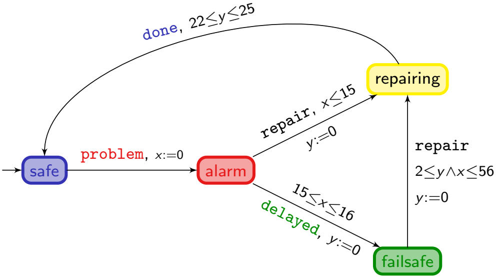
An example of a timed automaton



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe	
x	0		23		0		15.6		15.6	$\cdots$
y	0		23		23		38.6		0	

	failsafe	$\xrightarrow{2.3}$	failsafe
$\cdots$	15.6		17.9
	0		2.3

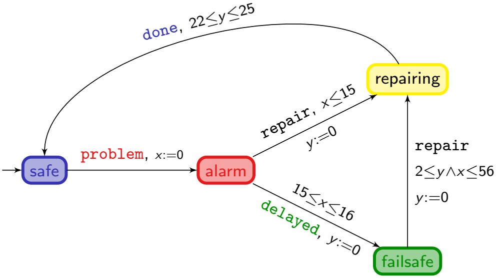
An example of a timed automaton



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe	
x	0		23		0		15.6		15.6	$\cdots$
y	0		23		23		38.6		0	

	failsafe	$\xrightarrow{2.3}$	failsafe	$\xrightarrow{\texttt{repair}}$	reparation
$\cdots$	15.6		17.9		17.9
	0		2.3		0

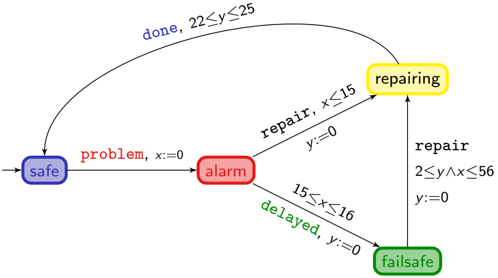
An example of a timed automaton



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe		
x	0		23		0		15.6		15.6	⋯
y	0		23		23		38.6		0	

	failsafe	$\xrightarrow{2.3}$	failsafe	$\xrightarrow{\texttt{repair}}$	reparation	$\xrightarrow{22.1}$	reparation
⋯	15.6		17.9		17.9		40
	0		2.3		0		22.1

An example of a timed automaton



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe	
x	0		23		0		15.6		15.6	⋯
y	0		23		23		38.6		0	

	failsafe	$\xrightarrow{2.3}$	failsafe	$\xrightarrow{\texttt{repair}}$	reparation	$\xrightarrow{22.1}$	reparation	$\xrightarrow{\texttt{done}}$	safe
⋯	15.6		17.9		17.9		40		40
	0		2.3		0		22.1		22.1

Timed automata

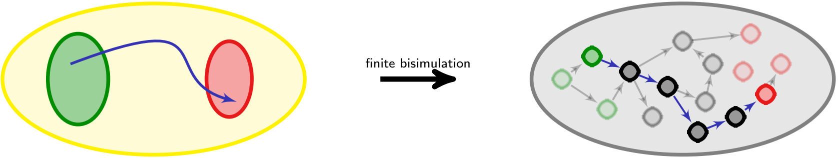
Theorem [AD90]

The reachability problem is decidable (and PSPACE-complete) for timed automata.

Timed automata

Theorem [AD90]

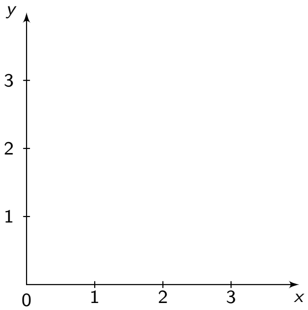
The reachability problem is decidable (and PSPACE-complete) for timed automata.



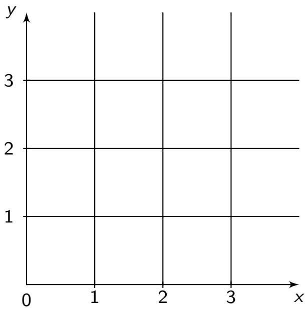
timed automaton

large (but finite) automaton
(region automaton)

The region abstraction

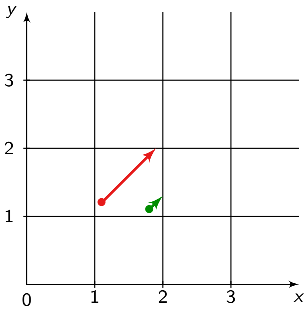


The region abstraction



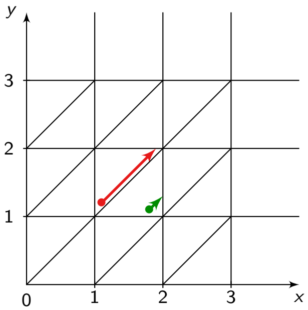
- "compatibility" between regions and constraints

The region abstraction



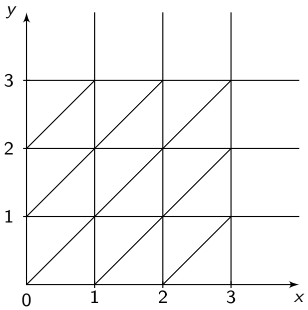
- "compatibility" between regions and constraints
- "compatibility" between regions and time elapsing

The region abstraction



- "compatibility" between regions and constraints
- "compatibility" between regions and time elapsing

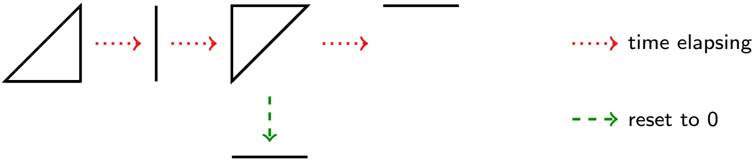
The region abstraction



- “compatibility” between regions and constraints
- “compatibility” between regions and time elapsing

☞ an equivalence of finite index
a time-abstract bisimulation

The region abstraction

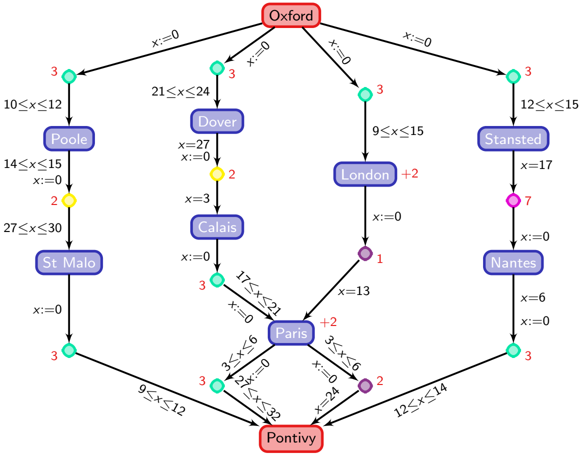


Time-optimal reachability

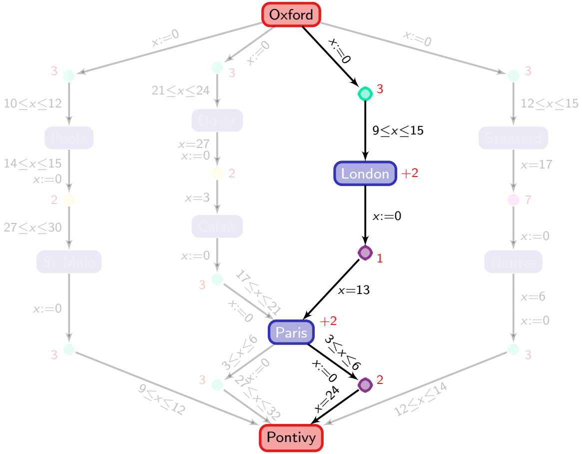
Theorem [CY92]

The time-optimal reachability problem is decidable (and PSPACE-complete) for timed automata.

A third model of the system



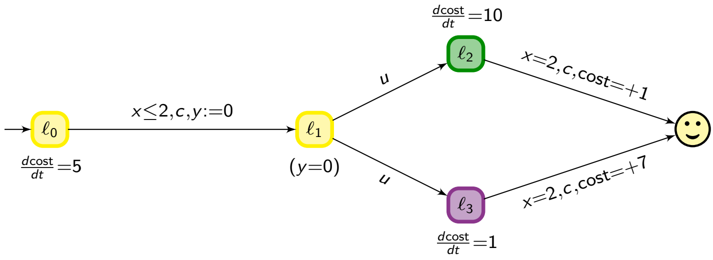
How much fuel will I use?



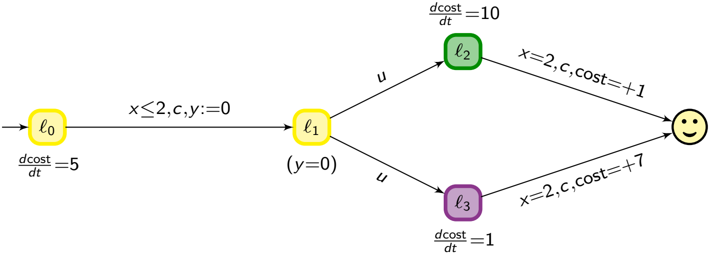
It is a *quantitative* (optimization) problem
in a priced timed automaton: at least 68 anti-planet units!

Outline

HSCC'01: weighted/timed automata



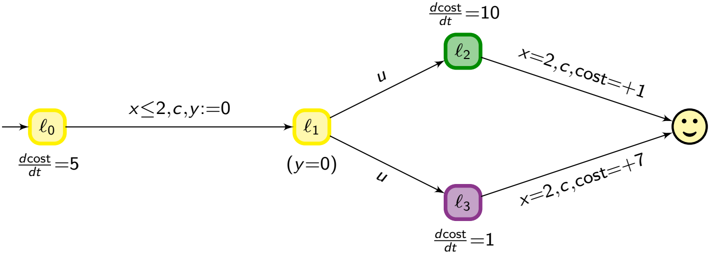
HSCC'01: weighted/timed automata



$$(\ell_0,(0,0)) \xrightarrow{1.3} (\ell_0,(1.3,1.3)) \xrightarrow{c} (\ell_1,(1.3,0)) \xrightarrow{u} (\ell_3,(1.3,0)) \xrightarrow{0.7} (\ell_3,(2,0.7)) \xrightarrow{c} ☺$$

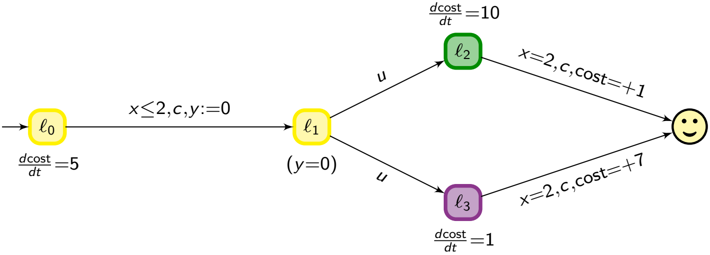
cost : $\quad 6.5 \quad + \quad 0 \quad + \quad 0 \quad + \quad 0.7 \quad + \quad 7 \quad = \quad 14.2$

HSCC'01: weighted/timed automata



Question: what is the optimal cost for reaching ☺?

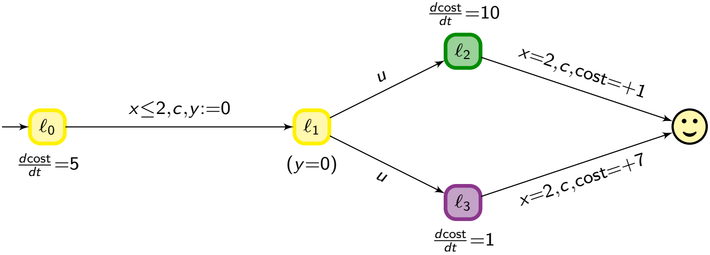
HSCC'01: weighted/timed automata



Question: what is the optimal cost for reaching ☺?

$$5t + 10(2-t) + 1 \;,\; 5t + (2-t) + 7$$

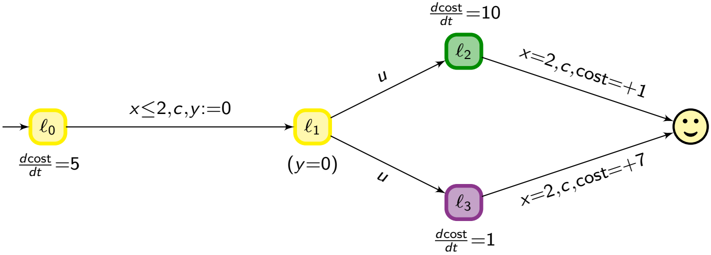
HSCC'01: weighted/timed automata



Question: what is the optimal cost for reaching ☺?

$$\min\ (\ 5t + 10(2-t) + 1\ ,\ 5t + (2-t) + 7\ )$$

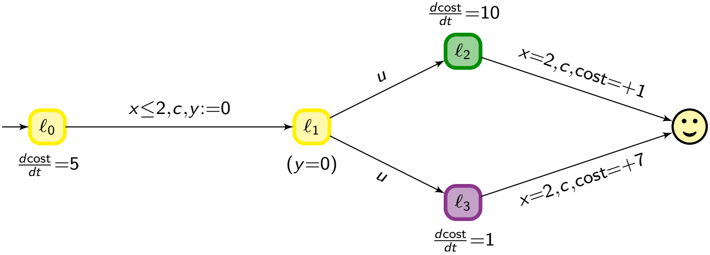
HSCC'01: weighted/timed automata



Question: what is the optimal cost for reaching ☺?

$$\inf_{0\leq t\leq 2} \min\left(\ 5t+10(2-t)+1\ ,\ 5t+(2-t)+7\ \right)=9$$

HSCC'01: weighted/timed automata



Question: what is the optimal cost for reaching ☺?

$$\inf_{0\leq t\leq 2} \min\ (\ 5t + 10(2-t) + 1\ ,\ 5t + (2-t) + 7\ ) = 9$$

➜ **strategy:** leave immediately ℓ_0, go to ℓ_3, and wait there 2 t.u.

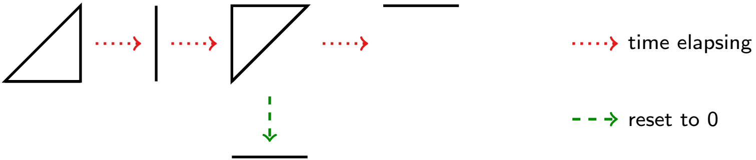
Optimal reachability

The idea "go through corners" extends in the general case.

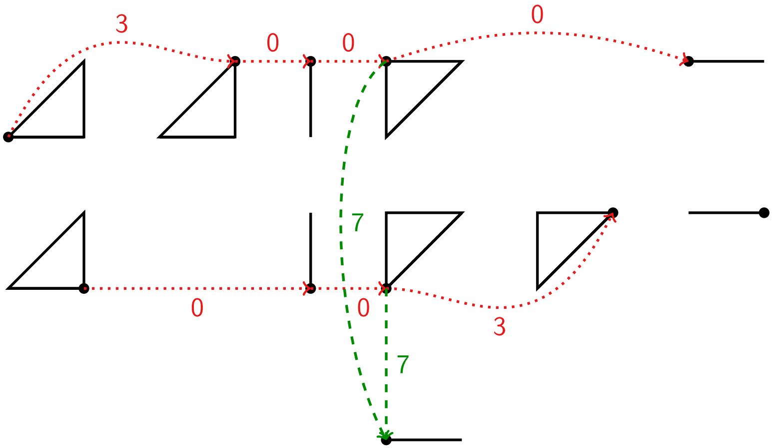
Theorem [ALP01,BFH+01,BBBR07]

Optimal reachability is decidable in timed automata.
It is PSPACE-complete.

The region abstraction is not fine enough



The corner-point abstraction



From timed to discrete behaviours

Optimal reachability as a linear programming problem

From timed to discrete behaviours

Optimal reachability as a linear programming problem

$$\circ \xrightarrow{t_1} \circ \xrightarrow{t_2} \circ \xrightarrow{t_3} \circ \xrightarrow{t_4} \circ \xrightarrow{t_5} \circ \quad \cdots$$

From timed to discrete behaviours

Optimal reachability as a linear programming problem

$$\circ \xrightarrow{t_1} \circ \xrightarrow[x\leq 2]{t_2} \circ \xrightarrow{t_3} \circ \xrightarrow{t_4} \circ \xrightarrow{t_5} \circ \quad \cdots \qquad \left\{ \begin{array}{l} t_1+t_2\leq 2 \\ \\ \end{array} \right.$$

From timed to discrete behaviours

Optimal reachability as a linear programming problem

$$\circ \xrightarrow[y:=0]{t_1} \circ \xrightarrow[x\leq 2]{t_2} \circ \xrightarrow{t_3} \circ \xrightarrow[y\geq 5]{t_4} \circ \xrightarrow{t_5} \circ \quad \cdots \qquad \begin{cases} t_1+t_2\leq 2 \\ t_2+t_3+t_4\geq 5 \end{cases}$$

From timed to discrete behaviours

Optimal reachability as a linear programming problem

$$\circ \xrightarrow[y:=0]{t_1} \circ \xrightarrow[x\leq 2]{t_2} \circ \xrightarrow{t_3} \circ \xrightarrow[y\geq 5]{t_4} \circ \xrightarrow{t_5} \circ \quad \cdots \qquad \begin{cases} t_1+t_2\leq 2 \\ t_2+t_3+t_4\geq 5 \end{cases}$$

Lemma

Let Z be a bounded zone and f be a function

$$f : (t_1, \ldots, t_n) \mapsto \sum_{i=1}^{n} c_i t_i + c$$

well-defined on $\overline{Z}$. Then $inf_Z f$ is obtained on the border of $\overline{Z}$ with integer coordinates.

From timed to discrete behaviours

Optimal reachability as a linear programming problem

$$\circ \xrightarrow[y:=0]{t_1} \circ \xrightarrow[x\leq 2]{t_2} \circ \xrightarrow{t_3} \circ \xrightarrow[y\geq 5]{t_4} \circ \xrightarrow{t_5} \circ \quad \cdots \qquad \begin{cases} t_1+t_2\leq 2 \\ t_2+t_3+t_4\geq 5 \end{cases}$$

Lemma

Let Z be a bounded zone and f be a function

$$f : (t_1, \ldots, t_n) \mapsto \sum_{i=1}^{n} c_i t_i + c$$

well-defined on $\overline{Z}$. Then $inf_Z f$ is obtained on the border of $\overline{Z}$ with integer coordinates.

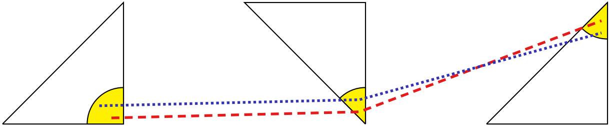
➜ for every finite path π in $\mathcal{A}$, there exists a path Π in $\mathcal{A}_{\mathrm{cp}}$ such that

$$\mathrm{cost}(\Pi) \leq \mathrm{cost}(\pi)$$

[Π is a "corner-point projection" of π]

From discrete to timed behaviours

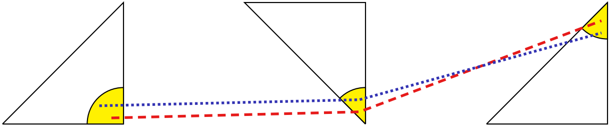
Approximation of abstract paths:



For any path Π of $\mathcal{A}_{\mathsf{cp}}$,

From discrete to timed behaviours

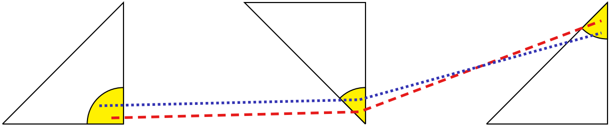
Approximation of abstract paths:



For any path Π of $\mathcal{A}_{\mathsf{cp}}$, for any $\varepsilon > 0$,

From discrete to timed behaviours

Approximation of abstract paths:

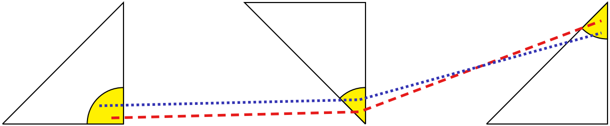


For any path Π of $\mathcal{A}_{\mathsf{cp}}$, for any $\varepsilon > 0$, there exists a path π_ε of $\mathcal{A}$ s.t.

$$\|\Pi - \pi_\varepsilon\|_\infty < \varepsilon$$

From discrete to timed behaviours

Approximation of abstract paths:



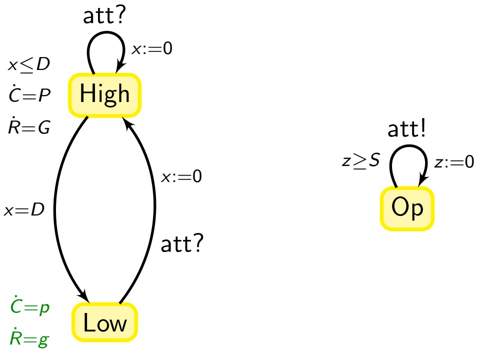
For any path Π of $\mathcal{A}_{\mathsf{cp}}$, for any $\varepsilon > 0$, there exists a path π_ε of $\mathcal{A}$ s.t.

$$\|\Pi - \pi_\varepsilon\|_\infty < \varepsilon$$

For every $\eta > 0$, there exists $\varepsilon > 0$ s.t.

$$\|\Pi - \pi_\varepsilon\|_\infty < \varepsilon \Rightarrow |\mathsf{cost}(\Pi) - \mathsf{cost}(\pi_\varepsilon)| < \eta$$

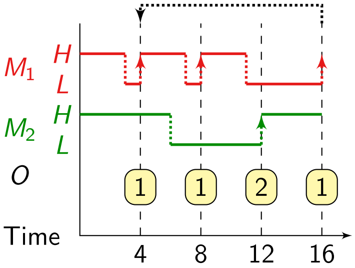
Mean-Cost Optimization



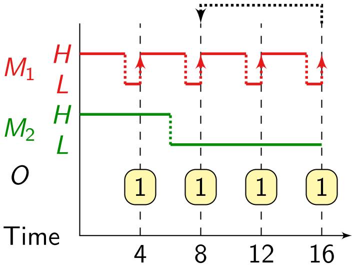
Question: How to minimize $\lim_{n\to+\infty} \frac{\text{accumulated cost}(\pi_n)}{\text{accumulated reward}(\pi_n)}$?

An example

Two machines $M_1(D = 3, P = 3, G = 4, p = 5, g = 3)$ and $M_2(D = 6, P = 3, G = 2, p = 5, g = 2)$.
An operator $O(4)$.



Schedule with ratio 1.455



Schedule with ratio 1.478

Mean-cost optimization

Theorem [BBL04,BBL08]

The mean-cost optimization problem is decidable (and PSPACE-complete) for priced timed automata.

☞ The corner-point abstraction is sound and complete.

From timed to discrete behaviours

➜ **Finite behaviours:** based on the following property

Lemma

Let Z be a bounded zone and f be a function

$$f : (t_1, \ldots, t_n) \mapsto \frac{\sum_{i=1}^{n} c_i t_i + c}{\sum_{i=1}^{n} r_i t_i + r}$$

well-defined on $\overline{Z}$. Then $inf_Z f$ is obtained on the border of $\overline{Z}$ with integer coordinates.

From timed to discrete behaviours

➜ **Finite behaviours:** based on the following property

Lemma

Let Z be a bounded zone and f be a function

$$f : (t_1, \ldots, t_n) \mapsto \frac{\sum_{i=1}^{n} c_i t_i + c}{\sum_{i=1}^{n} r_i t_i + r}$$

well-defined on $\overline{Z}$. Then $\mathit{inf}_Z f$ is obtained on the border of $\overline{Z}$ with integer coordinates.

➜ for every finite path π in $\mathcal{A}$, there exists a path Π in $\mathcal{A}_{\text{cp}}$ such that mean-cost$(\Pi) \leq$ mean-cost(π)

From timed to discrete behaviours

➜ **Finite behaviours:** based on the following property

Lemma

Let Z be a bounded zone and f be a function

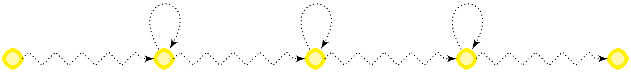
$$f : (t_1, \dots, t_n) \mapsto \frac{\sum_{i=1}^{n} c_i t_i + c}{\sum_{i=1}^{n} r_i t_i + r}$$

well-defined on $\overline{Z}$. Then $inf_Z f$ is obtained on the border of $\overline{Z}$ with integer coordinates.

➜ for every finite path π in $\mathcal{A}$, there exists a path Π in $\mathcal{A}_{\text{cp}}$ such that

$$\text{mean-cost}(\Pi) \leq \text{mean-cost}(\pi)$$

- **Infinite behaviours:** decompose each sufficiently long projection into cycles



The linear part will be negligible!

From timed to discrete behaviours

➜ **Finite behaviours:** based on the following property

Lemma

Let Z be a bounded zone and f be a function

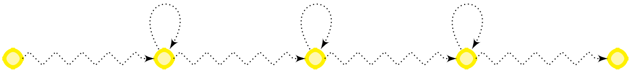
$$f : (t_1, \ldots, t_n) \mapsto \frac{\sum_{i=1}^{n} c_i t_i + c}{\sum_{i=1}^{n} r_i t_i + r}$$

well-defined on $\overline{Z}$. Then $inf_Z f$ is obtained on the border of $\overline{Z}$ with integer coordinates.

➜ for every finite path π in $\mathcal{A}$, there exists a path Π in $\mathcal{A}_{\mathrm{cp}}$ such that

$$\text{mean-cost}(\Pi) \leq \text{mean-cost}(\pi)$$

- **Infinite behaviours:** decompose each sufficiently long projection into cycles

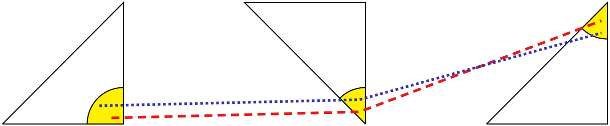


The linear part will be negligible!

➜ the optimal cycle of $\mathcal{A}_{\mathrm{cp}}$ is better than any infinite path of $\mathcal{A}$!

From discrete to timed behaviours

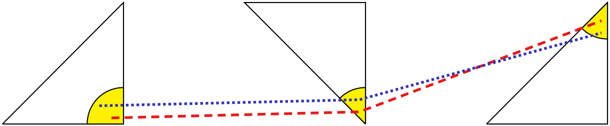
Approximation of abstract paths:



For any path Π of $\mathcal{A}_{\mathsf{cp}}$,

From discrete to timed behaviours

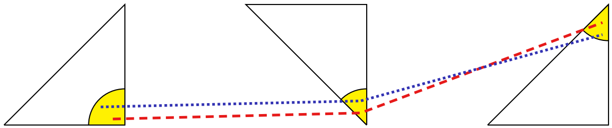
Approximation of abstract paths:



For any path Π of $\mathcal{A}_{\mathsf{cp}}$, for any $\varepsilon > 0$,

From discrete to timed behaviours

Approximation of abstract paths:

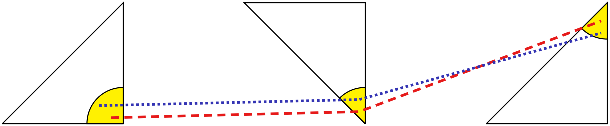


For any path Π of $\mathcal{A}_{\mathsf{cp}}$, for any $\varepsilon > 0$, there exists a path π_ε of $\mathcal{A}$ s.t.

$$\|\Pi - \pi_\varepsilon\|_\infty < \varepsilon$$

From discrete to timed behaviours

Approximation of abstract paths:



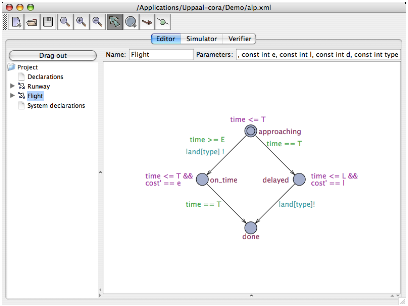
For any path Π of $\mathcal{A}_{\mathsf{cp}}$, for any $\varepsilon > 0$, there exists a path π_ε of $\mathcal{A}$ s.t.

$$\|\Pi - \pi_\varepsilon\|_\infty < \varepsilon$$

For every $\eta > 0$, there exists $\varepsilon > 0$ s.t.

$$\|\Pi - \pi_\varepsilon\|_\infty < \varepsilon \Rightarrow |\text{mean-cost}(\Pi) - \text{mean-cost}(\pi_\varepsilon)| < \eta$$

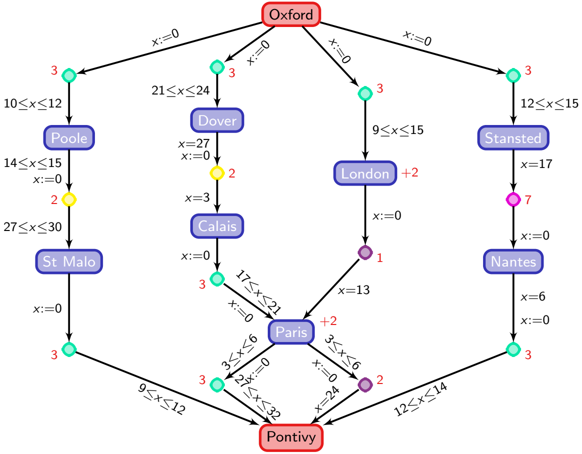
Uppaal Cora



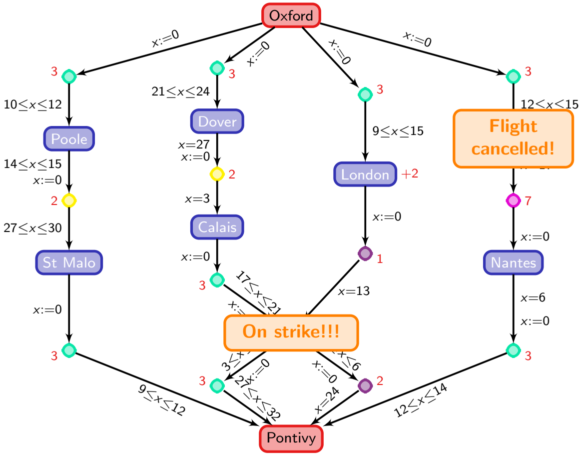
A branch of Uppaal for cost optimal reachability

Outline

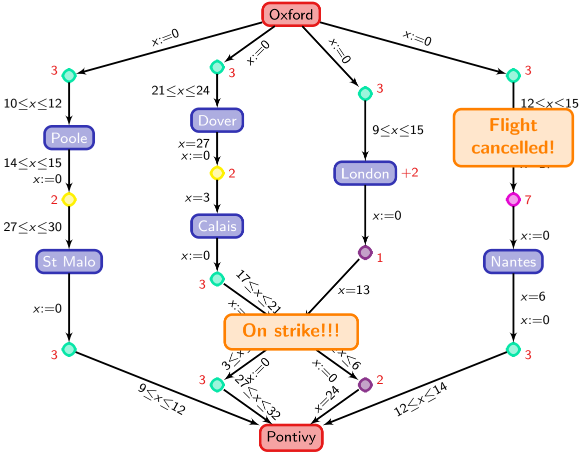
What if an unexpected event happens?



What if an unexpected event happens?

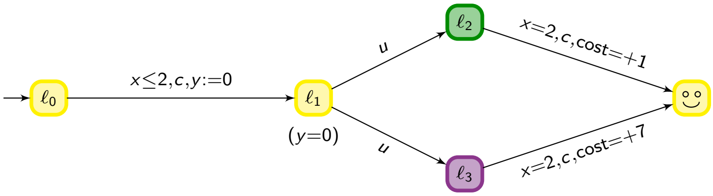


What if an unexpected event happens?

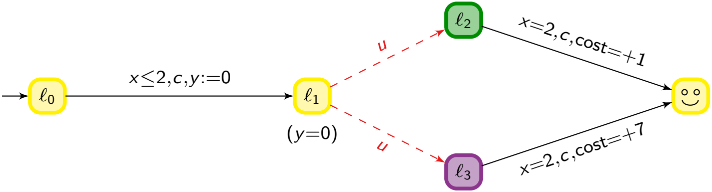


☞ modelled as timed games

A simple example of timed games



A simple example of timed games



Decidability of timed games

Theorem [AMPS98] [HK99]

Safety and reachability control in timed automata are decidable and EXPTIME-complete.

Decidability of timed games

Theorem [AMPS98] [HK99]

Safety and reachability control in timed automata are decidable and EXPTIME-complete.

(the attractor is computable...)

Decidability of timed games

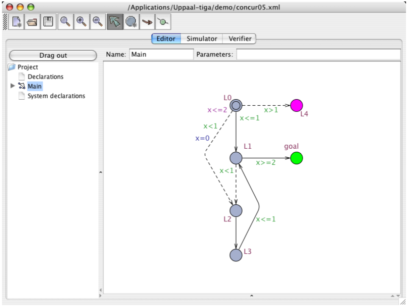
Theorem [AMPS98] [HK99]

Safety and reachability control in timed automata are decidable and EXPTIME-complete.

(the attractor is computable...)

☞ classical regions are sufficient for solving such problems

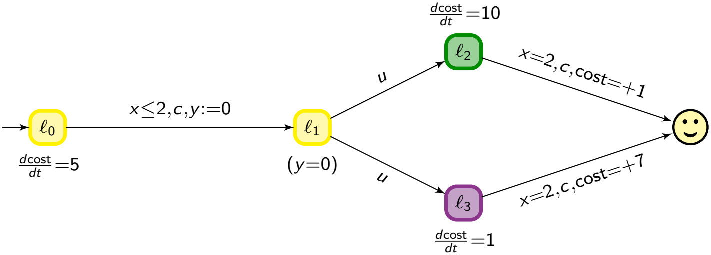
Uppaal Tiga



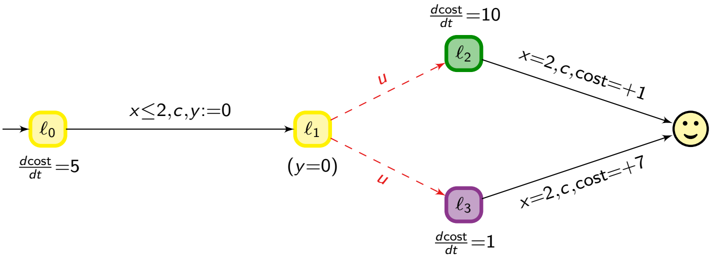
A forward on-the-fly algorithm for solving reachability timed games

☞ implemented as a branch of Uppaal

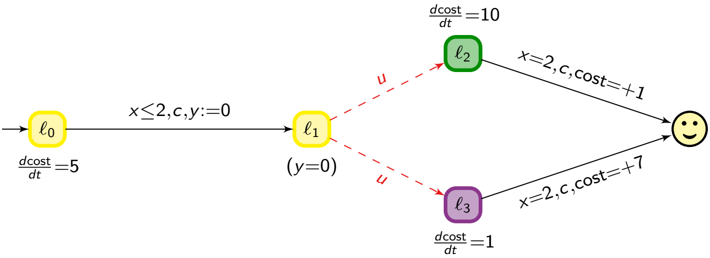
Back to the simple example



Back to the simple example

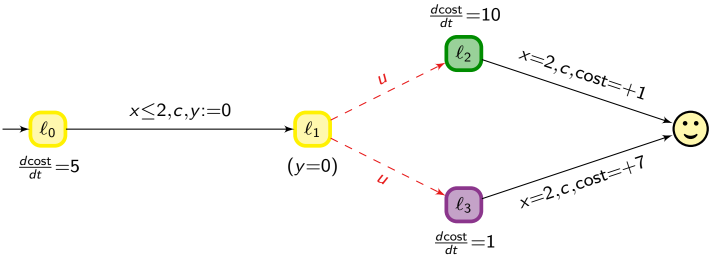


Back to the simple example



Question: what is the optimal cost we can ensure in state ℓ_0?

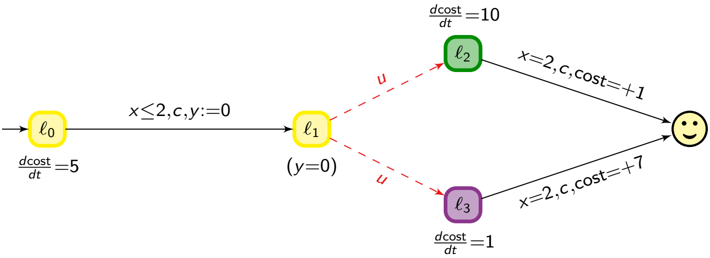
Back to the simple example



Question: what is the optimal cost we can ensure in state ℓ_0?

$$5t + 10(2-t) + 1 \;,\; 5t + (2-t) + 7$$

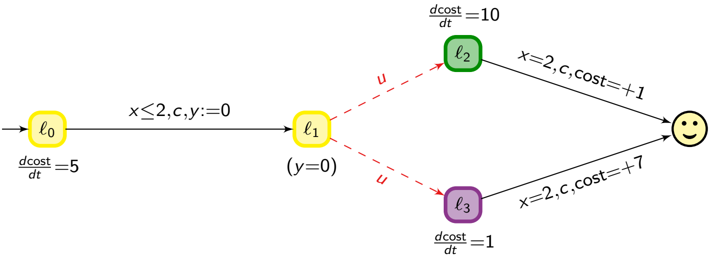
Back to the simple example



Question: what is the optimal cost we can ensure in state ℓ_0?

$$\max\ (\ 5t + 10(2-t) + 1\ ,\ 5t + (2-t) + 7\ )$$

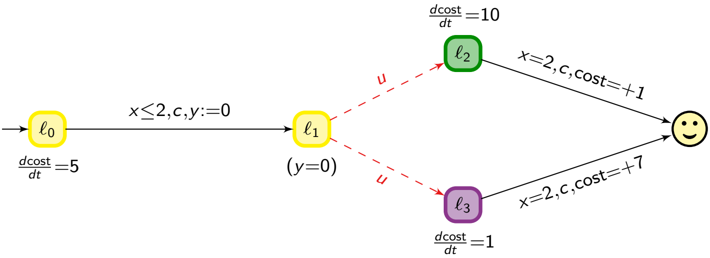
Back to the simple example



Question: what is the optimal cost we can ensure in state ℓ_0?

$$\inf_{0 \leq t \leq 2} \max\left(5t + 10(2-t) + 1 \, , \, 5t + (2-t) + 7 \right) = 14 + \frac{1}{3}$$

Back to the simple example

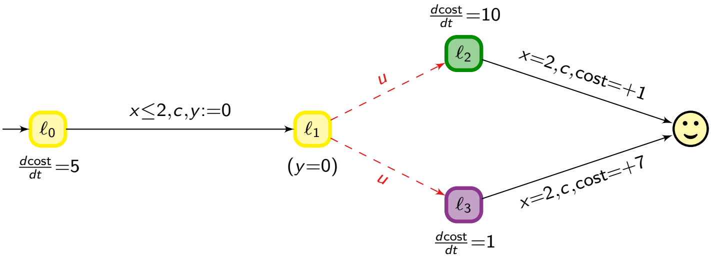


Question: what is the optimal cost we can ensure in state ℓ_0?

$$\inf_{0\leq t\leq 2} \max\left(\, 5t+10(2-t)+1 \,,\, 5t+(2-t)+7 \,\right) = 14+\frac{1}{3}$$

➜ **strategy:** wait in ℓ_0, and when $t=\frac{4}{3}$, go to ℓ_1

Back to the simple example



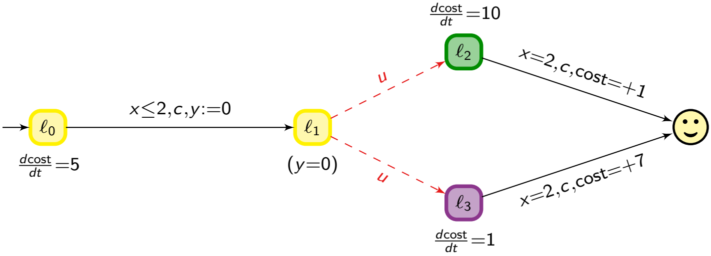
Question: what is the optimal cost we can ensure in state ℓ_0?

$$\inf_{0\le t\le 2} \max\left(\, 5t + 10(2-t) + 1 \,,\, 5t + (2-t) + 7 \,\right) = 14 + \frac{1}{3}$$

➜ **strategy:** wait in ℓ_0, and when $t = \frac{4}{3}$, go to ℓ_1

- How to automatically compute such optimal costs?

Back to the simple example



Question: what is the optimal cost we can ensure in state ℓ_0?

$$\inf_{0 \leq t \leq 2} \max\left(5t + 10(2-t) + 1 \,,\, 5t + (2-t) + 7 \right) = 14 + \frac{1}{3}$$

➜ **strategy:** wait in ℓ_0, and when $t = \frac{4}{3}$, go to ℓ_1

- How to automatically compute such optimal costs?
- How to synthesize optimal strategies (if one exists)?

A fairly hot topic!

- optimal time is computable in timed games [AM99]

A fairly hot topic!

- optimal time is computable in timed games [AM99]
- case of acyclic games [LMM02]

A fairly hot topic!

- optimal time is computable in timed games [AM99]
- case of acyclic games [LMM02]
- general case [ABM04]
  - complexity of k-step games
  - under a strongly non-*zeno* assumption, optimal cost is computable

A fairly hot topic!

- optimal time is computable in timed games [AM99]
- case of acyclic games [LMM02]
- general case [ABM04]
  - complexity of *k*-step games
  - under a strongly non-*zeno* assumption, optimal cost is computable
- general case [BCFL04]
  - structural properties of strategies (e.g. memory)
  - under a strongly non-*zeno* assumption, optimal cost is computable

A fairly hot topic!

- general case [BBR05]
  - with five clocks, optimal cost is not computable!
  - with one clock and one stopwatch cost, optimal cost is computable

A fairly hot topic!

- general case [BBR05]
  - with five clocks, optimal cost is not computable!
  - with one clock and one stopwatch cost, optimal cost is computable
- general case [BBM06]
  - with three clocks, optimal cost is not computable

A fairly hot topic!

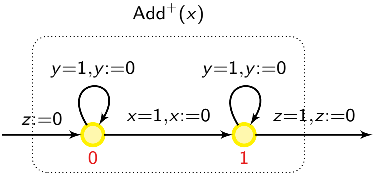
- general case [BBR05]
  - with five clocks, optimal cost is not computable!
  - with one clock and one stopwatch cost, optimal cost is computable
- general case [BBM06]
  - with three clocks, optimal cost is not computable
- the single-clock case [BLMR06]
  - with one clock, optimal cost is computable

Why is that hard?

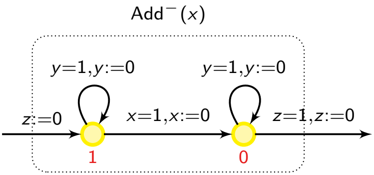
Given two clocks x and y, we can check whether $y = 2x$

Why is that hard?

Given two clocks x and y, we can check whether $y = 2x$



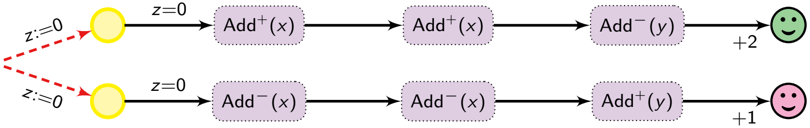
The cost is increased by x_0



The cost is increased by $1-x_0$

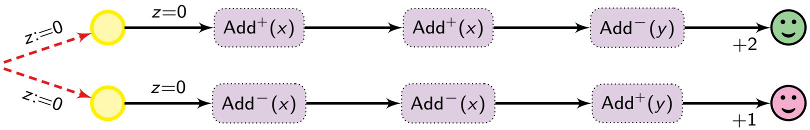
Why is that hard?

Given two clocks x and y, we can check whether $y = 2x$



Why is that hard?

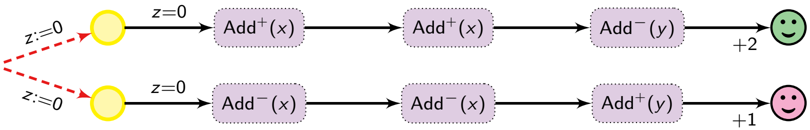
Given two clocks x and y, we can check whether $y = 2x$



- In , cost $= 2x_0 + (1 - y_0) + 2$

Why is that hard?

Given two clocks x and y, we can check whether $y = 2x$

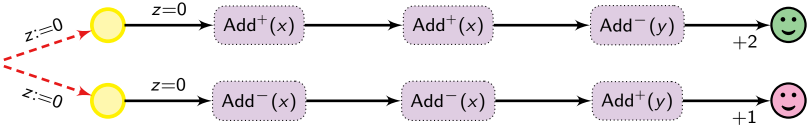


- In , cost $= 2x_0 + (1 - y_0) + 2$

  In , cost $= 2(1 - x_0) + y_0 + 1$

Why is that hard?

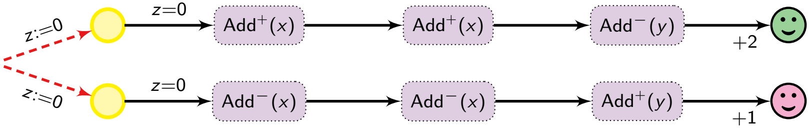
Given two clocks x and y, we can check whether $y = 2x$



- In , cost $= 2x_0 + (1 - y_0) + 2$
  In , cost $= 2(1 - x_0) + y_0 + 1$
- if $y_0 < 2x_0$, player 2 chooses the first branch: cost > 3

Why is that hard?

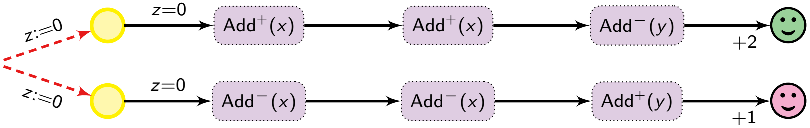
Given two clocks x and y, we can check whether $y = 2x$



- In , cost $= 2x_0 + (1 - y_0) + 2$
  In , cost $= 2(1 - x_0) + y_0 + 1$
- if $y_0 < 2x_0$, player 2 chooses the first branch: cost > 3
  if $y_0 > 2x_0$, player 2 chooses the second branch: cost > 3

Why is that hard?

Given two clocks x and y, we can check whether $y = 2x$

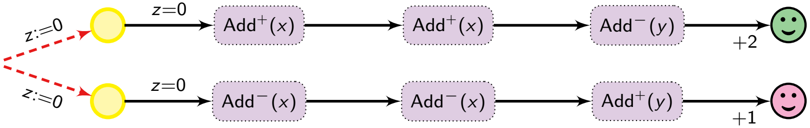


- In , cost $= 2x_0 + (1 - y_0) + 2$
  In , cost $= 2(1 - x_0) + y_0 + 1$

- if $y_0 < 2x_0$, player 2 chooses the first branch: cost > 3
  if $y_0 > 2x_0$, player 2 chooses the second branch: cost > 3
  if $y_0 = 2x_0$, in both branches, cost $= 3$

Why is that hard?

Given two clocks x and y, we can check whether $y = 2x$



- In , cost $= 2x_0 + (1 - y_0) + 2$
  In , cost $= 2(1 - x_0) + y_0 + 1$
- if $y_0 < 2x_0$, player 2 chooses the first branch: cost > 3
  if $y_0 > 2x_0$, player 2 chooses the second branch: cost > 3
  if $y_0 = 2x_0$, in both branches, cost $= 3$
- Player 1 has a winning strategy with cost ≤ 3 iff $y_0 = 2x_0$

Outline

Conclusion

Priced timed automata, a model and framework to represent quantitative constraints on timed systems:

- several interesting optimization problems

Conclusion

Priced timed automata, a model and framework to represent quantitative constraints on timed systems:

- several interesting optimization problems

Not mentioned here

- all works on model-checking issues (extensions of CTL, LTL)
  - very few decidability results

    [BBR04,BBM06,BLM07,BM07]
- discounted cost

Conclusion

Priced timed automata, a model and framework to represent quantitative constraints on timed systems:

- several interesting optimization problems

Not mentioned here

- all works on model-checking issues (extensions of CTL, LTL)
  - very few decidability results

[BBR04,BBM06,BLM07,BM07]

- discounted cost

Further work

- approximate optimal timed games to circumvent undecidability results