Quantitative Models for Verification

— A timed-automata-based perspective —

Patricia Bouyer-Decitre

LSV, CNRS & ENS Cachan, France

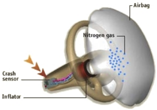
September 22, 2011

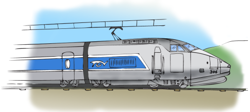
Time-dependent systems

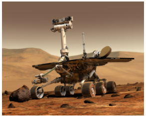
- We are interested in timed systems

Time-dependent systems

- We are interested in timed systems

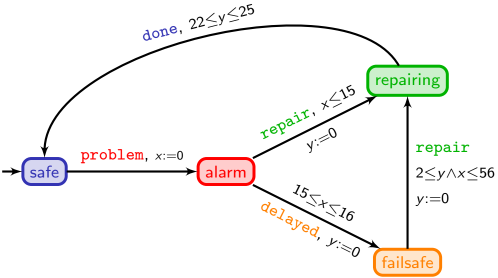






The standard timed automaton model [AD90,AD94]

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	⋯
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)*.

Why should we go further?

- Not only time should be modelled, but further (time-dependent, or not) information might be of interest.
- Examples:
  - Interaction
  - Uncertainty
  - Resources
  - ...

Task graph scheduling problems

Compute $D\times(C\times(A+B))+(A+B)+(C\times D)$ using two processors:

Task graph scheduling problems

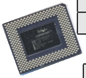
Compute $D \times (C \times (A + B)) + (A + B) + (C \times D)$ using two processors:

P_1 (fast):

time	
$+$	2 picoseconds
$\times$	3 picoseconds

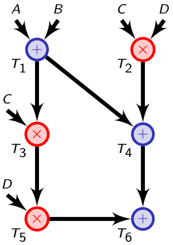
energy	
idle	10 Watt
in use	90 Watts

P_2 (slow):



time	
$+$	5 picoseconds
$\times$	7 picoseconds

energy	
idle	20 Watts
in use	30 Watts



Task graph scheduling problems

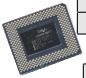
Compute $D \times (C \times (A + B)) + (A + B) + (C \times D)$ using two processors:

P_1 (fast):

time	
$+$	2 picoseconds
$\times$	3 picoseconds

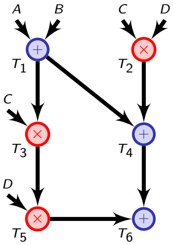
energy	
idle	10 Watt
in use	90 Watts

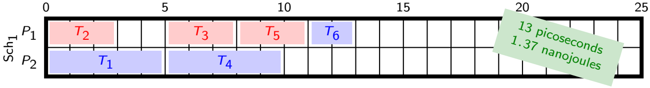
P_2 (slow):



time	
$+$	5 picoseconds
$\times$	7 picoseconds

energy	
idle	20 Watts
in use	30 Watts





Task graph scheduling problems

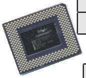
Compute $D \times (C \times (A + B)) + (A + B) + (C \times D)$ using two processors:

P_1 (fast):

time	
$+$	2 picoseconds
$\times$	3 picoseconds

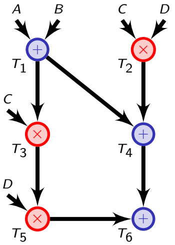
energy	
idle	10 Watt
in use	90 Watts

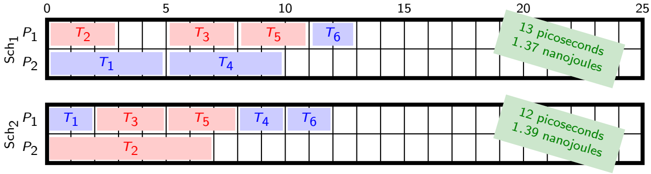
P_2 (slow):



time	
$+$	5 picoseconds
$\times$	7 picoseconds

energy	
idle	20 Watts
in use	30 Watts





Task graph scheduling problems

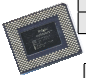
Compute $D\times(C\times(A+B))+(A+B)+(C\times D)$ using two processors:

P_1 (fast):

time	
$+$	2 picoseconds
$\times$	3 picoseconds

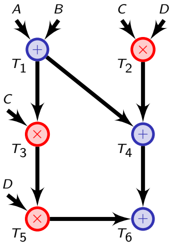
energy	
idle	10 Watt
in use	90 Watts

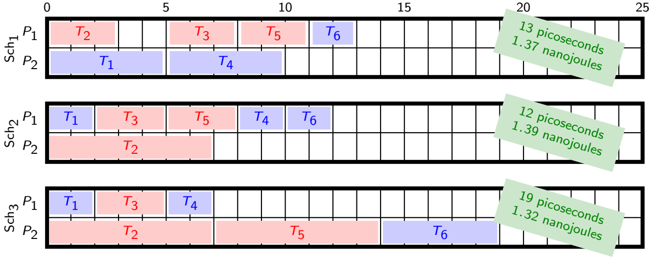
P_2 (slow):



time	
$+$	5 picoseconds
$\times$	7 picoseconds

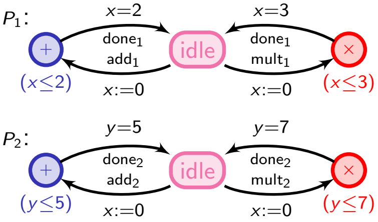
energy	
idle	20 Watts
in use	30 Watts





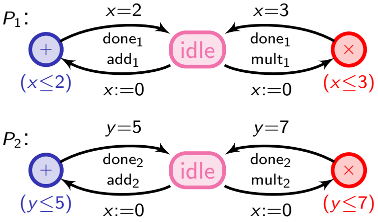
Modelling the task graph scheduling problem

- Processors

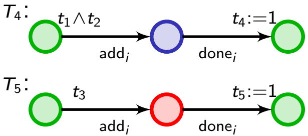


Modelling the task graph scheduling problem

- Processors

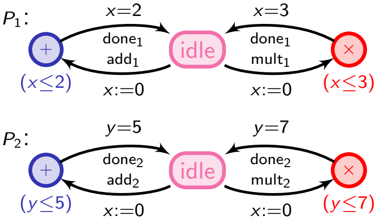


- Tasks

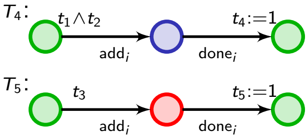


Modelling the task graph scheduling problem

- Processors



- Tasks



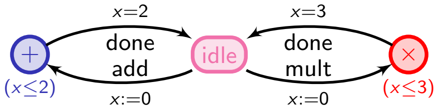
Global system: $(P_1 \parallel P_2) \parallel_s (T_1 \parallel T_2 \parallel \cdots \parallel T_6)$

A schedule: a path in the global system which reaches $t_1 \wedge \cdots \wedge t_6$

Why (timed) games?

- to model uncertainty

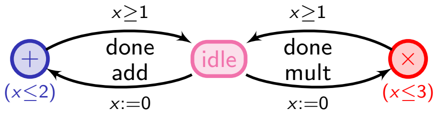
Example of a processor in the taskgraph example



Why (timed) games?

- to model uncertainty

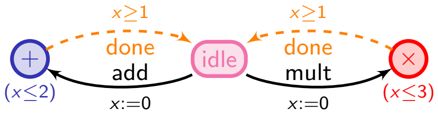
Example of a processor in the taskgraph example



Why (timed) games?

- to model uncertainty

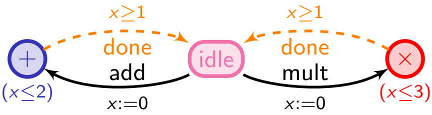
Example of a processor in the taskgraph example



Why (timed) games?

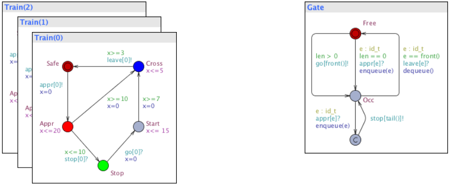
- to model uncertainty

Example of a processor in the taskgraph example



- to model an interaction with an environment

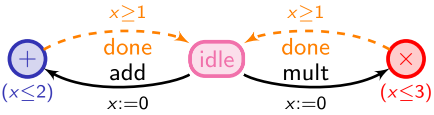
Example of the gate in the train/gate example



Why (timed) games?

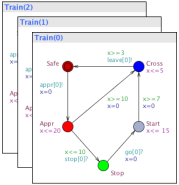
- to model uncertainty

Example of a processor in the taskgraph example



- to model an interaction with an environment

Example of the gate in the train/gate example

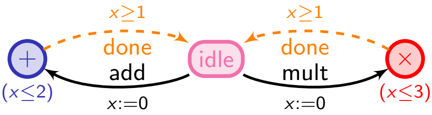


?

Why (timed) games?

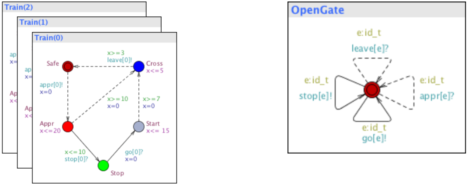
- to model uncertainty

Example of a processor in the taskgraph example

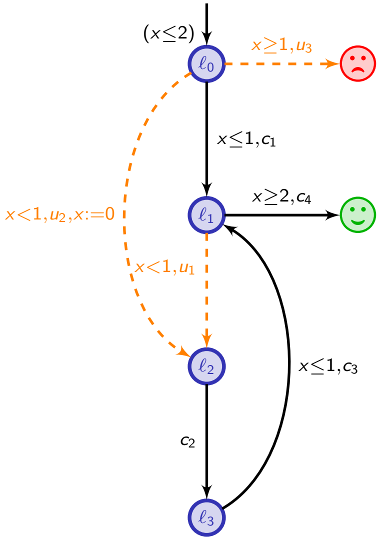


- to model an interaction with an environment

Example of the gate in the train/gate example



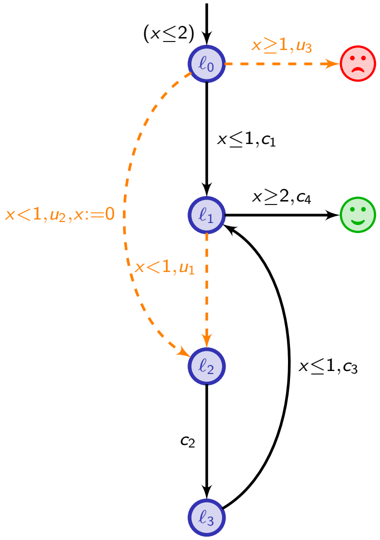
An example of a timed game



Rule of the game

- Aim: avoid ☹ and reach ☺

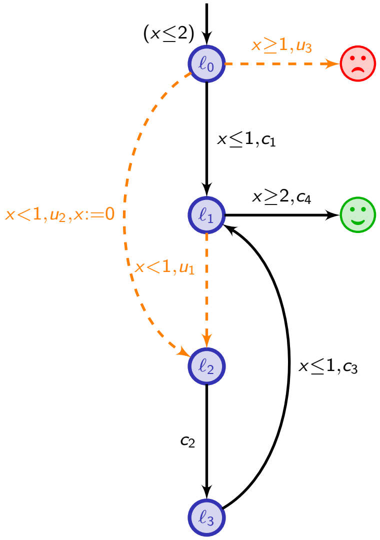
An example of a timed game



Rule of the game

- Aim: avoid ☹ and reach ☺
- How do we play? According to a strategy:

An example of a timed game

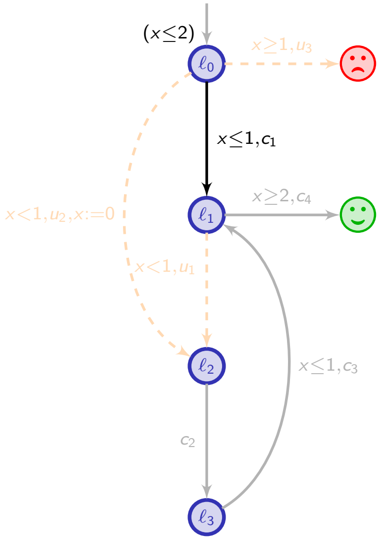


Rule of the game

- Aim: avoid ☹ and reach ☺
- How do we play? According to a strategy:

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

An example of a timed game



Rule of the game

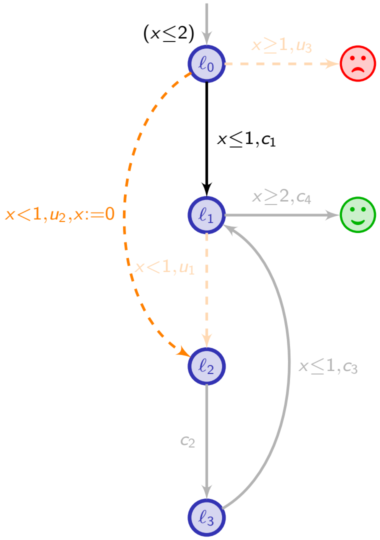
- Aim: avoid ☹ and reach ☺
- How do we play? According to a strategy:

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$

An example of a timed game



Rule of the game

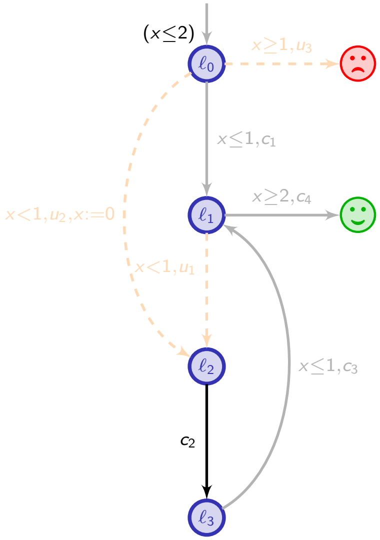
- Aim: avoid ☹ and reach ☺
- How do we play? According to a strategy:

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$
  $\leadsto$ can be preempted by u_2

An example of a timed game



Rule of the game

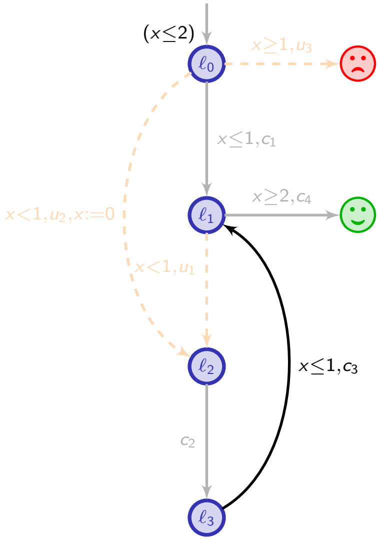
- Aim: avoid ☹ and reach ☺
- How do we play? According to a strategy:

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$
  $\rightsquigarrow$ can be preempted by u_2
- from $(\ell_2, \star)$, play $(1 - \star, c_2)$

An example of a timed game



Rule of the game

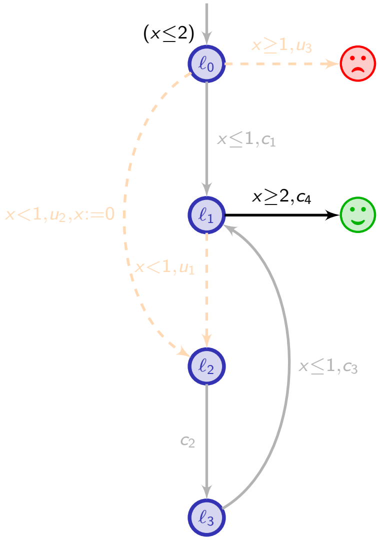
- Aim: avoid ☹ and reach ☺
- How do we play? According to a strategy:

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$
  $\rightsquigarrow$ can be preempted by u_2
- from $(\ell_2, \star)$, play $(1 - \star, c_2)$
- from $(\ell_3, 1)$, play $(0, c_3)$

An example of a timed game



Rule of the game

- Aim: avoid ☹ and reach ☺
- How do we play? According to a strategy:

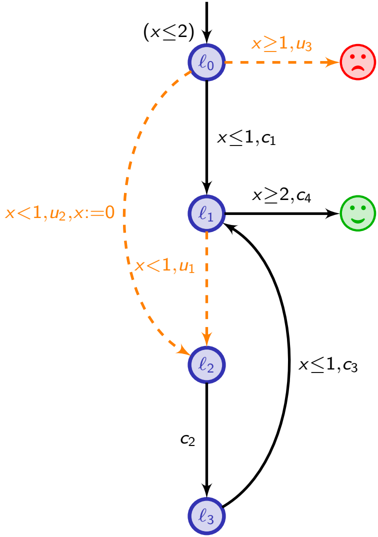
$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$

  $\rightsquigarrow$ can be preempted by u_2
- from $(\ell_2, \star)$, play $(1 - \star, c_2)$
- from $(\ell_3, 1)$, play $(0, c_3)$
- from $(\ell_1, 1)$, play $(1, c_4)$

An example of a timed game



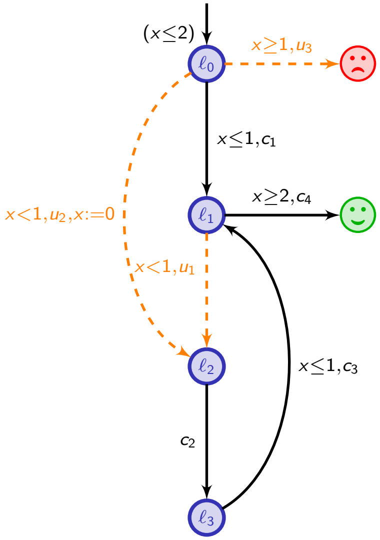
Rule of the game

- Aim: avoid ☹ and reach ☺
- How do we play? According to a strategy:

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

Problems to be considered

An example of a timed game



Rule of the game

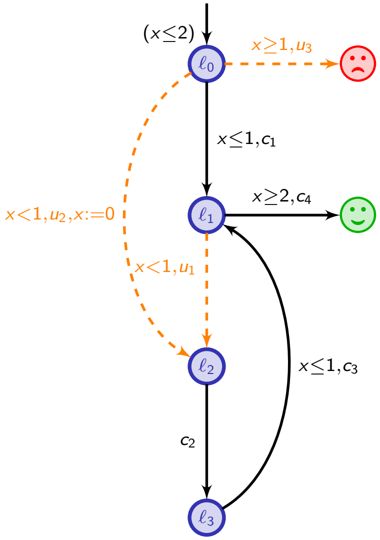
- Aim: avoid ☹ and reach ☺
- How do we play? According to a strategy:

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

Problems to be considered

- Does there exist a winning strategy?

An example of a timed game



Rule of the game

- Aim: avoid ☹ and reach ☺
- How do we play? According to a strategy:

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

Problems to be considered

- Does there exist a winning strategy?
- If yes, compute one (as simple as possible).

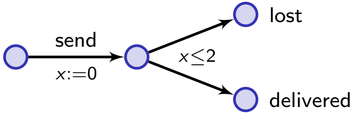
Why add stochastic features? And how?

- to model probabilistic behaviours

Why add stochastic features? And how?

- to model probabilistic behaviours

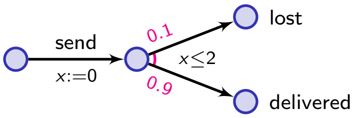
Example of losses when sending messages



Why add stochastic features? And how?

- to model probabilistic behaviours

Example of losses when sending messages



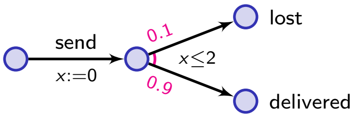
$\rightsquigarrow$ the probabilistic timed automata model used e.g. in PRISM and UPPAAL-PRO [KNSS02]

[KNSS02] Automatic verification of real-time systems with discrete probability distributions *(TCS)*.

Why add stochastic features? And how?

- to model probabilistic behaviours

Example of losses when sending messages



$\rightsquigarrow$ the probabilistic timed automata model used e.g. in PRISM and UPPAAL-PRO [KNSS02]

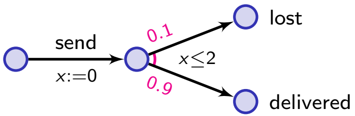
- to model uncertainty on delays

[KNSS02] Automatic verification of real-time systems with discrete probability distributions *(TCS)*.

Why add stochastic features? And how?

- to model probabilistic behaviours

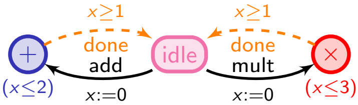
Example of losses when sending messages



$\rightsquigarrow$ the probabilistic timed automata model used e.g. in PRISM and UPPAAL-PRO [KNSS02]

- to model uncertainty on delays

Example of a processor in the taskgraph example

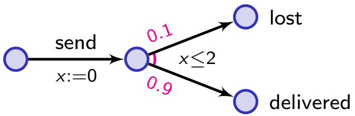


[KNSS02] Automatic verification of real-time systems with discrete probability distributions *(TCS)*.

Why add stochastic features? And how?

- to model probabilistic behaviours

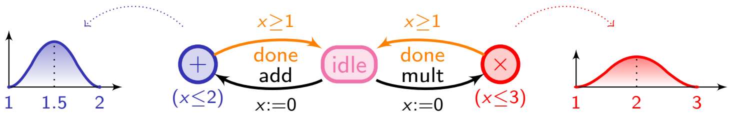
Example of losses when sending messages



⤳ the probabilistic timed automata model used e.g. in PRISM and UPPAAL-PRO [KNSS02]

- to model uncertainty on delays

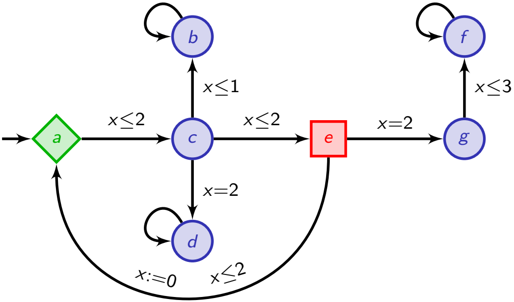
Example of a processor in the taskgraph example



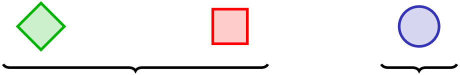
⤳ the stochastic timed automata model [BBB+08,BF09]

[KNSS02] Automatic verification of real-time systems with discrete probability distributions *(TCS)*.
[BBB+08] Baier, Bertrand, Bouyer, Brihaye, Größer. Almost-sure model checking of infinite paths in one-clock timed automata *(LICS'08)*.
[BF09] Bouyer, Forejt. Reachability in stochastic timed games *(ICALP'09)*.

Stochastic timed game: an example

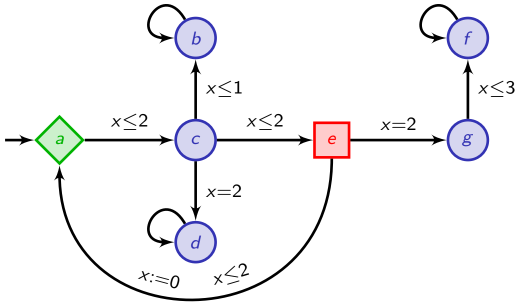


- Timed graph with vertices partitioned among three players:

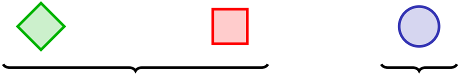


classical players playing “turn-based” | the Nature stochastic player

Stochastic timed game: an example



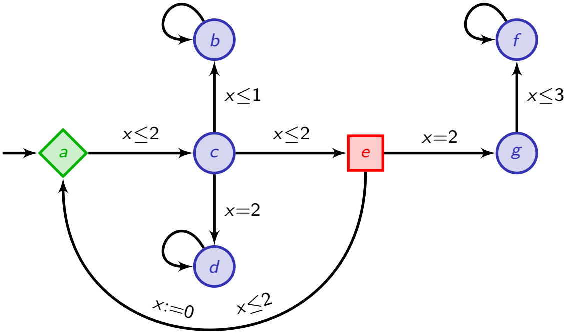
- Timed graph with vertices partitioned among three players:



classical players playing "turn-based" | the Nature stochastic player

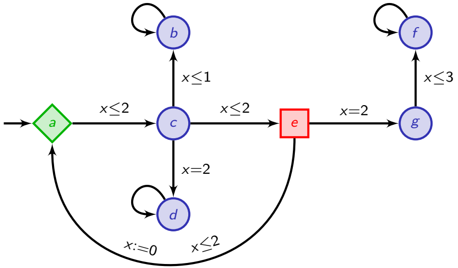
- There are prescribed probability distributions from vertices.

How is this game played?

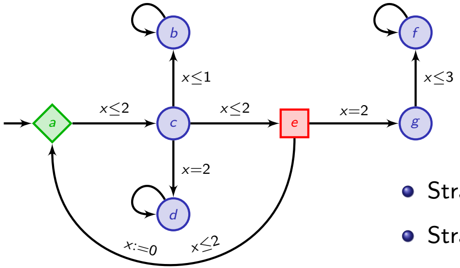


- Players  and  play according to standard strategies
- Player  plays according to the prescribed probability distributions:
  - choose a delay according to some distribution
  - choose an action according to some discrete distribution

Play, an example

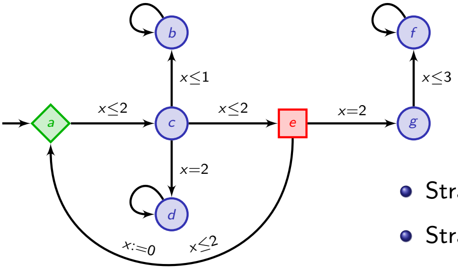


Play, an example



- Strategy for : go to c when $x = 1$
- Strategy for  : go to g when $x = 2$

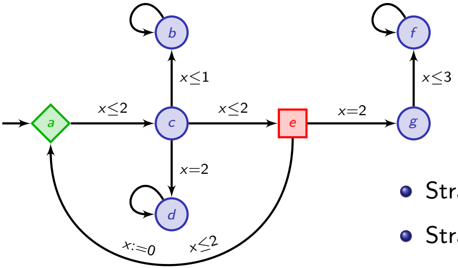
Play, an example



- Strategy for : go to c when $x = 1$
- Strategy for : go to g when $x = 2$

- From the game and the strategies we obtain a Markov chain:

Play, an example

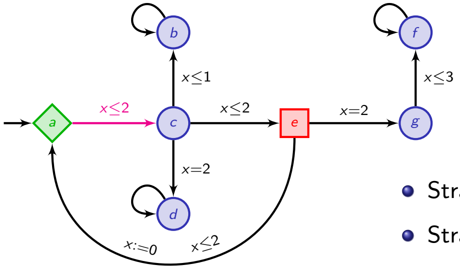


- Strategy for : go to c when $x = 1$
- Strategy for : go to g when $x = 2$

- From the game and the strategies we obtain a Markov chain:

$(a,0)$

Play, an example

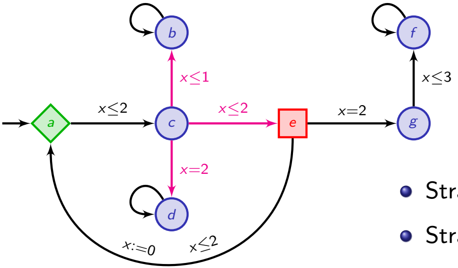


- Strategy for ◇: go to c when $x = 1$
- Strategy for □ : go to g when $x = 2$

- From the game and the strategies we obtain a Markov chain:

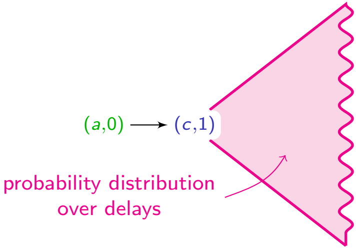
$$(a,0) \longrightarrow (c,1)$$

Play, an example

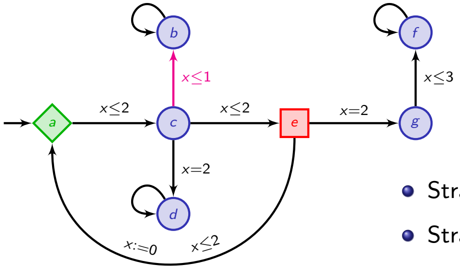


- Strategy for ◇: go to c when $x = 1$
- Strategy for □ : go to g when $x = 2$

- From the game and the strategies we obtain a Markov chain:

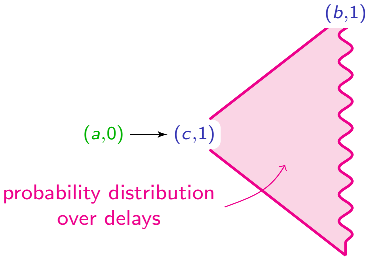


Play, an example

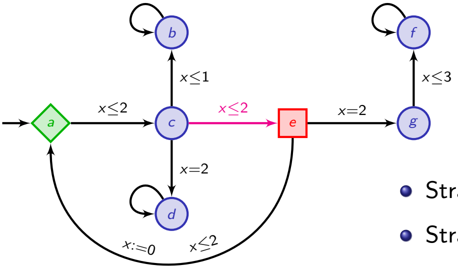


- Strategy for ◇: go to c when $x = 1$
- Strategy for □ : go to g when $x = 2$

- From the game and the strategies we obtain a Markov chain:

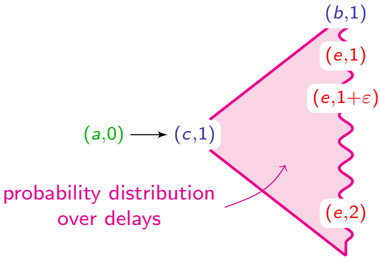


Play, an example

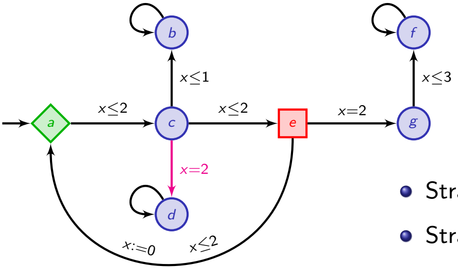


- Strategy for $\Diamond$: go to c when $x = 1$
- Strategy for $\square$: go to g when $x = 2$

- From the game and the strategies we obtain a Markov chain:

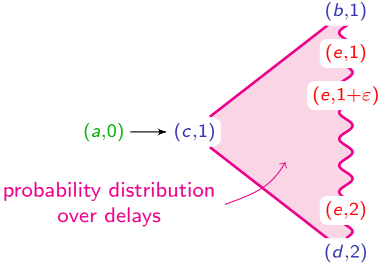


Play, an example

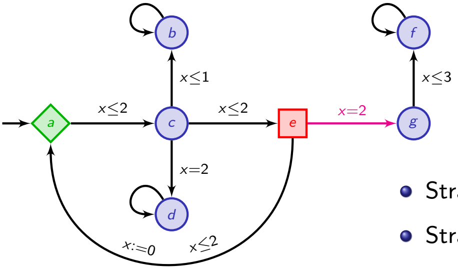


- Strategy for ◇: go to c when $x = 1$
- Strategy for □ : go to g when $x = 2$

- From the game and the strategies we obtain a Markov chain:

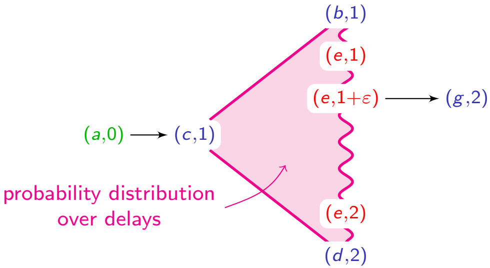


Play, an example

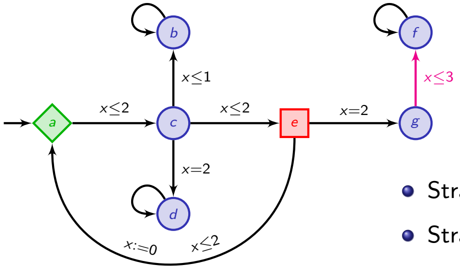


- Strategy for $\Diamond$: go to c when $x = 1$
- Strategy for $\Box$: go to g when $x = 2$

- From the game and the strategies we obtain a Markov chain:

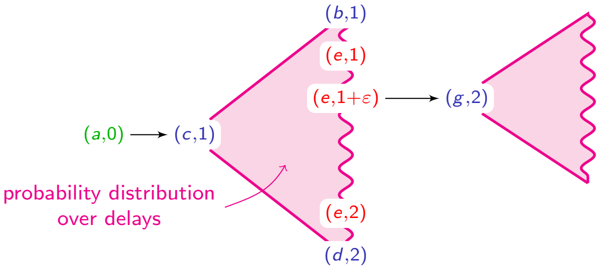


Play, an example

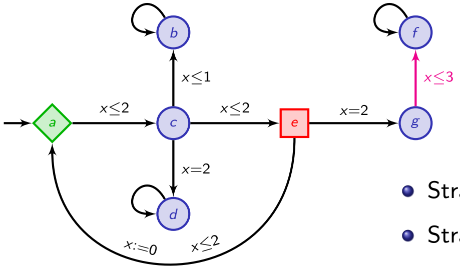


- Strategy for ◇: go to c when $x = 1$
- Strategy for □ : go to g when $x = 2$

- From the game and the strategies we obtain a Markov chain:

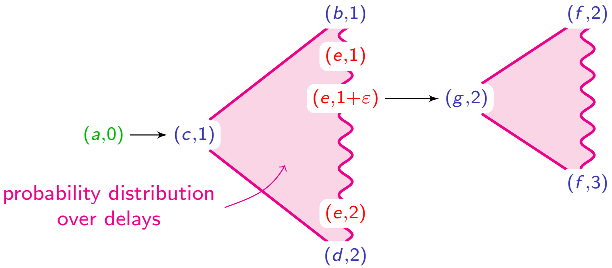


Play, an example



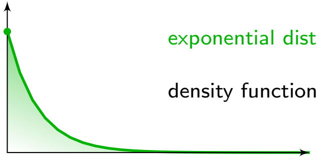
- Strategy for ◇: go to c when $x = 1$
- Strategy for □ : go to g when $x = 2$

- From the game and the strategies we obtain a Markov chain:



How can we attach probabilities to delays?

- The example of continuous-time Markov chains

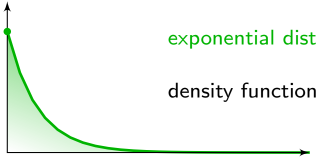


exponential distribution

density function $t \mapsto \begin{cases} \lambda \cdot \exp(-\lambda t) & \text{if } t \geq 0 \\ 0 & \text{otherwise} \end{cases}$

How can we attach probabilities to delays?

- The example of continuous-time Markov chains



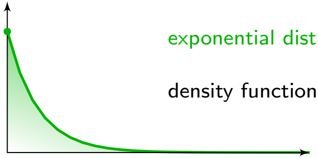
exponential distribution

density function $t \mapsto \begin{cases} \lambda \cdot \exp(-\lambda t) & \text{if } t \geq 0 \\ 0 & \text{otherwise} \end{cases}$

$\rightsquigarrow$ this is ok if delays are in $[0, +\infty)$

How can we attach probabilities to delays?

- The example of continuous-time Markov chains



exponential distribution

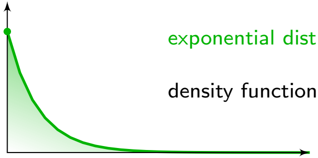
density function $t \mapsto \begin{cases} \lambda \cdot \exp(-\lambda t) & \text{if } t \geq 0 \\ 0 & \text{otherwise} \end{cases}$

$\rightsquigarrow$ this is ok if delays are in $[0, +\infty)$

- But what if bounded intervals?

How can we attach probabilities to delays?

- The example of continuous-time Markov chains

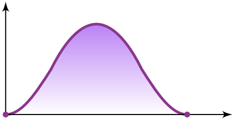


exponential distribution

density function $t \mapsto \begin{cases} \lambda \cdot \exp(-\lambda t) & \text{if } t \geq 0 \\ 0 & \text{otherwise} \end{cases}$

$\rightsquigarrow$ this is ok if delays are in $[0, +\infty)$

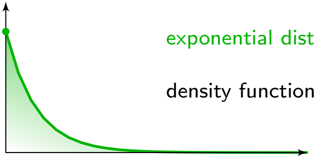
- But what if bounded intervals?



truncated normal distribution

How can we attach probabilities to delays?

- The example of continuous-time Markov chains

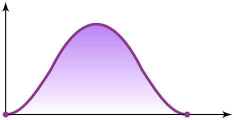


exponential distribution

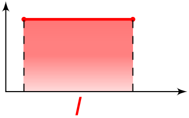
density function $t \mapsto \begin{cases} \lambda \cdot \exp(-\lambda t) & \text{if } t \geq 0 \\ 0 & \text{otherwise} \end{cases}$

$\rightsquigarrow$ this is ok if delays are in $[0, +\infty)$

- But what if bounded intervals?



truncated normal distribution



uniform distribution
density function $t \mapsto \begin{cases} \frac{1}{|I|} & \text{if } t \geq 0 \\ 0 & \text{otherwise} \end{cases}$

The semantics for $\frac{1}{2}$-player games in a nutshell

- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \ldots \xrightarrow{e_n})$

The semantics for $\frac{1}{2}$-player games in a nutshell

- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \ldots \xrightarrow{e_n})$
- Idea:

From state s:

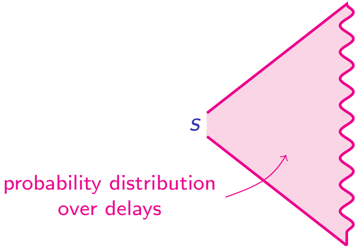
s

The semantics for $\frac{1}{2}$-player games in a nutshell

- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$
- Idea:

From state s:

- randomly choose a delay

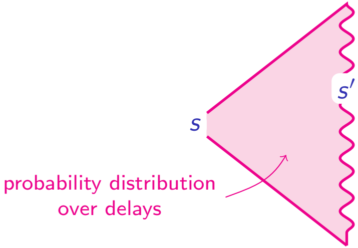


The semantics for $\frac{1}{2}$-player games in a nutshell

- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \ldots \xrightarrow{e_n})$
- Idea:

From state s:

- randomly choose a delay

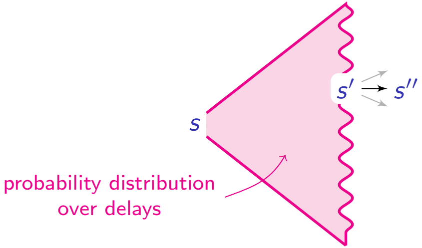


The semantics for $\frac{1}{2}$-player games in a nutshell

- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \ldots \xrightarrow{e_n})$
- Idea:

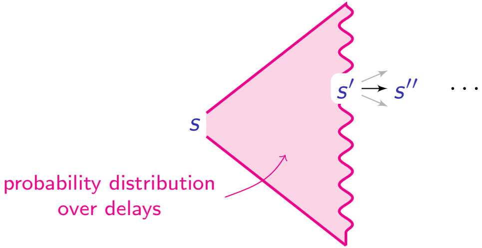
From state s:

- randomly choose a delay
- then randomly select an edge



The semantics for $\frac{1}{2}$-player games in a nutshell

- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$
- Idea:



From state s:

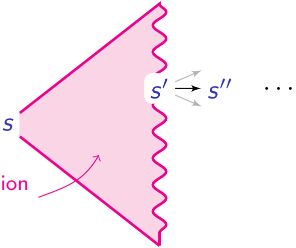
- randomly choose a delay
- then randomly select an edge
- then continue

The semantics for $\frac{1}{2}$-player games in a nutshell

- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$
- Idea:

From state s:

- randomly choose a delay
- then randomly select an edge
- then continue



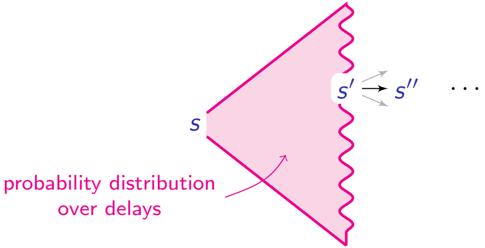
- $$\mathbb{P}\Big(\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\Big) = \int_{t \in I(s,e_1)} p_{s+t}(e_1)\, \mathbb{P}\Big(\pi(s_t \xrightarrow{e_2} \cdots \xrightarrow{e_n})\Big)\, \mathrm{d}\mu_s(t)$$

The semantics for $\frac{1}{2}$-player games in a nutshell

- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$
- Idea:

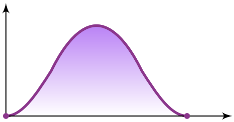
From state s:

- randomly choose a delay
- then randomly select an edge
- then continue



- $$\mathbb{P}\Big(\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\Big) = \int_{t \in I(s,e_1)} p_{s+t}(e_1)\, \mathbb{P}\Big(\pi(s_t \xrightarrow{e_2} \cdots \xrightarrow{e_n})\Big)\, \mathrm{d}\mu_s(t)$$

  - $I(s, e_1) = \{\tau \mid s \xrightarrow{\tau, e_1}\}$ and μ_s distribution over $I(s) = \bigcup_e I(s, e)$

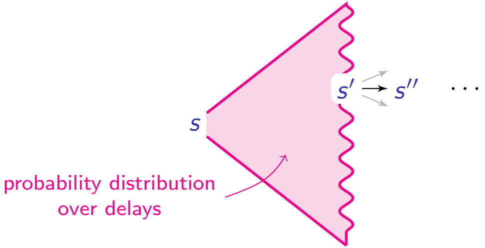


The semantics for $\frac{1}{2}$-player games in a nutshell

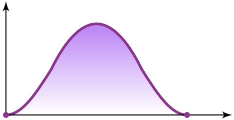
- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})$
- Idea:

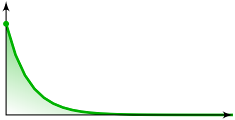
From state s:

- randomly choose a delay
- then randomly select an edge
- then continue



- $$\mathbb{P}\Big(\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\Big) = \int_{t \in I(s,e_1)} p_{s+t}(e_1)\, \mathbb{P}\Big(\pi(s_t \xrightarrow{e_2} \cdots \xrightarrow{e_n})\Big)\, \mathrm{d}\mu_s(t)$$
  - $I(s,e_1) = \{\tau \mid s \xrightarrow{\tau, e_1}\}$ and μ_s distribution over $I(s) = \bigcup_e I(s,e)$



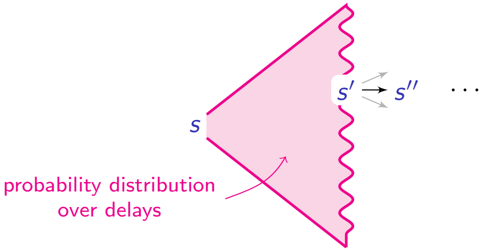


The semantics for $\frac{1}{2}$-player games in a nutshell

- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$
- Idea:

From state s:

- randomly choose a delay
- then randomly select an edge
- then continue



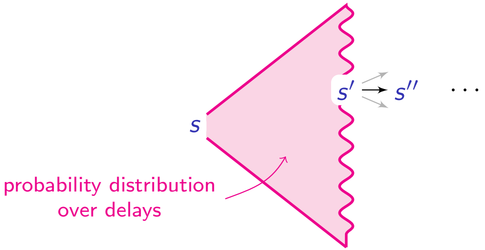
- $$\mathbb{P}\Big(\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\Big) = \int_{t \in I(s,e_1)} p_{s+t}(e_1)\, \mathbb{P}\Big(\pi(s_t \xrightarrow{e_2} \cdots \xrightarrow{e_n})\Big)\, \mathrm{d}\mu_s(t)$$
  - $I(s, e_1) = \{\tau \mid s \xrightarrow{\tau, e_1}\}$ and μ_s distribution over $I(s) = \bigcup_e I(s, e)$
  - p_{s+t} distribution over transitions enabled in $s + t$ (given by weights on transitions)

The semantics for $\frac{1}{2}$-player games in a nutshell

- We measure symbolic cylinders of the form $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$
- Idea:

From state s:

- randomly choose a delay
- then randomly select an edge
- then continue



- $$\mathbb{P}\Big(\pi\big(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}\big)\Big) = \int_{t \in I(s,e_1)} p_{s+t}(e_1)\,\mathbb{P}\Big(\pi\big(s_t \xrightarrow{e_2} \cdots \xrightarrow{e_n}\big)\Big)\,\mathrm{d}\mu_s(t)$$

  - $I(s, e_1) = \{\tau \mid s \xrightarrow{\tau, e_1}\}$ and μ_s distribution over $I(s) = \bigcup_e I(s, e)$
  - p_{s+t} distribution over transitions enabled in $s + t$ (given by weights on transitions)
  - $s \xrightarrow{t} s + t \xrightarrow{e_1} s_t$

Some remarks

- $\frac{1}{2}$-player games define purely stochastic processes.

Some remarks

- $\frac{1}{2}$-player games define purely stochastic processes.
- Continuous-time Markov chains $=$ timed automata with a single "useless" clock which is reset on all transitions. The distributions on delays are exponential distributions with a rate per location.

Some remarks

- $\frac{1}{2}$-player games define purely stochastic processes.
- Continuous-time Markov chains $=$ timed automata with a single "useless" clock which is reset on all transitions. The distributions on delays are exponential distributions with a rate per location.
- The semantics can be extended in a natural way to several players:

$$\mathbb{P}\Big(\pi\big(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}\big)\Big) = \int_{t \in I(s,e_1)} p_{s+t}(e_1)\, \mathbb{P}\Big(\pi\big(s_t \xrightarrow{e_2} \cdots \xrightarrow{e_n}\big)\Big)\, \mathrm{d}\mu_s(t)$$

mass distribution given by the strategy
if s is a player $\Diamond$ vertex

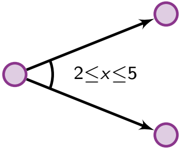
Some remarks

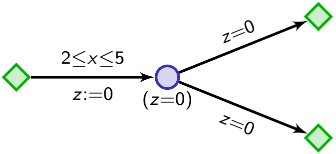
- $\frac{1}{2}$-player games define purely stochastic processes.
- Continuous-time Markov chains = timed automata with a single "useless" clock which is reset on all transitions. The distributions on delays are exponential distributions with a rate per location.
- The semantics can be extended in a natural way to several players:

$$\mathbb{P}\Big(\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\Big) = \int_{t \in I(s,e_1)} p_{s+t}(e_1)\,\mathbb{P}\Big(\pi(s_t \xrightarrow{e_2} \cdots \xrightarrow{e_n})\Big)\,\mathrm{d}\mu_s(t)$$

mass distribution given by the strategy
if s is a player vertex

- Probabilistic timed automata = a subclass of the $1\frac{1}{2}$-player games





The synthesis problem

Problem statement

Given a game G, a (linear-time) property φ, a rational threshold $\bowtie r$,

is there a strategy $f_\Diamond$ for player $\Diamond$ s.t.

for all strategies $f_\Box$ of player $\Box$, $\mathbb{P}\left(G_{f_\Diamond, f_\Box} \models \varphi\right) \bowtie r$?

The synthesis problem

Problem statement

Given a game G, a (linear-time) property φ, a rational threshold $\bowtie r$,

is there a strategy $f_\Diamond$ for player $\Diamond$ s.t.

for all strategies $f_\square$ of player $\square$, $\mathbb{P}\left(G_{f_\Diamond, f_\square} \models \varphi\right) \bowtie r$?

Number of players

- $2\frac{1}{2}$-player games: $\Diamond$ $\square$ $\bigcirc$
- $1\frac{1}{2}$-player games: $\Diamond$ $\bigcirc$ (“Markov decision process”)
- $\frac{1}{2}$-player games: $\bigcirc$ (“Markov chain”)

The synthesis problem

Problem statement

Given a game G, a (linear-time) property φ, a rational threshold $\bowtie r$,

is there a strategy $f_{\diamond}$ for player ◇ s.t.

for all strategies $f_{\square}$ of player □, $\mathbb{P}\left(G_{f_{\diamond},f_{\square}} \models \varphi\right) \bowtie r$?

Number of players

- $2\frac{1}{2}$-player games: ◇ □ ○
- $1\frac{1}{2}$-player games: ◇ ○ ("Markov decision process")
- $\frac{1}{2}$-player games: ○ ("Markov chain")

- Possibility to ask 'performance evaluation' questions [BHHK10]

[BHHK10] Baier, Hermanns, Haverkort, Katoen. Performance Evaluation and Model Checking Join Forces *(CACM)*.

Why model time-dependent resources?

- System resources might be relevant and even crucial information

Why model time-dependent resources?

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

Why model time-dependent resources?

- 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!

Why model time-dependent resources?

- 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

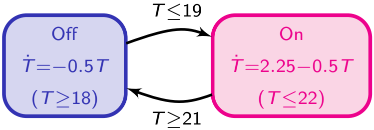
Why model time-dependent resources?

- 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



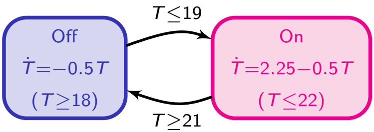
Why model time-dependent resources?

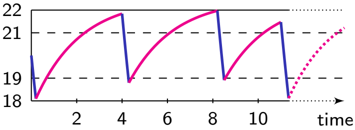
- System resources might be relevant and even crucial information
  - energy consumption,
  - memory usage,
  - bandwidth,
  - …
  - price to pay,
  - benefits,
  - temperature,

  $\rightsquigarrow$ timed automata are not powerful enough!

- A possible solution: use hybrid automata

The thermostat example





Why model time-dependent resources?

- 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

Theorem [HKPV95]

The reachability problem is undecidable in hybrid automata.

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

Why model time-dependent resources?

- 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]

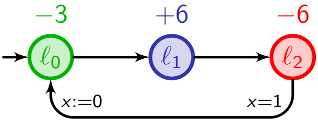
  $\rightsquigarrow$ hybrid variables are observer variables
  (they do not constrain *a priori* the system)

[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)



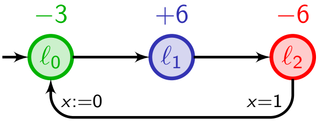
Run $(\ell_0, 0) \xrightarrow{\text{delay}(\frac{1}{6})} (\ell_0, \frac{1}{6}) \rightarrow (\ell_1, \frac{1}{6}) \xrightarrow{\text{delay}(\frac{1}{2})} (\ell_1, \frac{2}{3}) \rightarrow (\ell_2, \frac{2}{3}) \ldots$

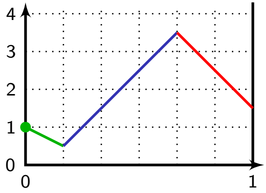
[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

A simple example of weighted timed automata (WTA)

[ALP01,BFH+01]

Example (with a linear observer)





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

[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)*.

The taskgraph scheduling example

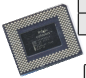
Compute $D \times (C \times (A+B)) + (A+B) + (C \times D)$ using two processors:

P_1 (fast):

time	
$+$	2 picoseconds
$\times$	3 picoseconds

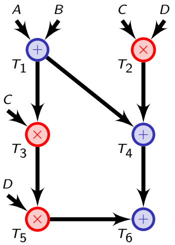
energy	
idle	10 Watt
in use	90 Watts

P_2 (slow):



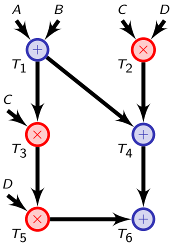
time	
$+$	5 picoseconds
$\times$	7 picoseconds

energy	
idle	20 Watts
in use	30 Watts



The taskgraph scheduling example

Compute $D\times(C\times(A+B))+(A+B)+(C\times D)$ using two processors:

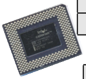


P_1 (fast):

time	
+	2 picoseconds
×	3 picoseconds

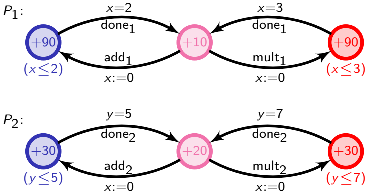
energy	
idle	10 Watt
in use	90 Watts

P_2 (slow):

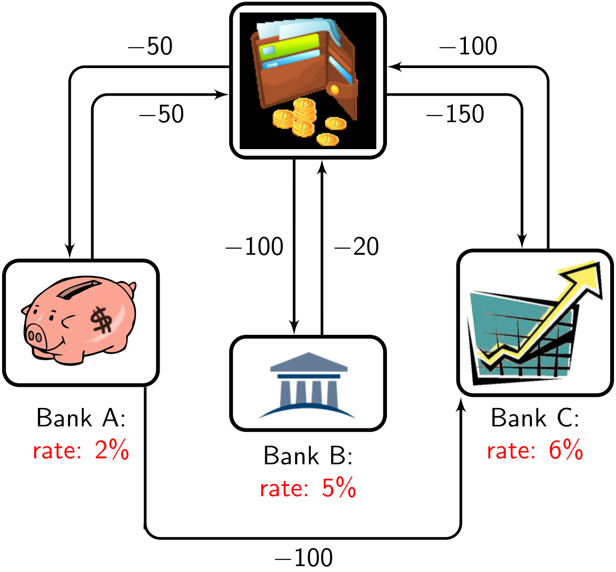


time	
+	5 picoseconds
×	7 picoseconds

energy	
idle	20 Watts
in use	30 Watts



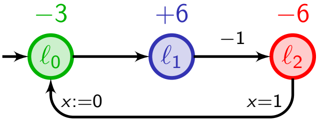
Beyond linear observers...



Beyond linear observers...

Example

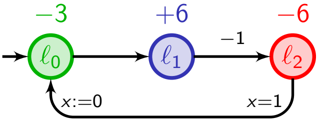
We also consider PTA with an exponential observer:



Beyond linear observers...

Example

We also consider PTA with an exponential observer:



Rate -3 in location ℓ_0 means

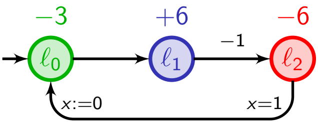
$$\frac{\partial \text{ cost}}{\partial \text{ time}} = -3 \times \text{cost}$$

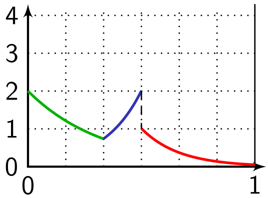
$$\text{cost} = \text{cost}_0 \cdot e^{-3 \times t}$$

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)

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)

- 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)*.

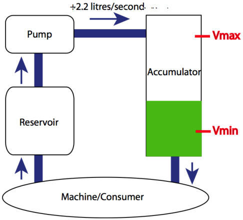
Relevant questions

- Various optimization questions (optimal reachability, optimal mean-cost or discounted infinite schedules, *etc*)

  ⤳ an abundant literature since 2001 (for the linear observers only)

- Scheduling under energy constraints (resource management): are there scheduling policies/strategies when energy is constrained? [BFLMS08]

  ⤳ *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)*.

Conclusion

- Timed automata have been proven to be a convenient model for representing real-time systems
- However it is not expressive enough to faithfully represent some important features of systems
  - interaction with the environment (antagonistic, stochastic, cooperative...)
  - modelling of resources or energy
  - probabilities
- A number of extensions have been proposed to adequately represent such features (we can mix them)
  - The algorithmics of such systems is difficult (in general)
  - But a huge effort is put to develop methods for (approximate) verification