Energy management in timed systems

Patricia Bouyer-Decitre

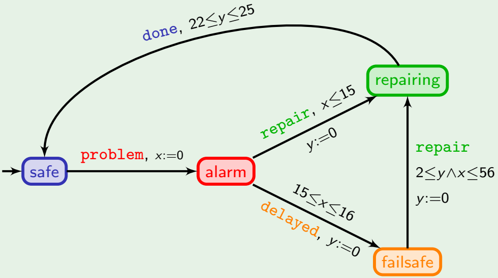
LSV, CNRS & ENS Cachan, France

Based on joint works with Uli Fahrenberg, Kim G. Larsen, Nicolas Markey and Jiří Srba

Outline

The standard timed automaton model [AD90,AD94]

Example



	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	

[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

Modelling resources in timed systems

- System resources might be relevant and even crucial information

Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - bandwidth,
  - ...
  - price to pay,
  - benefits,
  - temperature,

Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - bandwidth,
  - price to pay,
  - benefits,
  - temperature,
  - ...

$\leadsto$ timed automata are not powerful enough!

Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - bandwidth,
  - ...
  - price to pay,
  - benefits,
  - temperature,

  $\leadsto$ timed automata are not powerful enough!

- A possible solution: use hybrid automata

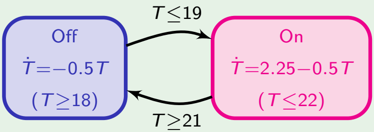
Modelling resources in timed systems

- System resources might be relevant and even crucial information
    - energy consumption,
    - memory usage,
    - bandwidth,
    - price to pay,
    - benefits,
    - temperature,
    - ...

  $\rightsquigarrow$ timed automata are not powerful enough!

- A possible solution: use hybrid automata

The thermostat example



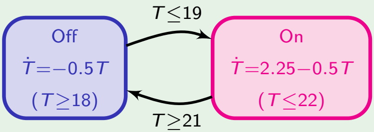
Modelling resources in timed systems

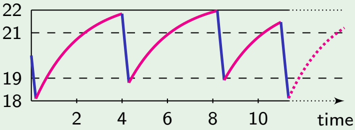
- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - bandwidth,
  - ...
  - price to pay,
  - benefits,
  - temperature,

  $\rightsquigarrow$ timed automata are not powerful enough!

- A possible solution: use hybrid automata

The thermostat example





Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - bandwidth,
  - price to pay,
  - benefits,
  - temperature,
  - ...

$\rightsquigarrow$ timed automata are not powerful enough!

- A possible solution: use hybrid automata

Theorem [HKPV95]

The reachability problem is undecidable in hybrid automata.

[HKPV95] Henzinger, Kopke, Puri, Varaiya. What's decidable wbout hybrid automata? *(SToC'95)*.

Modelling resources in timed systems

- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - bandwidth,
  - price to pay,
  - benefits,
  - temperature,
  - ...

$\rightsquigarrow$ timed automata are not powerful enough!

- A possible solution: use hybrid automata

Theorem [HKPV95]

The reachability problem is undecidable in hybrid automata.

- An alternative: priced/weighted timed automata [ALP01,BFH+01]

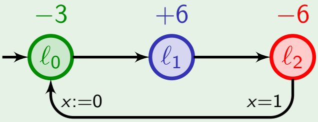
$\rightsquigarrow$ hybrid variables are observer variables
(they do not constrain *a priori* the system)

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

A simple example of weighted timed automata (WTA)

[ALP01,BFH+01]

Example (with a linear observer)



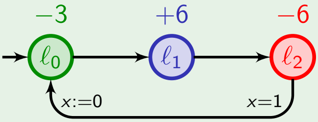
Run $(\ell_0, 0) \xrightarrow{\text{delay}(\frac{1}{6})} (\ell_0, \frac{1}{6}) \rightarrow (\ell_1, \frac{1}{6}) \xrightarrow{\text{delay}(\frac{1}{2})} (\ell_1, \frac{2}{3}) \rightarrow (\ell_2, \frac{2}{3}) \ldots$

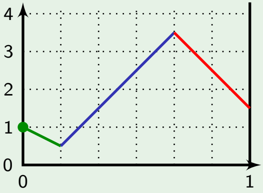
[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

A simple example of weighted timed automata (WTA)

[ALP01,BFH+01]

Example (with a linear observer)

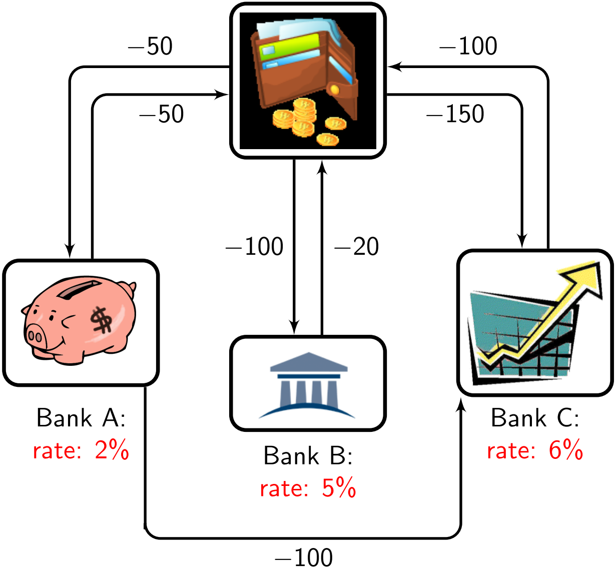




Run $(\ell_0, 0) \xrightarrow{\text{delay}(\frac{1}{6})} (\ell_0, \frac{1}{6}) \to (\ell_1, \frac{1}{6}) \xrightarrow{\text{delay}(\frac{1}{2})} (\ell_1, \frac{2}{3}) \to (\ell_2, \frac{2}{3}) \ldots$

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

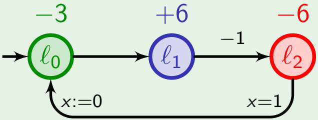
Beyond linear observers...



Beyond linear observers...

Example

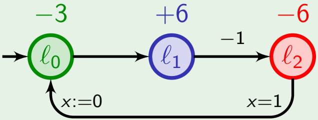
We also consider PTA with an exponential observer:



Beyond linear observers...

Example

We also consider PTA with an exponential observer:



Rate -3 in location ℓ_0 means

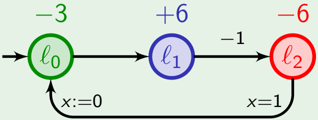
$$\frac{\partial\ \text{cost}}{\partial\ \text{time}} = -3 \times \text{cost}$$

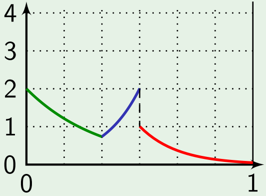
$$\text{cost} = \text{cost}_0 \cdot e^{-3 \times t}$$

Beyond linear observers...

Example

We also consider PTA with an exponential observer:





Rate -3 in location ℓ_0 means

$$\frac{\partial \text{ cost}}{\partial \text{ time}} = -3 \times \text{cost}$$

$$\text{cost} = \text{cost}_0 \cdot e^{-3 \times t}$$

Relevant questions

- Various optimization questions (optimal reachability, optimal mean-cost or discounted infinite schedules, *etc*)
  - $\rightsquigarrow$ an abundant literature since 2001 (for the linear observers only)
  - $\rightsquigarrow$ *cf* tutorial of Kim G. Larsen

Relevant questions

- Various optimization questions (optimal reachability, optimal mean-cost or discounted infinite schedules, *etc*)
  - $\leadsto$ an abundant literature since 2001 (for the linear observers only)
  - $\leadsto$ *cf* tutorial of Kim G. Larsen

- Scheduling under energy constraints (resource management): are there scheduling policies/strategies when energy is constrained?

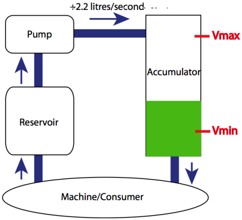
  [BFLMS08]

[BFLMS08] Bouyer, Fahrenberg, Larsen, Markey, Srba. Infinite runs in weighted timed automata with energy constraints *(FORMATS'08)*.

Relevant questions

- Various optimization questions (optimal reachability, optimal mean-cost or discounted infinite schedules, *etc*)
  - $\rightsquigarrow$ an abundant literature since 2001 (for the linear observers only)
  - $\rightsquigarrow$ *cf* tutorial of Kim G. Larsen

- Scheduling under energy constraints (resource management): are there scheduling policies/strategies when energy is constrained? [BFLMS08]
  - $\rightsquigarrow$ *An example:* an oil pump control system [CJL+09]

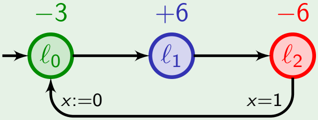


[BFLMS08] Bouyer, Fahrenberg, Larsen, Markey, Srba. Infinite runs in weighted timed automata with energy constraints *(FORMATS'08)*.
[CJL+09] Cassez, Jessen, Larsen, Raskin, Reynier. Automatic synthesis of robust and optimal controllers - An industrial case study *(HSCC'09)*.

Outline

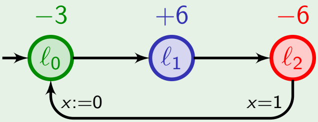
Scheduling/feasible runs under energy constraints

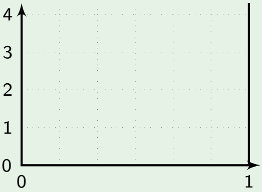
Example



Scheduling/feasible runs under energy constraints

Example



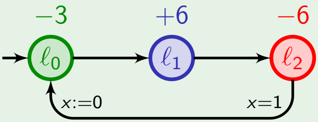


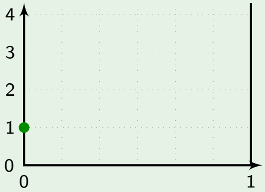
"energy is $\geq$0"

- Lower-bound problem (**L**)

Scheduling/feasible runs under energy constraints

Example



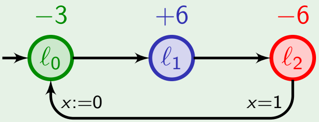


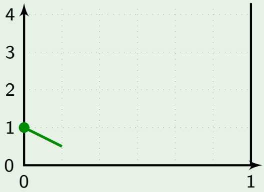
"energy is ≥ 0"

- Lower-bound problem (**L**)

Scheduling/feasible runs under energy constraints

Example



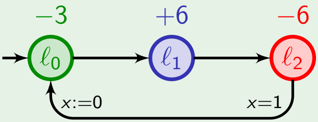


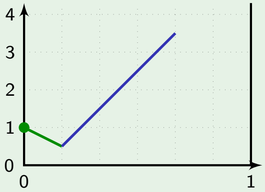
"energy is ≥ 0"

- Lower-bound problem (**L**)

Scheduling/feasible runs under energy constraints

Example



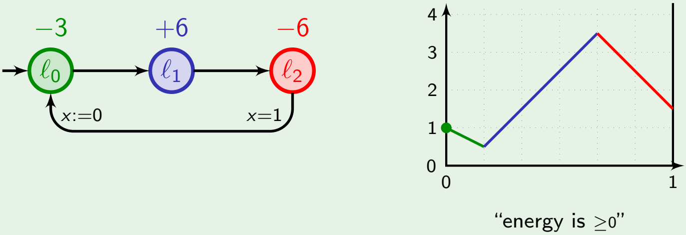


"energy is $\geq$0"

- Lower-bound problem (**L**)

Scheduling/feasible runs under energy constraints

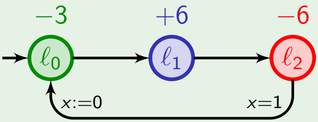
Example

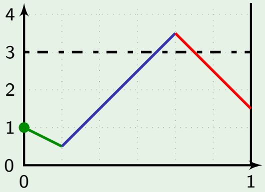


- Lower-bound problem (**L**)

Scheduling/feasible runs under energy constraints

Example



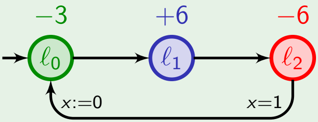


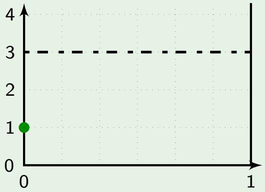
"energy is in [0,3]"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



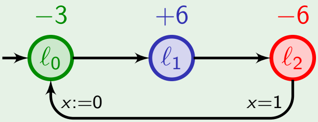


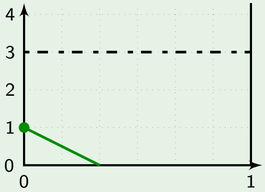
"energy is in $[0,3]$"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



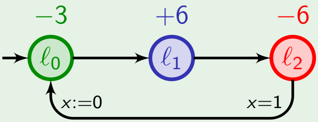


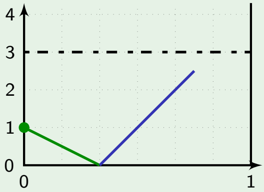
"energy is in $[0,3]$"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



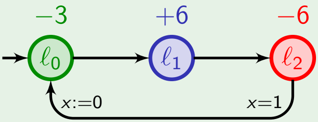


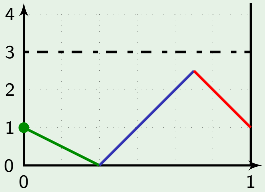
"energy is in $[0,3]$"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



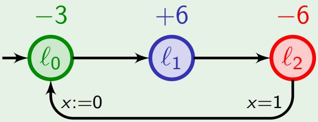


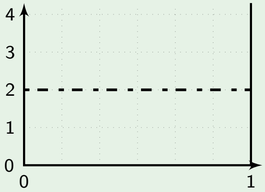
"energy is in [0,3]"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



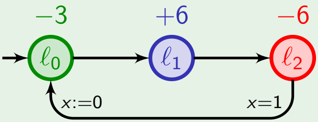


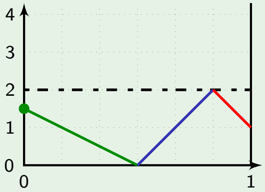
"energy is in [0,2]"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



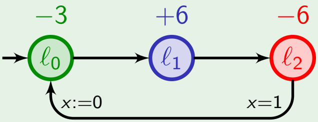


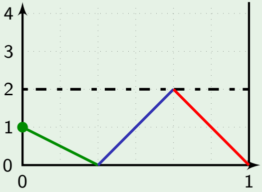
"energy is in [0,2]"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



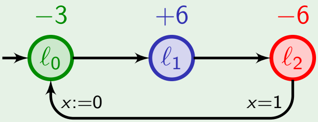


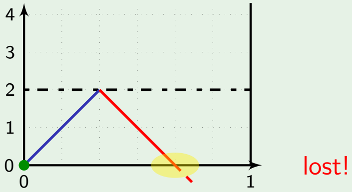
"energy is in [0,2]"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



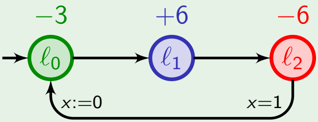


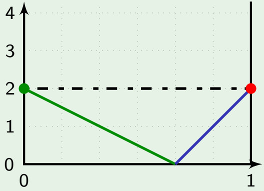
"energy is in [0,2]"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



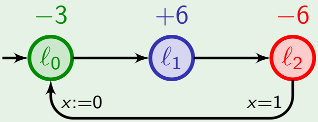


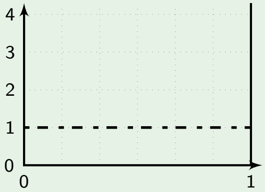
"energy is in [0,2]"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



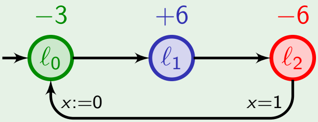


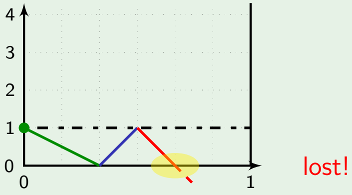
"energy is in $[0,1]$"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example



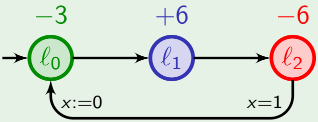


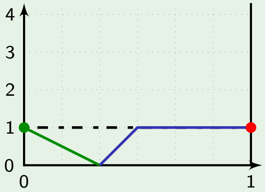
"energy is in $[0,1]$"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)

Scheduling/feasible runs under energy constraints

Example





"energy is in $[0,1]$ with a weak upper bound"

- Lower-bound problem (**L**)
- Lower-and-upper-bound problem (**L+U**)
- Lower-and-weak-upper-bound problem (**L+W**)

Results for the untimed case (only discrete costs)

is there a *feasible* infinite run? — are all runs *feasible*? — is there a winning strategy?

	exist. problem	**univ. problem**	**games**
L	$\in$ P	$\in$ P	$\in$ UP $\cap$ coUP P-hard
L+W	$\in$ P	$\in$ P	$\in$ NP $\cap$ coNP P-hard
L+U	$\in$ PSPACE NP-hard	$\in$ P	EXPTIME-c.

[BFLMS08] Bouyer, Fahrenberg, Larsen, Markey, Srba. Infinite runs in weighted timed automata with energy constraints *(FORMATS'08)*.

Results for the 1-clock case (linear observer)

	exist. problem	univ. problem	games
L	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+W	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+U	?	?	undecidable

[BFLMS08] Bouyer, Fahrenberg, Larsen, Markey, Srba. Infinite runs in weighted timed automata with energy constraints *(FORMATS'08)*.
[BFLM10] Bouyer, Fahrenberg, Larsen, Markey. Timed automata with observers under energy constraints *(HSCC'10)*.

Results for the general (n-clock) case (linear observer)

	exist. problem	**univ. problem**	**games**
L	?	?	?
L+W	?	?	?
L+U	?	?	undecidable

[BFLMS08] Bouyer, Fahrenberg, Larsen, Markey, Srba. Infinite runs in weighted timed automata with energy constraints *(FORMATS'08)*.

Outline

Outline

The results in the single-clock framework

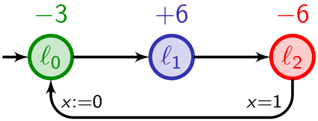
	exist. problem	**univ. problem**	**games**
L	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+W	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+U	?	?	undecidable

The results in the single-clock framework

	exist. problem	**univ. problem**	**games**
L	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+W	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+U	?	?	undecidable

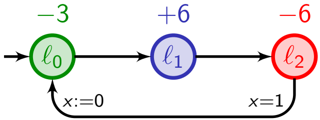
L+U-problem and fixpoint computation

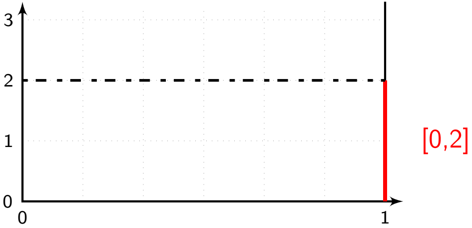
The backward fixpoint computation, which is correct in the limit, does not terminate in general.



L+U-problem and fixpoint computation

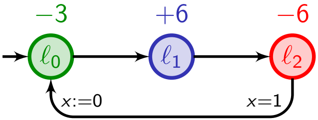
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

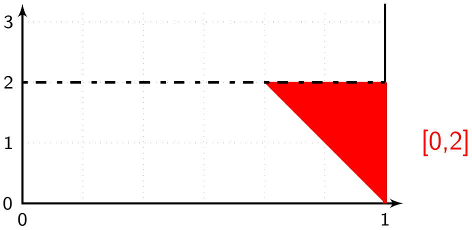




L+U-problem and fixpoint computation

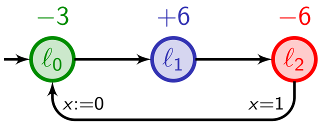
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

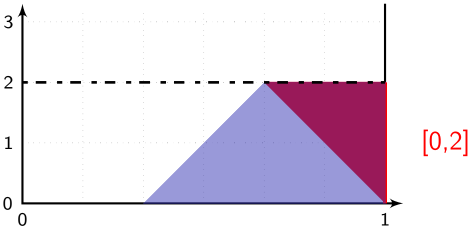




L+U-problem and fixpoint computation

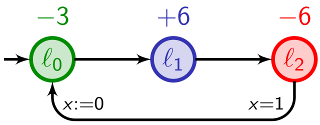
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

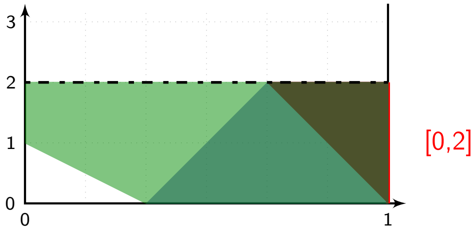




L+U-problem and fixpoint computation

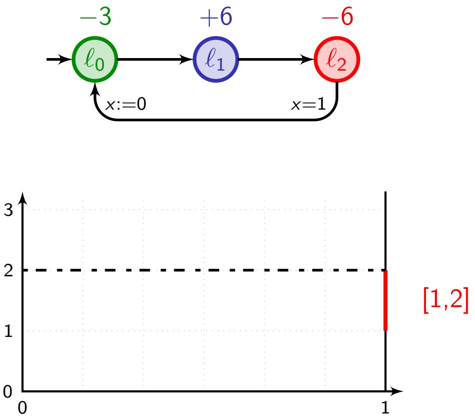
The backward fixpoint computation, which is correct in the limit, does not terminate in general.





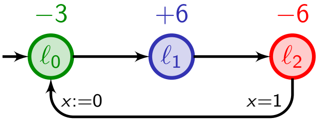
L+U-problem and fixpoint computation

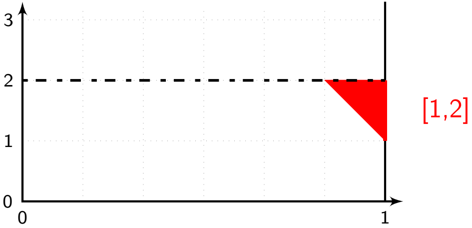
The backward fixpoint computation, which is correct in the limit, does not terminate in general.



L+U-problem and fixpoint computation

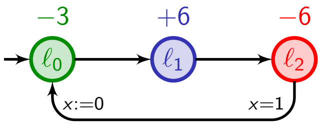
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

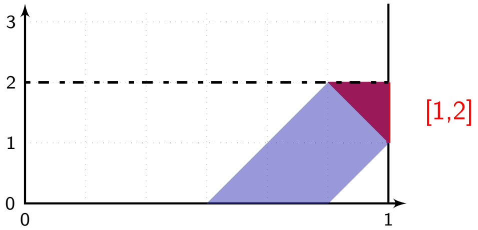




L+U-problem and fixpoint computation

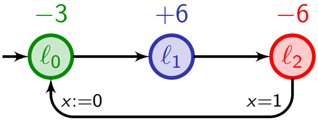
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

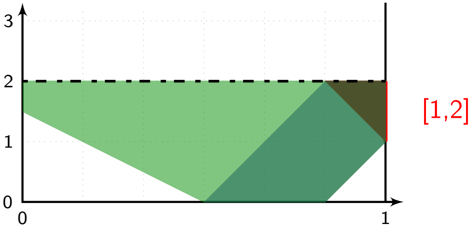




L+U-problem and fixpoint computation

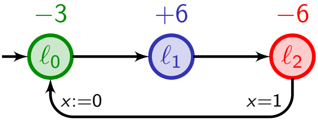
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

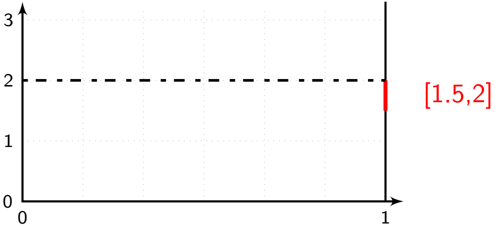




L+U-problem and fixpoint computation

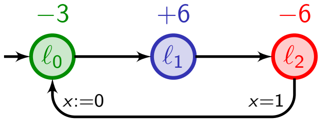
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

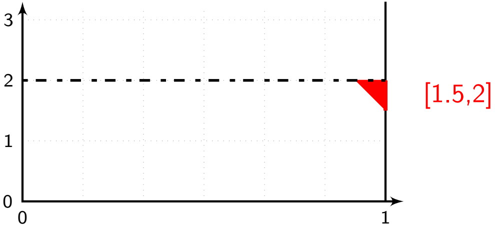




L+U-problem and fixpoint computation

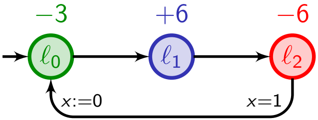
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

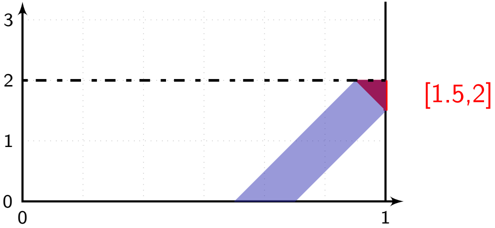




L+U-problem and fixpoint computation

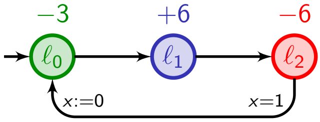
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

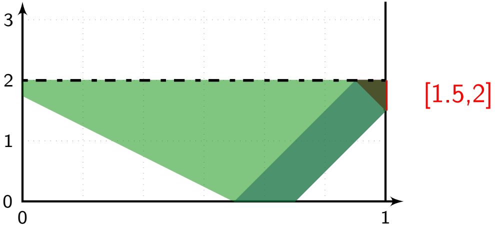




L+U-problem and fixpoint computation

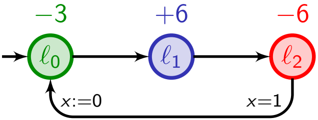
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

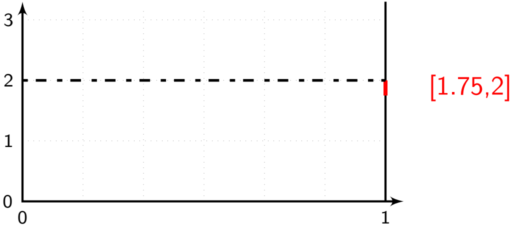




L+U-problem and fixpoint computation

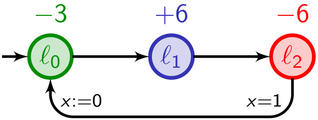
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

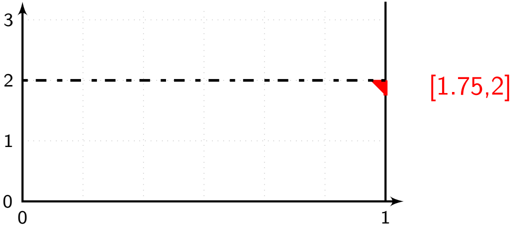




L+U-problem and fixpoint computation

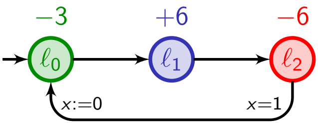
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

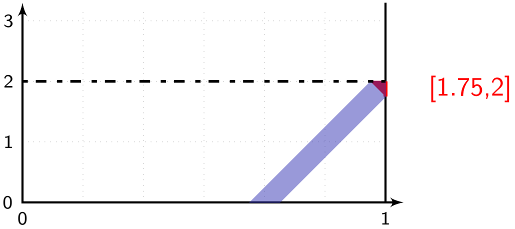




L+U-problem and fixpoint computation

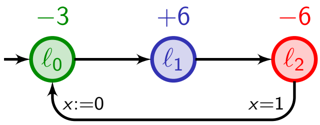
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

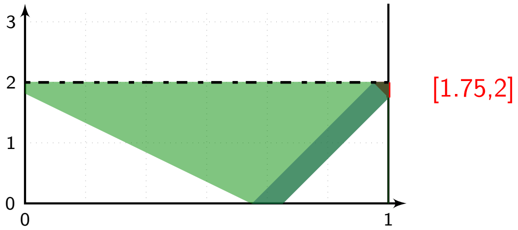




L+U-problem and fixpoint computation

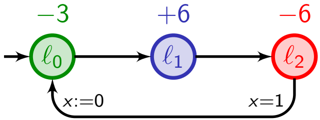
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

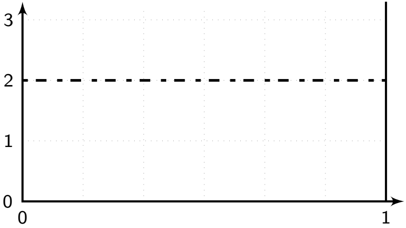




L+U-problem and fixpoint computation

The backward fixpoint computation, which is correct in the limit, does not terminate in general.

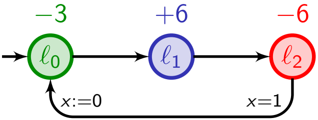


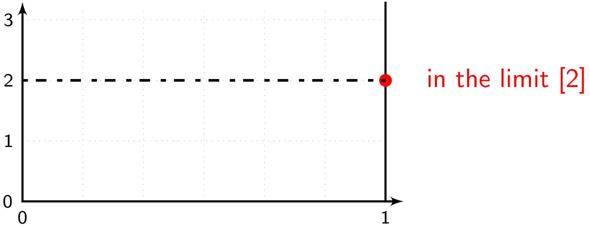


etc...

L+U-problem and fixpoint computation

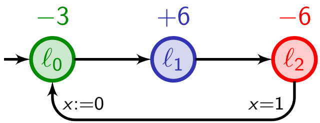
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

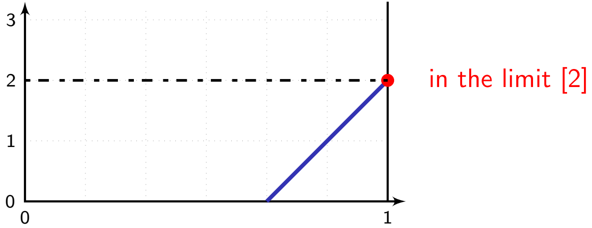




L+U-problem and fixpoint computation

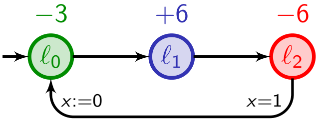
The backward fixpoint computation, which is correct in the limit, does not terminate in general.

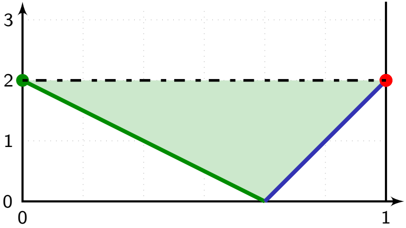




L+U-problem and fixpoint computation

The backward fixpoint computation, which is correct in the limit, does not terminate in general.





in the limit [2]

The results in the single-clock framework

	exist. problem	**univ. problem**	**games**
L	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+W	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+U	?	?	undecidable

The results in the single-clock framework

	exist. problem	**univ. problem**	**games**
L	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+W	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+U	?	?	undecidable

1 clock: **L+U**-games

Theorem

The single-clock **L**+**U**-games are undecidable.

1 clock: **L+U**-games

Theorem

The single-clock **L**+**U**-games are undecidable.

We encode the behaviour of a two-counter machine:

- each instruction is encoded as a module;
- the values c_1 and c_2 of the counters are encoded by the energy level

$$e = 5 - \frac{1}{2^{c_1} \cdot 3^{c_2}}$$

when entering the corresponding module.

1 clock: **L+U**-games

Theorem

The single-clock **L**+**U**-games are undecidable.

We encode the behaviour of a two-counter machine:

- each instruction is encoded as a module;
- the values c_1 and c_2 of the counters are encoded by the energy level

$$e = 5 - \frac{1}{2^{c_1} \cdot 3^{c_2}}$$

  when entering the corresponding module.

There is an infinite execution in the two-counter machine iff there is a strategy in the single-clock timed game under which the energy level remains between 0 and 5.

1 clock: **L+U**-games

Theorem

The single-clock **L+U**-games are undecidable.

We encode the behaviour of a two-counter machine:

- each instruction is encoded as a module;
- the values c_1 and c_2 of the counters are encoded by the energy level

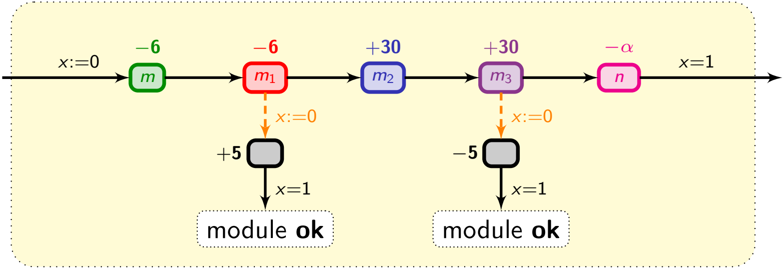
$$e = 5 - \frac{1}{2^{c_1} \cdot 3^{c_2}}$$

when entering the corresponding module.

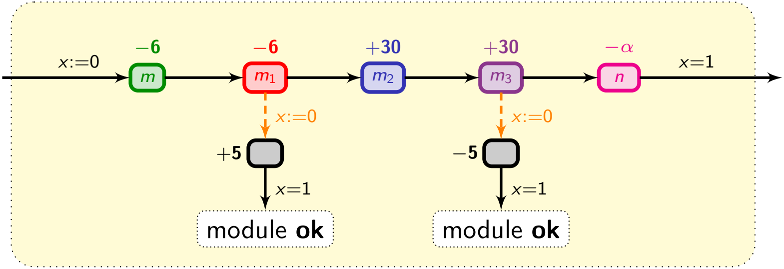
There is an infinite execution in the two-counter machine iff there is a strategy in the single-clock timed game under which the energy level remains between 0 and 5.

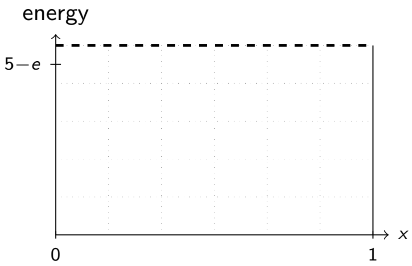
$\rightsquigarrow$ We present a generic construction for incrementing/decrementing the counters.

Generic module for incrementing/decrementing

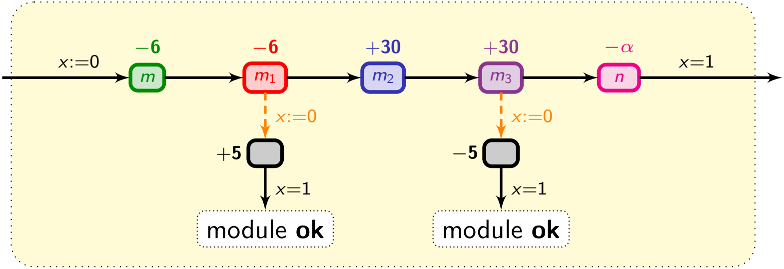


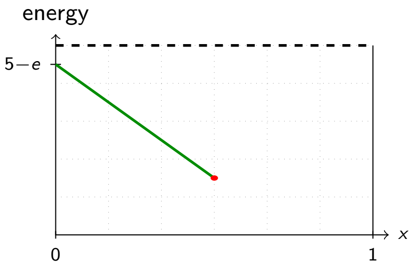
Generic module for incrementing/decrementing



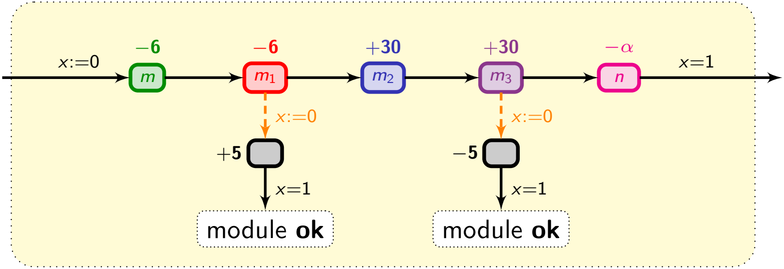


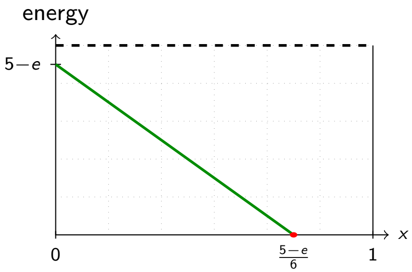
Generic module for incrementing/decrementing



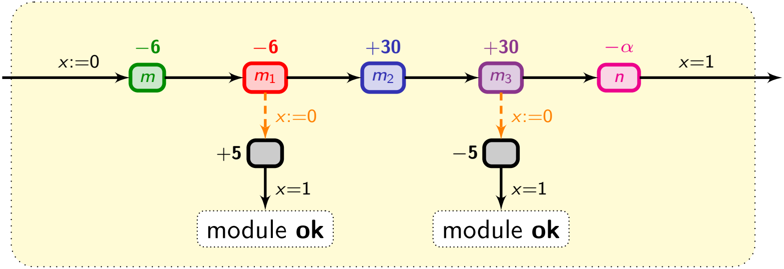


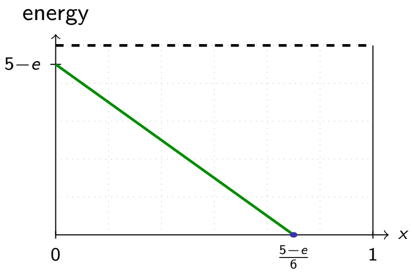
Generic module for incrementing/decrementing



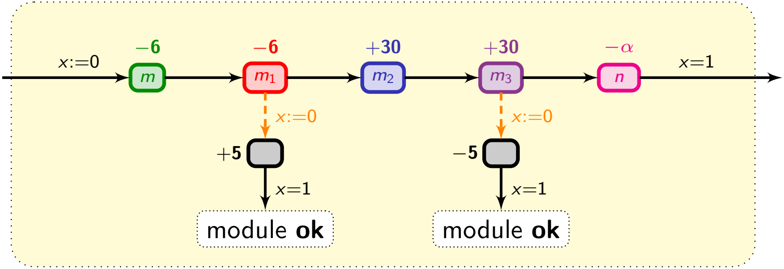


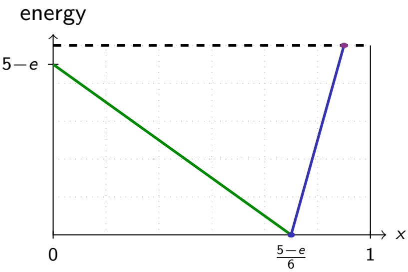
Generic module for incrementing/decrementing



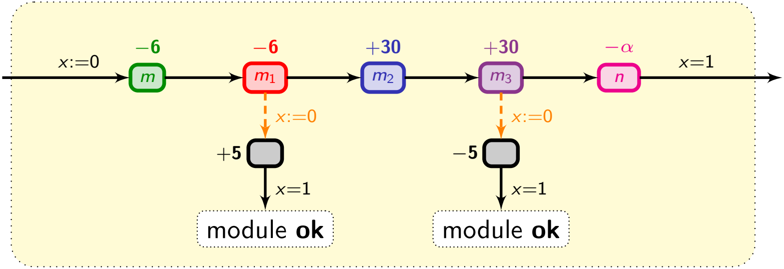


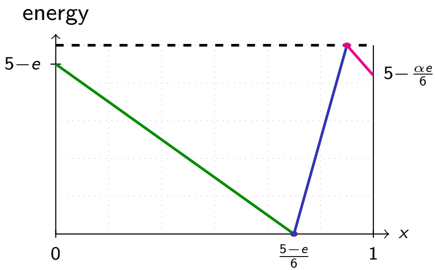
Generic module for incrementing/decrementing



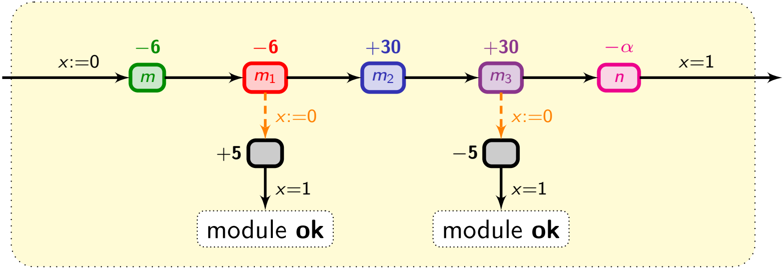


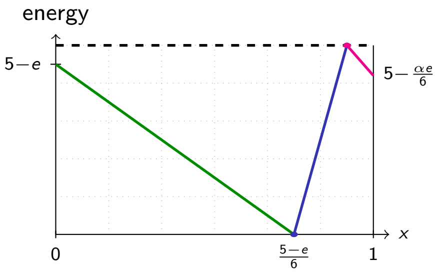
Generic module for incrementing/decrementing





Generic module for incrementing/decrementing





- α=3: increment c_1
- α=2: increment c_2
- α=12: decrement c_1
- α=18: decrement c_2

Outline

The results in the single-clock framework

	exist. problem	**univ. problem**	**games**
L	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+W	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+U	?	?	undecidable

The results in the single-clock framework

	exist. problem	**univ. problem**	**games**
L	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+W	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+U	?	?	undecidable

L- and **L+W**-cases: use the corner-point abstraction?

Idea: delay in the most profitable location

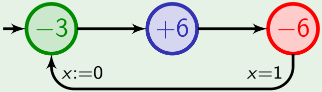
$\rightsquigarrow$ the corner-point abstraction

L- and **L+W**-cases: use the corner-point abstraction?

Idea: delay in the most profitable location

$\rightsquigarrow$ the corner-point abstraction

Example

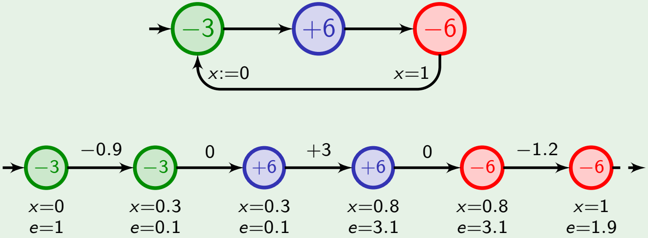


L- and **L+W**-cases: use the corner-point abstraction?

Idea: delay in the most profitable location

$\rightsquigarrow$ the corner-point abstraction

Example

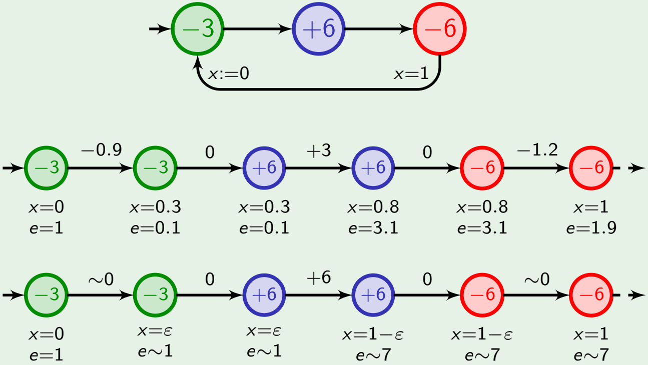


L- and **L+W**-cases: use the corner-point abstraction?

Idea: delay in the most profitable location

$\rightsquigarrow$ the corner-point abstraction

Example

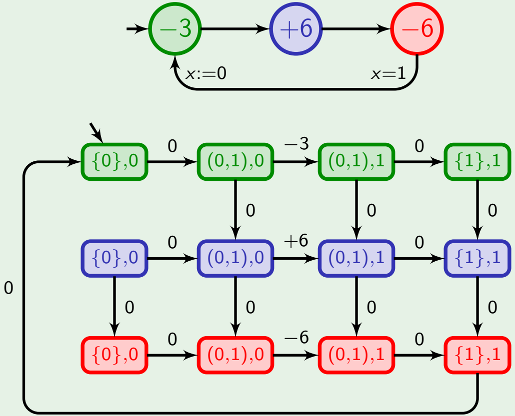


L- and **L+W**-cases: use the corner-point abstraction?

Idea: delay in the most profitable location

$\rightsquigarrow$ the corner-point abstraction

Example



L- and **L+W**-cases: use the corner-point abstraction?

Idea: delay in the most profitable location

$\rightsquigarrow$ the corner-point abstraction

Theorem [BFLMS08]

The corner-point abstraction is sound and complete for single-clock PTA with a linear observer and with no discrete costs. Hence the existential **L**- and **L+W**-problems are in P in that case.

[BFLMS08] Bouyer, Fahrenberg, Larsen, Markey, Srba. Infinite runs in weighted timed automata with energy constraints *(FORMATS'08)*.

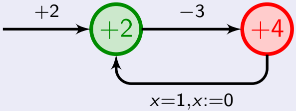
L- and **L+W**-cases: use the corner-point abstraction?

Idea: delay in the most profitable location

$\rightsquigarrow$ the corner-point abstraction

Remark

The corner-point abstraction is not correct with discrete costs.



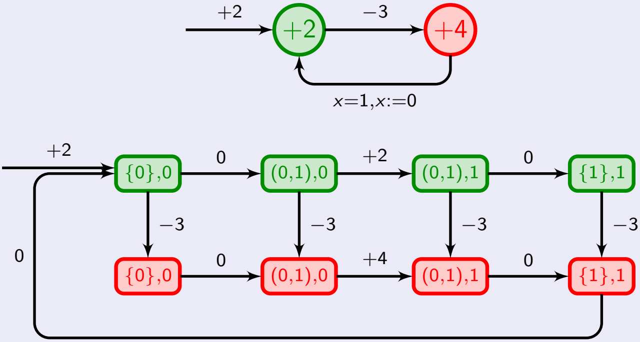
L- and **L+W**-cases: use the corner-point abstraction?

Idea: delay in the most profitable location

$\rightsquigarrow$ the corner-point abstraction

Remark

The corner-point abstraction is not correct with discrete costs.



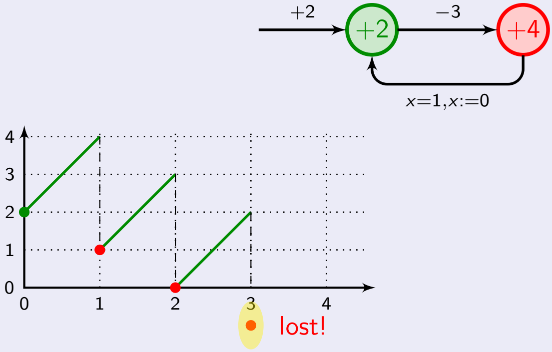
L- and **L+W**-cases: use the corner-point abstraction?

Idea: delay in the most profitable location

$\rightsquigarrow$ the corner-point abstraction

Remark

The corner-point abstraction is not correct with discrete costs.



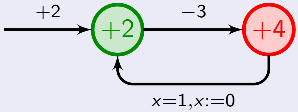
L- and **L+W**-cases: use the corner-point abstraction?

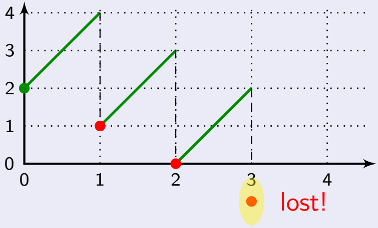
Idea: delay in the most profitable location

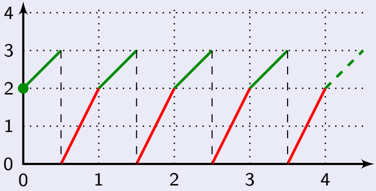
$\rightsquigarrow$ the corner-point abstraction

Remark

The corner-point abstraction is not correct with discrete costs.







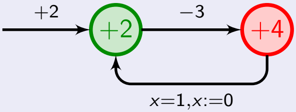
L- and **L+W**-cases: use the corner-point abstraction?

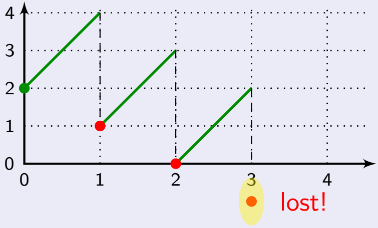
Idea: delay in the most profitable location

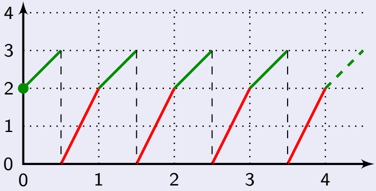
$\rightsquigarrow$ the corner-point abstraction

Remark

The corner-point abstraction is not correct with discrete costs.



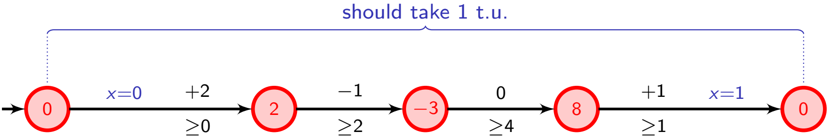




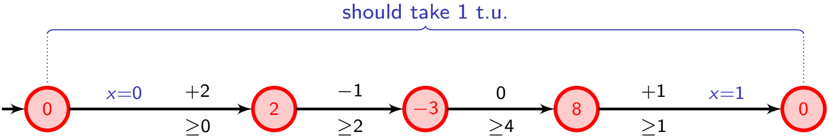
$\rightsquigarrow$ requires new developments!

Outline

Annotated unit path

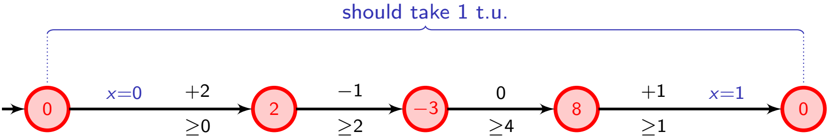


Annotated unit path



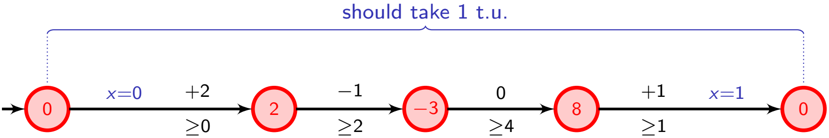
- starting with initial credit 0, it is not possible to reach the last location;

Annotated unit path



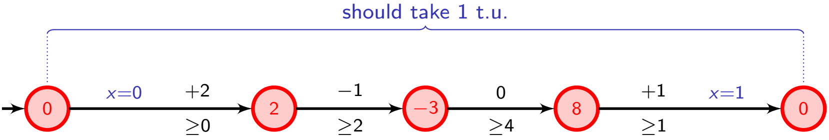
- starting with initial credit 0, it is not possible to reach the last location;
- starting with credit 1, we can exit with credit 5;

Annotated unit path



- starting with initial credit 0, it is not possible to reach the last location;
- starting with credit 1, we can exit with credit 5;
- starting with credit 3, it is possible to exit with final credit 13.

Annotated unit path

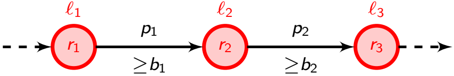


- starting with initial credit 0, it is not possible to reach the last location;
- starting with credit 1, we can exit with credit 5;
- starting with credit 3, it is possible to exit with final credit 13.

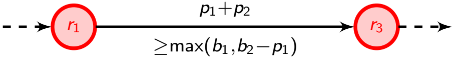
$\rightsquigarrow$ we will compute the **energy function**

"initial credit $\mapsto$ maximal final credit"

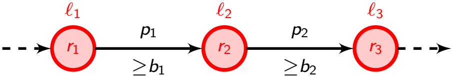
Simplifying annotated unit paths



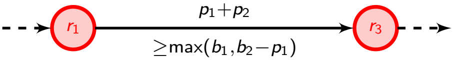
If (for some reason) no time should elapse in ℓ_2, then this path is equivalent (regarding the final cost) to



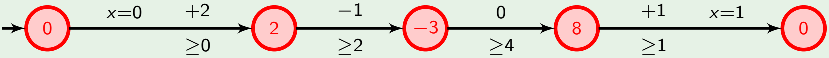
Simplifying annotated unit paths



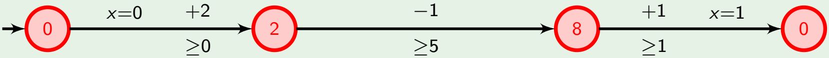
If (for some reason) no time should elapse in ℓ_2, then this path is equivalent (regarding the final cost) to



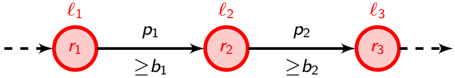
Example



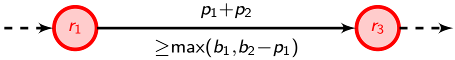
is equivalent to



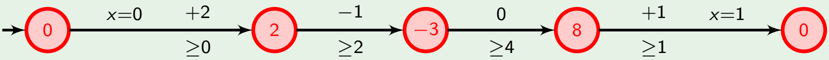
Simplifying annotated unit paths



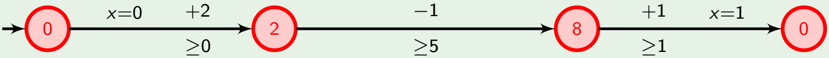
If (for some reason) no time should elapse in ℓ_2, then this path is equivalent (regarding the final cost) to



Example



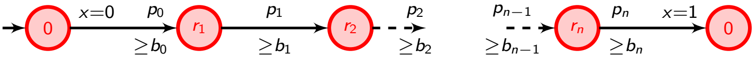
is equivalent to



$\rightsquigarrow$ we can select locations with increasing rates
(if some rate is positive...)

Normal form for unit paths

An annotated unit path

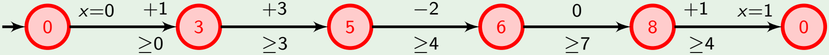


is in normal form if one of the following cases holds:

- $n = 1$ (trivial normal form);
- the rates r_i are positive and increasing, and $b_{i-1} + p_{i-1} < b_i$ for all $1 \leq i \leq n-1$ (positive normal form);
- the rates r_i are negative and decreasing, and $b_{i-1} + p_{i-1} > b_i$ for all $1 \leq i \leq n-1$ (negative normal form).

Normal form for unit paths

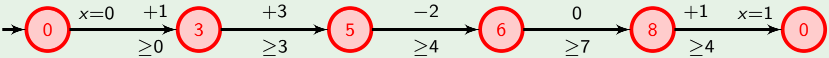
Example



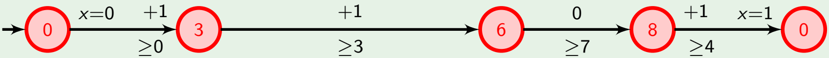
is not in normal form.

Normal form for unit paths

Example



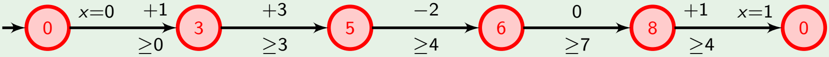
is not in normal form.



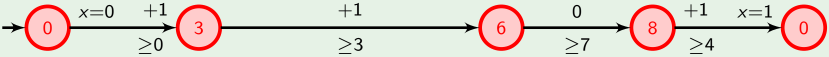
is in (positive) normal form.

Normal form for unit paths

Example



is not in normal form.



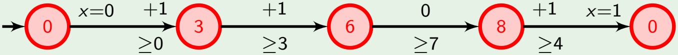
is in (positive) normal form.

Lemma

Any annotated unit path can be transformed into an equivalent (w.r.t. maximal final cost) normal form path.

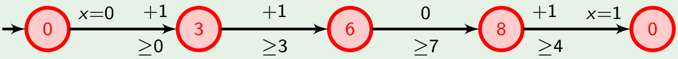
Computing optimal delays

Example



Computing optimal delays

Example

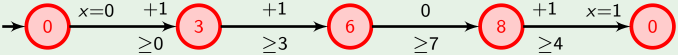


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};

Computing optimal delays

Example

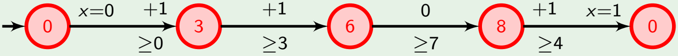


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—				—

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};

Computing optimal delays

Example

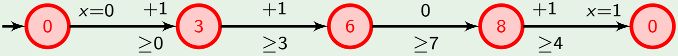


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—			0	—

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};

Computing optimal delays

Example

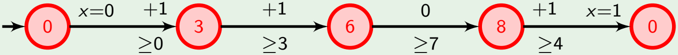


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—		$\frac{1}{2}$	0	—

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};

Computing optimal delays

Example

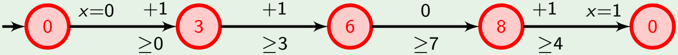


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};

Computing optimal delays

Example



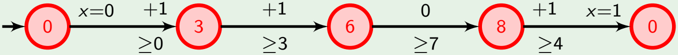
t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—

minimal initial credit required: $\frac{1}{2}$, yields final credit 8.

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};

Computing optimal delays

Example

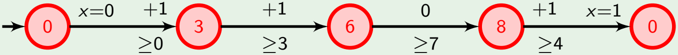


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};
- compute other points on the energy function curve.

Computing optimal delays

Example



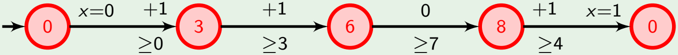
t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—

initial credit
$\frac{1}{2} + \delta$

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};
- compute other points on the energy function curve.

Computing optimal delays

Example



t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—

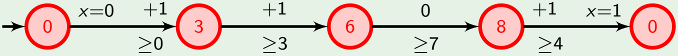
initial credit
$\frac{1}{2} + \delta$

$\frac{1}{2} - \frac{\delta}{3}$

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};
- compute other points on the energy function curve.

Computing optimal delays

Example

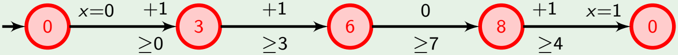


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—
initial credit $\frac{1}{2}+\delta$		$\frac{1}{2}-\frac{\delta}{3}$	$\frac{1}{2}$		

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};
- compute other points on the energy function curve.

Computing optimal delays

Example

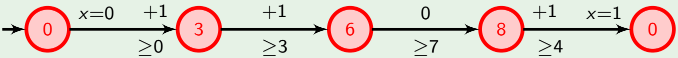


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—
initial credit $\frac{1}{2}+\delta$		$\frac{1}{2}-\frac{\delta}{3}$	$\frac{1}{2}$	$\frac{\delta}{3}$	

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};
- compute other points on the energy function curve.

Computing optimal delays

Example

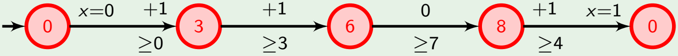


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—
initial credit $\frac{1}{2} + \delta$		$\frac{1}{2} - \frac{\delta}{3}$	$\frac{1}{2}$	$\frac{\delta}{3}$	final credit $8 + \frac{8}{3}\delta$

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};
- compute other points on the energy function curve.

Computing optimal delays

Example

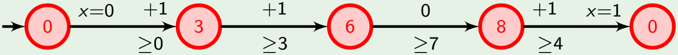


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—
	initial credit 2	0	$\frac{1}{2}$	$\frac{1}{2}$	final credit 12

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};
- compute other points on the energy function curve.

Computing optimal delays

Example

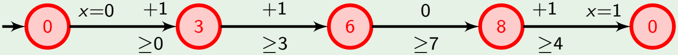


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—
initial credit $2+\delta$		0	$\frac{1}{2}-\frac{\delta}{6}$	$\frac{1}{2}+\frac{\delta}{6}$	final credit $12+\frac{8}{6}\delta$

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};
- compute other points on the energy function curve.

Computing optimal delays

Example

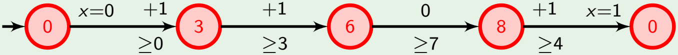


t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—
initial credit 5		0	0	1	final credit 16

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};
- compute other points on the energy function curve.

Computing optimal delays

Example



t_{opt}:	—	$\frac{2}{3}$	$\frac{1}{2}$	—	—
t^*:	—	$\frac{1}{2}$	$\frac{1}{2}$	0	—
initial credit $5+\delta$		0	0	1	final credit $16+\delta$

- compute optimal delays t_{opt} in ℓ_1 to ℓ_{n-1};
- compute optimal possible delays t^* in ℓ_1 to ℓ_{n-1};
- compute other points on the energy function curve.

Computing optimal delays

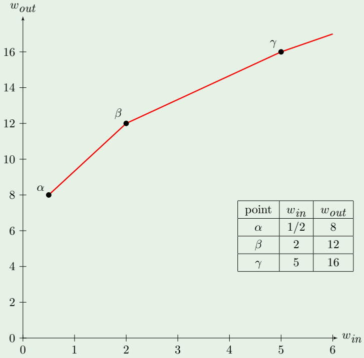
Example

Original automaton:

$$(0) \xrightarrow[\{c\}\quad \geq 0]{c=0\quad 1} (3) \xrightarrow[\geq 3]{3} (5) \xrightarrow[\geq 4]{-2} (6) \xrightarrow[\geq 7]{0} (8) \xrightarrow[\{c\}\quad \geq 4]{c=1\quad 1} (0)$$

Normal-form automaton:

$$(0) \xrightarrow[\{c\}\quad \geq 0]{c=0\quad 1} (3) \xrightarrow[\geq 3]{1} (6) \xrightarrow[\geq 7]{0} (8) \xrightarrow[\{c\}\quad \geq 4]{c=1\quad 1} (0)$$

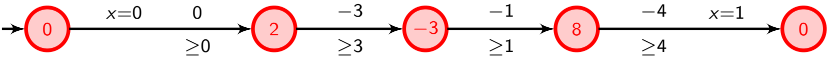


point	w_{in}	w_{out}
α	1/2	8
β	2	12
γ	5	16

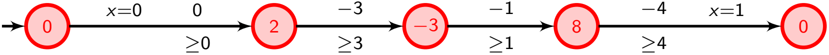
Outline

⏩ Time is almost over?

Restricted unit path

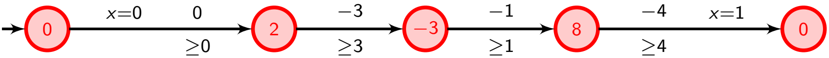


Restricted unit path



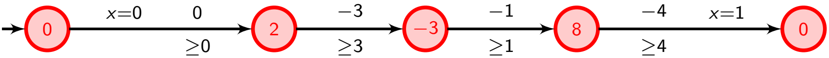
- starting with initial credit 0, it is not possible to reach the final location;

Restricted unit path



- starting with initial credit 0, it is not possible to reach the final location;
- starting with credit 1 and spending 1 t.u. in 2, we have credit $exp(2) \sim 7.39$ when exiting 2, which is not sufficient to reach the final location;

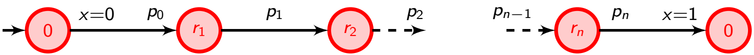
Restricted unit path



- starting with initial credit 0, it is not possible to reach the final location;
- starting with credit 1 and spending 1 t.u. in 2, we have credit $exp(2) \sim 7.39$ when exiting 2, which is not sufficient to reach the final location;
- starting with credit 1,
  - spending 0.8 t.u. in 2, we have credit $\exp(2 * 0.8) \sim 4.95$;
  - we reach 8 with credit around 0.95;
  - spending the remaining 0.2 t.u. there, we exit the path with credit approx. 0.72.

Normal form for exponential observers

A restricted unit path



is in normal form if one of the following cases holds:

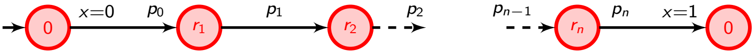
- $n = 1$ (trivial normal form);
- the rates r_i are positive and increasing, and

$$\frac{p_{i-1} \cdot r_{i-1} \cdot r_i}{r_{i-1} - r_i} < \frac{p_i \cdot r_i \cdot r_{i+1}}{r_i - r_{i+1}}$$

  for all $2 \leq i \leq n - 1$ (positive normal form);

Normal form for exponential observers

A restricted unit path



is in normal form if one of the following cases holds:

- $n = 1$ (trivial normal form);
- the rates r_i are positive and increasing, and

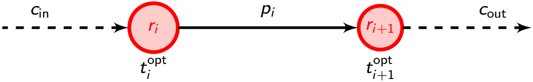
$$\frac{p_{i-1} \cdot r_{i-1} \cdot r_i}{r_{i-1} - r_i} < \frac{p_i \cdot r_i \cdot r_{i+1}}{r_i - r_{i+1}}$$

  for all $2 \leq i \leq n-1$ (positive normal form);

Lemma

Any restricted unit path can be transformed into an equivalent (w.r.t. maximal final credit) normal form path.

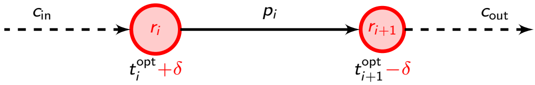
Normal form for exponential observers – Intuition



We have

$$c_{\text{out}} = \left(c_{\text{in}} \cdot e^{r_i t_i^{\text{opt}}} + p_i\right) \cdot e^{r_{i+1} t_{i+1}^{\text{opt}}}$$

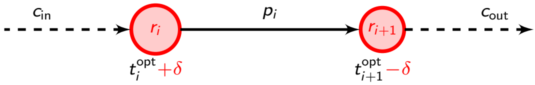
Normal form for exponential observers – Intuition



We have

$$c_{\text{out}} = \left(c_{\text{in}} \cdot e^{r_i(t_i^{\text{opt}}+\delta)} + p_i\right) \cdot e^{r_{i+1}(t_{i+1}^{\text{opt}}-\delta)}$$

Normal form for exponential observers – Intuition

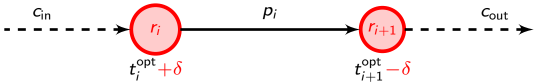


We have

$$c_{\text{out}} = \left(c_{\text{in}} \cdot e^{r_i(t_i^{\text{opt}} + \delta)} + p_i\right) \cdot e^{r_{i+1}(t_{i+1}^{\text{opt}} - \delta)}$$

$$\begin{aligned}\frac{\partial c_{\text{out}}}{\partial \delta} = r_i c_{\text{in}} \cdot e^{r_i(t_i^{\text{opt}} + \delta)} \cdot e^{r_{i+1}(t_{i+1}^{\text{opt}} - \delta)} \\ - \left(r_{i+1}\left(c_{\text{in}} \cdot e^{r_i(t_i^{\text{opt}} + \delta)} + p_i\right) \cdot e^{r_{i+1}(t_{i+1}^{\text{opt}} - \delta)}\end{aligned}$$

Normal form for exponential observers – Intuition

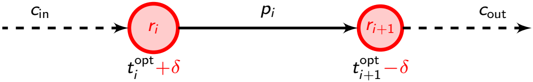


We have

$$c_{\text{out}} = \left(c_{\text{in}} \cdot e^{r_i (t_i^{\text{opt}} + \delta)} + p_i\right) \cdot e^{r_{i+1}(t_{i+1}^{\text{opt}} - \delta)}$$

$$\begin{aligned} \frac{\partial c_{\text{out}}}{\partial \delta} &= r_i c_{\text{in}} \cdot e^{r_i t_i^{\text{opt}}} \cdot e^{r_{i+1} t_{i+1}^{\text{opt}}} - \left(r_{i+1}\left(c_{\text{in}} \cdot e^{r_i t_i^{\text{opt}}} + p_i\right) \cdot e^{r_{i+1} t_{i+1}^{\text{opt}}}\right. \\ &= 0 \end{aligned}$$

Normal form for exponential observers – Intuition



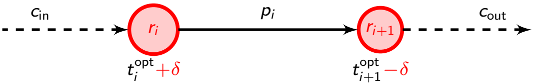
We have

$$c_{\text{out}} = \left(c_{\text{in}} \cdot e^{r_i(t_i^{\text{opt}}+\delta)} + p_i\right) \cdot e^{r_{i+1}(t_{i+1}^{\text{opt}}-\delta)}$$

Hence

$$c_{\text{in}} \cdot e^{r_i t_i^{\text{opt}}} = \frac{p_i \cdot r_{i+1}}{r_i - r_{i+1}}$$

Normal form for exponential observers – Intuition



We have

$$c_{\text{out}} = \left(c_{\text{in}} \cdot e^{r_i(t_i^{\text{opt}} + \delta)} + p_i\right) \cdot e^{r_{i+1}(t_{i+1}^{\text{opt}} - \delta)}$$

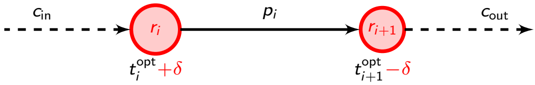
Hence

$$c_{\text{in}} \cdot e^{r_i t_i^{\text{opt}}} = \frac{p_i \cdot r_{i+1}}{r_i - r_{i+1}}$$

Lemma

The optimal credit with which to exit ℓ_i is $\frac{p_i \cdot r_{i+1}}{r_i - r_{i+1}}$.

Normal form for exponential observers – Intuition



We have

$$c_{\text{out}} = \left(c_{\text{in}} \cdot e^{r_i(t_i^{\text{opt}} + \delta)} + p_i\right) \cdot e^{r_{i+1}(t_{i+1}^{\text{opt}} - \delta)}$$

Hence

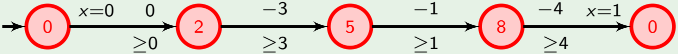
$$c_{\text{in}} \cdot e^{r_i t_i^{\text{opt}}} = \frac{p_i \cdot r_{i+1}}{r_i - r_{i+1}}$$

Lemma

Optimal runs spend no time in ℓ_i if $\frac{p_{i-1} \cdot r_i}{r_{i-1} - r_i} + p_{i-1} \geq \frac{p_i \cdot r_{i+1}}{r_i - r_{i+1}}$.

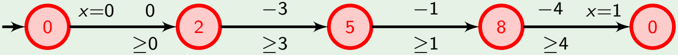
Computing optimal delays

Example



Computing optimal delays

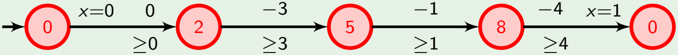
Example



c^{opt}	—	5	$\frac{8}{3}$	—	—

Computing optimal delays

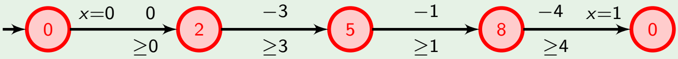
Example



c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—			—	—

Computing optimal delays

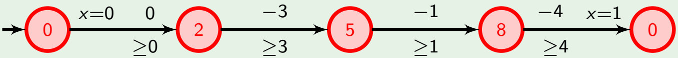
Example



c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—		$\frac{1}{5} \cdot \ln\left(\frac{8/3}{2}\right)$	—	—

Computing optimal delays

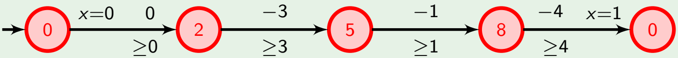
Example



c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2} \cdot \ln\left(\frac{5}{c_{\text{in}}}\right)$	$\frac{1}{5} \cdot \ln\left(\frac{8/3}{2}\right)$	—	—

Computing optimal delays

Example



c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2} \cdot \ln\left(\frac{5}{c_{\text{in}}}\right)$	$\frac{1}{5} \cdot \ln\left(\frac{8/3}{2}\right)$	—	—

Lemma

The optimal strategy is to delay t_i^{opt} as long as possible.

Computing optimal delays

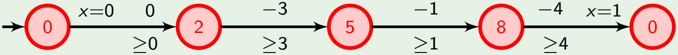
Example

$$\begin{tikzcd}[column sep=large] {} \arrow[r] & \boxed{0} \arrow[r, "x=0\quad 0", "\geq 0"'] & \boxed{2} \arrow[r, "-3", "\geq 3"'] & \boxed{5} \arrow[r, "-1", "\geq 1"'] & \boxed{8} \arrow[r, "-4\quad x=1", "\geq 4"'] & \boxed{0} \end{tikzcd}$$

c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2} \cdot \ln\left(\frac{5}{c_{\mathrm{in}}}\right)$	$\frac{1}{5} \cdot \ln\left(\frac{8/3}{2}\right)$	—	—
t^{min}:	—			$\frac{1}{8} \cdot \ln\left(\frac{4}{5/3}\right)$	—

Computing optimal delays

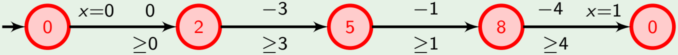
Example



c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2} \cdot \ln\left(\frac{5}{c_{\mathrm{in}}}\right)$	$\frac{1}{5} \cdot \ln\left(\frac{8/3}{2}\right)$	—	—
$t^{\min}$:	—	$\frac{1}{2} \cdot \ln\left(\frac{5}{c_{\mathrm{in}}}\right)$	$\frac{1}{5} \cdot \ln\left(\frac{8/3}{2}\right)$	$\frac{1}{8} \cdot \ln\left(\frac{4}{5/3}\right)$	—

Computing optimal delays

Example



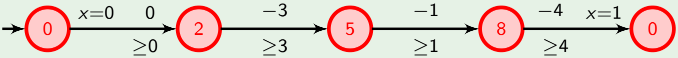
c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2}\cdot\ln(\frac{5}{c_{\text{in}}})$	$\frac{1}{5}\cdot\ln(\frac{8/3}{2})$	—	—
t^{min}:	—	$\frac{1}{2}\cdot\ln(\frac{5}{c_{\text{in}}})$	$\frac{1}{5}\cdot\ln(\frac{8/3}{2})$	$\frac{1}{8}\cdot\ln(\frac{4}{5/3})$	—

The minimal initial credit to reach the final location is

$$c^{\text{min}} = 5\cdot e^{-2}\cdot\left(\frac{12}{5}\right)^{1/4}\cdot\left(\frac{4}{3}\right)^{2/5}.$$

Computing optimal delays

Example



c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2} \cdot \ln\left(\frac{5}{c_{\text{in}}}\right)$	$\frac{1}{5} \cdot \ln\left(\frac{8/3}{2}\right)$	—	—
t^{min}:	—	$\frac{1}{2} \cdot \ln\left(\frac{5}{c_{\text{in}}}\right)$	$\frac{1}{5} \cdot \ln\left(\frac{8/3}{2}\right)$	$\frac{1}{8} \cdot \ln\left(\frac{4}{5/3}\right)$	—

Starting with credit $k \cdot c_{\text{min}}$ (between c_{min} and 5):

Computing optimal delays

Example

$$\rightarrow 0 \xrightarrow[\geq 0]{x=0 \quad 0} 2 \xrightarrow[\geq 3]{-3} 5 \xrightarrow[\geq 1]{-1} 8 \xrightarrow[\geq 4]{-4 \quad x=1} 0$$

	0	2	5	8	0
c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2} \cdot \ln(\frac{5}{c_{\text{in}}})$	$\frac{1}{5} \cdot \ln(\frac{8/3}{2})$	—	—
t^{min}:	—	$\frac{1}{2} \cdot \ln(\frac{5}{c_{\text{in}}})$	$\frac{1}{5} \cdot \ln(\frac{8/3}{2})$	$\frac{1}{8} \cdot \ln(\frac{4}{5/3})$	—

Starting with credit $k \cdot c_{\text{min}}$ (between c_{min} and 5):

- we spend $\frac{1}{2} \ln(\frac{5}{k \cdot c_{\text{min}}})$ in location 2;

Computing optimal delays

Example

$$\begin{tikzcd}[column sep=large]
{} \arrow[r] & 0 \arrow[r, "x=0\quad 0", "\geq 0"'] & 2 \arrow[r, "-3", "\geq 3"'] & 5 \arrow[r, "-1", "\geq 1"'] & 8 \arrow[r, "-4\quad x=1", "\geq 4"'] & 0
\end{tikzcd}$$

c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2} \cdot \ln(\frac{5}{c_{\text{in}}})$	$\frac{1}{5} \cdot \ln(\frac{8/3}{2})$	—	—
t^{min}:	—	$\frac{1}{2} \cdot \ln(\frac{5}{c_{\text{in}}})$	$\frac{1}{5} \cdot \ln(\frac{8/3}{2})$	$\frac{1}{8} \cdot \ln(\frac{4}{5/3})$	—

Starting with credit $k \cdot c_{\min}$ (between $c_{\min}$ and 5):

- we spend $\frac{1}{2} \ln(\frac{5}{k \cdot c_{\min}}) = t_{\min} - \frac{1}{2} \cdot \ln(k)$ in location 2;

Computing optimal delays

Example

$$\rightarrow 0 \xrightarrow[\geq 0]{x=0 \quad 0} 2 \xrightarrow[\geq 3]{-3} 5 \xrightarrow[\geq 1]{-1} 8 \xrightarrow[\geq 4]{-4 \quad x=1} 0$$

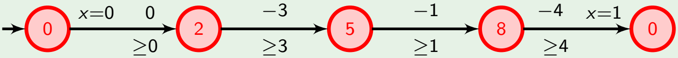
c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2} \cdot \ln(\frac{5}{c_{\text{in}}})$	$\frac{1}{5} \cdot \ln(\frac{8/3}{2})$	—	—
t^{min}:	—	$\frac{1}{2} \cdot \ln(\frac{5}{c_{\text{in}}})$	$\frac{1}{5} \cdot \ln(\frac{8/3}{2})$	$\frac{1}{8} \cdot \ln(\frac{4}{5/3})$	—

Starting with credit $k \cdot c_{\min}$ (between $c_{\min}$ and 5):

- we spend $\frac{1}{2}\ln(\frac{5}{k \cdot c_{\min}}) = t_{\min} - \frac{1}{2} \cdot \ln(k)$ in location 2;
- we transfer $\frac{1}{2} \cdot \ln(k)$ t.u. to location 8;

Computing optimal delays

Example



c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2} \cdot \ln(\frac{5}{c_{\mathrm{in}}})$	$\frac{1}{5} \cdot \ln(\frac{8/3}{2})$	—	—
$t^{\min}$:	—	$\frac{1}{2} \cdot \ln(\frac{5}{c_{\mathrm{in}}})$	$\frac{1}{5} \cdot \ln(\frac{8/3}{2})$	$\frac{1}{8} \cdot \ln(\frac{4}{5/3})$	—

Starting with credit $k \cdot c_{\min}$ (between $c_{\min}$ and 5):

- we spend $\frac{1}{2} \ln(\frac{5}{k \cdot c_{\min}}) = t_{\min} - \frac{1}{2} \cdot \ln(k)$ in location 2;
- we transfer $\frac{1}{2} \cdot \ln(k)$ t.u. to location 8;
- the final credit is $4 \cdot e^{8 \cdot \frac{1}{2} \cdot \ln(k)} - 4$.

Computing optimal delays

Example

$$\to \textcircled{0} \xrightarrow[\geq 0]{x=0 \quad 0} \textcircled{2} \xrightarrow[\geq 3]{-3} \textcircled{5} \xrightarrow[\geq 1]{-1} \textcircled{8} \xrightarrow[\geq 4]{-4 \quad x=1} \textcircled{0}$$

	0	2	5	8	0
c^{opt}	—	5	$\frac{8}{3}$	—	—
t^{opt}:	—	$\frac{1}{2} \cdot \ln\left(\frac{5}{c_{\mathrm{in}}}\right)$	$\frac{1}{5} \cdot \ln\left(\frac{8/3}{2}\right)$	—	—
t^{min}:	—	$\frac{1}{2} \cdot \ln\left(\frac{5}{c_{\mathrm{in}}}\right)$	$\frac{1}{5} \cdot \ln\left(\frac{8/3}{2}\right)$	$\frac{1}{8} \cdot \ln\left(\frac{4}{5/3}\right)$	—

Starting with credit $k \cdot c_{\mathrm{min}}$ (between c_{min} and 5):

- we spend $\frac{1}{2}\ln\left(\frac{5}{k \cdot c_{\mathrm{min}}}\right) = t_{\mathrm{min}} - \frac{1}{2} \cdot \ln(k)$ in location 2;
- we transfer $\frac{1}{2} \cdot \ln(k)$ t.u. to location 8;
- the final credit is $4 \cdot e^{8 \cdot \frac{1}{2} \cdot \ln(k)} - 4 = 4 \cdot \left(\frac{c_{\mathrm{in}}}{c_{\mathrm{min}}}\right)^4 - 4$.

Computing optimal delays

Theorem

Given a restricted unit path and an initial credit, we can compute in polynomial time the optimal final credit (in closed form).

Computing optimal delays

Theorem

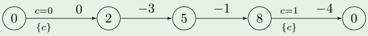
Given a restricted unit path and an initial credit, we can compute in polynomial time the optimal final credit (in closed form).

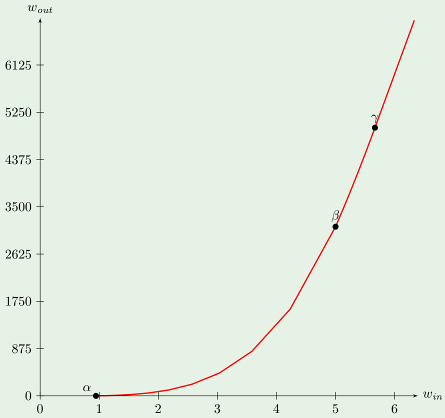
Moreover the energy function:

- is piecewise of the form $\alpha \cdot (c_{\text{in}} - \beta)^{r_i/r_j} + \gamma$, with $r_i \geq r_j$;
- has continuous derivative.

Computing optimal delays

Example





point	w_{in}	w_{out}
α	$e^{-2}*5*(12/5)^{1/4}*(4/3)^{2/5}$ ≈ 0.944951	0
β	5	$e^{8}*5/3*(3/4)^{8/5}-4$ ≈ 3131.47
γ	$17/3$	$e^{8}*5/3-4$ ≈ 4964.26

interval	equation of the curve
$\alpha - \beta$	$w_{\text{out}} = \frac{5}{3} \cdot \left(\frac{w_{\text{in}}}{e^{-2} * 5 * (4/3)^{2/5}}\right)^4 - 4$
$\beta - \gamma$	$w_{\text{out}} = \frac{5}{3} \cdot \left(\frac{w_{\text{in}} - 3}{e^{-5} * 8/3}\right)^{8/5} - 4$
$\gamma - +\infty$	$w_{\text{out}} = (w_{\text{in}} - 4) \cdot e^{8} - 4$

Outline

Basic simplifications

Remark

For the sake of simplicity, we restrict to closed timed automata.

Basic simplifications

Lemma

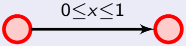
We can assume that there is a global invariant $x \leq 1$, and that there are only three kind of transitions:

$\bigcirc \xrightarrow{0 \leq x \leq 1} \bigcirc$ or $\bigcirc \xrightarrow{x=0} \bigcirc$ or $\bigcirc \xrightarrow[x:=0]{x=1} \bigcirc$

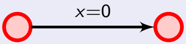
Basic simplifications

Lemma

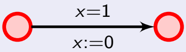
We can assume that there is a global invariant $x \leq 1$, and that there are only three kind of transitions:



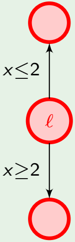
or



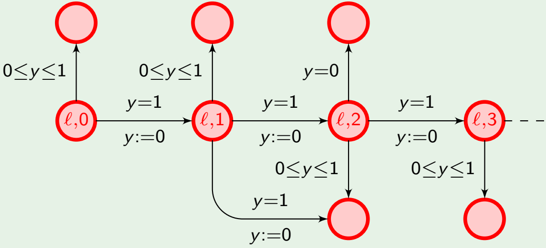
or



Example



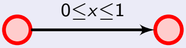
$\rightsquigarrow$



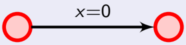
Basic simplifications

Lemma

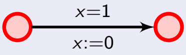
We can assume that there is a global invariant $x \leq 1$, and that there are only three kind of transitions:



or



or



Example

Handling non-resetting cycles

Lemma

For each location ℓ, we can compute a value $w_{\mathsf{Zeno}}(\ell)$ such that there is an infinite non-resetting feasible run from ℓ with initial credit w iff $w \geq w_{\mathsf{Zeno}}(\ell)$.

Handling non-resetting cycles

Lemma

For each location ℓ, we can compute a value $w_{\mathsf{Zeno}}(\ell)$ such that there is an infinite non-resetting feasible run from ℓ with initial credit w iff $w \geq w_{\mathsf{Zeno}}(\ell)$.

Lemma

From an automaton $\mathcal{A}$, we can compute an equivalent automaton $\mathcal{A}'$ labelled with w_{Zeno} and not containing any non-resetting cycle.

Main result for the **L+W**-problem

Theorem [BFLM10]

Optimization, reachability and existence of infinite runs satisfying the constraint ≥ 0 can be decided in EXPTIME in single-clock PTA

- either with a linear observer;
- or with an exponential observer with non-positive discrete costs.

[BFLM10] Bouyer, Fahrenberg, Larsen, Markey. Timed automata with observers under energy constraints *(HSCC'10)*.

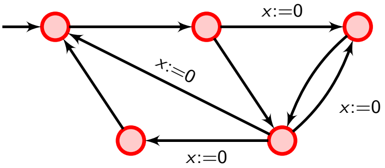
Main result for the **L+W**-problem

Theorem [BFLM10]

Optimization, reachability and existence of infinite runs satisfying the constraint ≥ 0 can be decided in EXPTIME in single-clock PTA

- either with a linear observer;
- or with an exponential observer with non-positive discrete costs.

- transform the automaton into an automaton with energy functions;



[BFLM10] Bouyer, Fahrenberg, Larsen, Markey. Timed automata with observers under energy constraints *(HSCC'10)*.

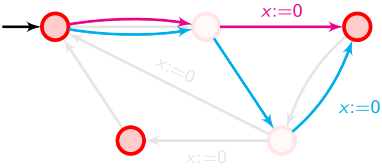
Main result for the **L+W**-problem

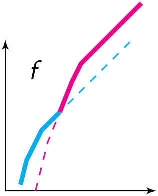
Theorem [BFLM10]

Optimization, reachability and existence of infinite runs satisfying the constraint ≥ 0 can be decided in EXPTIME in single-clock PTA

- either with a linear observer;
- or with an exponential observer with non-positive discrete costs.

- transform the automaton into an automaton with energy functions;





[BFLM10] Bouyer, Fahrenberg, Larsen, Markey. Timed automata with observers under energy constraints *(HSCC'10)*.

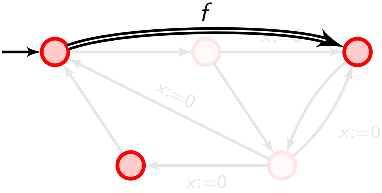
Main result for the L+W-problem

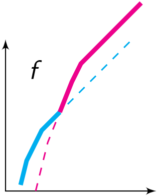
Theorem [BFLM10]

Optimization, reachability and existence of infinite runs satisfying the constraint ≥ 0 can be decided in EXPTIME in single-clock PTA

- either with a linear observer;
- or with an exponential observer with non-positive discrete costs.

- transform the automaton into an automaton with energy functions;





[BFLM10] Bouyer, Fahrenberg, Larsen, Markey. Timed automata with observers under energy constraints *(HSCC'10)*.

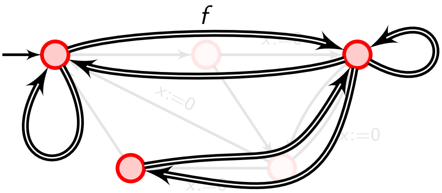
Main result for the **L+W**-problem

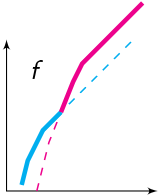
Theorem [BFLM10]

Optimization, reachability and existence of infinite runs satisfying the constraint ≥ 0 can be decided in EXPTIME in single-clock PTA

- either with a linear observer;
- or with an exponential observer with non-positive discrete costs.

- transform the automaton into an automaton with energy functions;





[BFLM10] Bouyer, Fahrenberg, Larsen, Markey. Timed automata with observers under energy constraints *(HSCC'10)*.

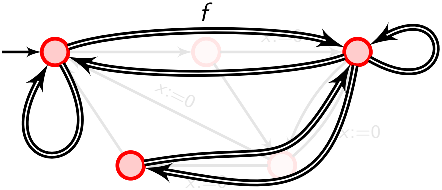
Main result for the **L+W**-problem

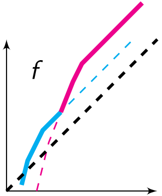
Theorem [BFLM10]

Optimization, reachability and existence of infinite runs satisfying the constraint ≥ 0 can be decided in EXPTIME in single-clock PTA

- either with a linear observer;
- or with an exponential observer with non-positive discrete costs.

- transform the automaton into an automaton with energy functions;





- check if simple cycles can be iterated, or if a Zeno cycle can be reached (use of w_{Zeno}).

[BFLM10] Bouyer, Fahrenberg, Larsen, Markey. Timed automata with observers under energy constraints *(HSCC'10)*.

Outline

Conclusion & future work

- The energy management problem:
  - nice problem statement
  - has inspired some developments in mean-payoff games
  - non-trivial to solve in general!

Conclusion & future work

- The energy management problem:
  - nice problem statement
  - has inspired some developments in mean-payoff games
  - non-trivial to solve in general!
- The single-clock framework:
  - computation of (infinite) schedules satisfying some simple energy constraints (energy should remain ≥ 0 – the **L**-problem)
  - surprisingly decidable for exponential observers?
  - required an optimization algorithm along unit paths

Conclusion & future work

- Many open problems:
  - can we go beyond linear and exponential observers? (In particular can we handle observers that mix linear and exponential evolutions?)
  - what if there are upper bounds on the observer variable? (**L+U**-problem)
  - what if there are more than one clock?
  - what if there is an interacting environment (games)?
  - ...

	exist. problem	univ. problem	games
L	$\in$ P	$\in$ P	$\in$ UP $\cap$ coUP P-hard
L+W	$\in$ P	$\in$ P	$\in$ NP $\cap$ coNP P-hard
L+U	$\in$ PSPACE NP-hard	$\in$ P	EXPTIME-c.

	exist. problem	univ. problem	games
L	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+W	$\in$ P $\in$ EXPTIME	$\in$ P $\in$ EXPTIME	?
L+U	?	?	undecidable

	exist. problem	univ. problem	games
L	?	?	?
L+W	?	?	?
L+U	?	?	undecidable