Stochastic Timed Automata

Patricia Bouyer-Decitre

LSV, CNRS & ENS Cachan, France

Based on joint works with Nathalie Bertrand, Thomas Brihaye, Pierre Carlier, Quentin Menet, Christel Baier, ...

Outline

A story that started in 2006...

- My background: timed automata

A story that started in 2006...

- My background: timed automata
- I was strolling around with Thomas Brihaye in Seattle (venue of LICS'06), after having attended a very inspiring talk
  - "Temporal Logics and Model Checking for Fairly Correct Systems"
  - By Daniele Varacca and Hagen Völzer

A story that started in 2006...

- My background: timed automata
- I was strolling around with Thomas Brihaye in Seattle (venue of LICS'06), after having attended a very inspiring talk
  - "Temporal Logics and Model Checking for Fairly Correct Systems"
  - By Daniele Varacca and Hagen Völzer
- We had in mind that this would be relevant to add probabilities to timed automata

A story that started in 2006...

- My background: timed automata
- I was strolling around with Thomas Brihaye in Seattle (venue of LICS'06), after having attended a very inspiring talk
  - "Temporal Logics and Model Checking for Fairly Correct Systems"
  - By Daniele Varacca and Hagen Völzer
- We had in mind that this would be relevant to add probabilities to timed automata
- Why probabilities?
  - for modelling uncertainty (imprecisions in system inputs, unpredictable delays)
  - for measuring system performance (time before failure, ...)
  - for modelling soft real-time delays

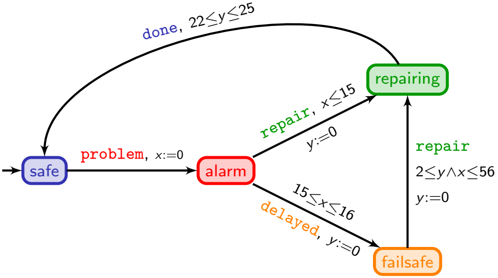
A story that started in 2006...

- My background: timed automata
- I was strolling around with Thomas Brihaye in Seattle (venue of LICS'06), after having attended a very inspiring talk
  - "Temporal Logics and Model Checking for Fairly Correct Systems"
  - By Daniele Varacca and Hagen Völzer
- We had in mind that this would be relevant to add probabilities to timed automata
- Why probabilities?
  - for modelling uncertainty (imprecisions in system inputs, unpredictable delays)
  - for measuring system performance (time before failure, ...)
  - for modelling soft real-time delays

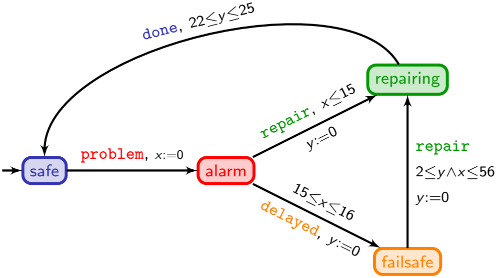
> - FSTTCS'07, LICS'08, QEST'08, QEST'12
> - 72-pages LMCS journal paper
> - Pierre Carlier, joint PhD student between Mons and Cachan, now works on that subject

Outline

The model of timed automata



The model of timed automata



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

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

An example: The task graph scheduling problem

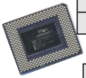
Compute $D\times(C\times(A+B))+(A+B)+(C\times D)$ using two processors:

P_1 (fast):

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

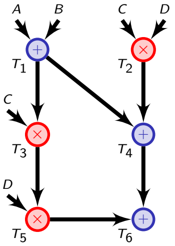
energy	
idle	10 Watt
in use	90 Watts

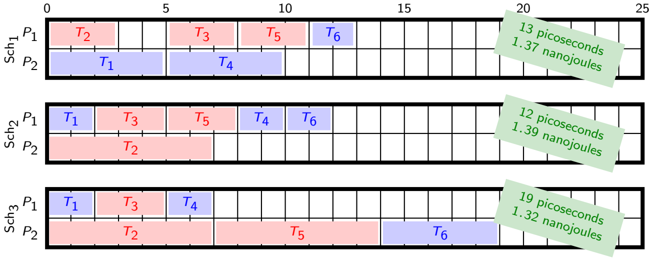
P_2 (slow):



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

energy	
idle	20 Watts
in use	30 Watts

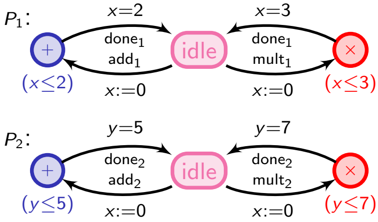




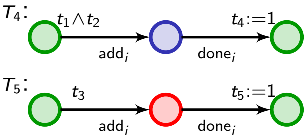
[BFLM10] Bouyer, Fahrenberg, Larsen, Markey. Quantitative Analysis of Real-Time Systems using Priced Timed Automata *(CACM)*.

Modelling the task graph scheduling problem

- Processors

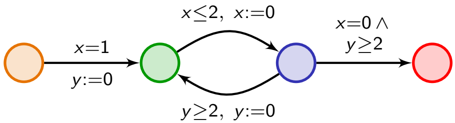


- Tasks

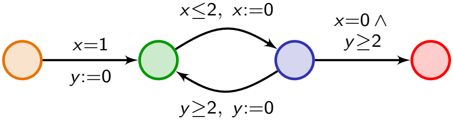


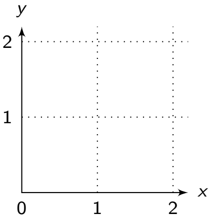
A schedule is a path in the product automaton

Analyzing timed automata

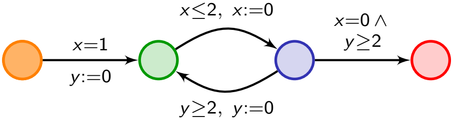


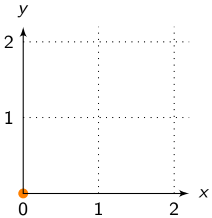
Analyzing timed automata



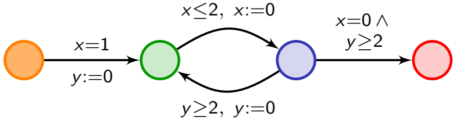


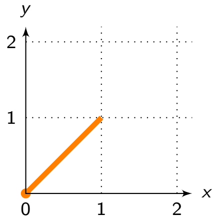
Analyzing timed automata





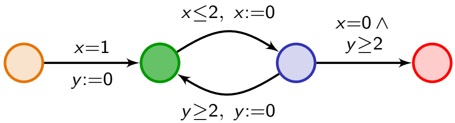
Analyzing timed automata

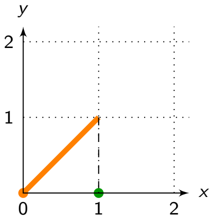




[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

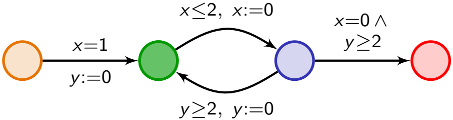
Analyzing timed automata

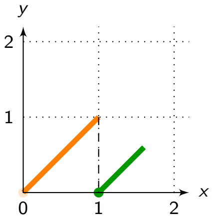




[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

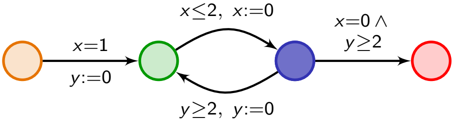
Analyzing timed automata

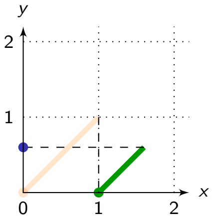




[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

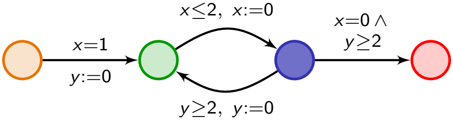
Analyzing timed automata

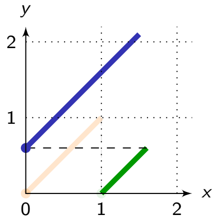




[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

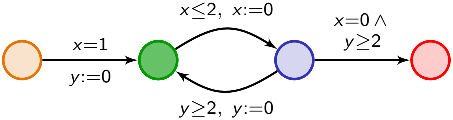
Analyzing timed automata

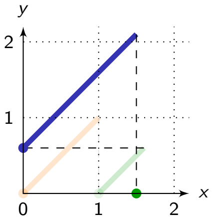




[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

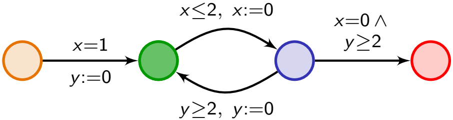
Analyzing timed automata

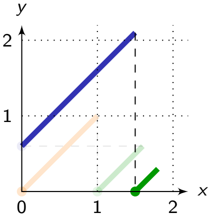




[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

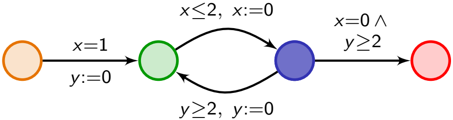
Analyzing timed automata

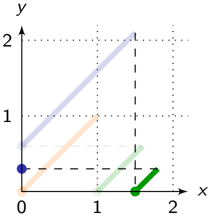




[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

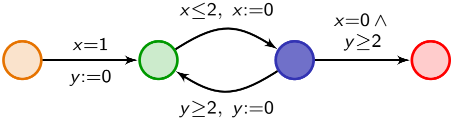
Analyzing timed automata

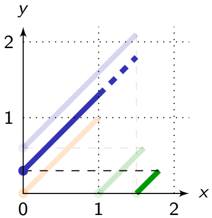




[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

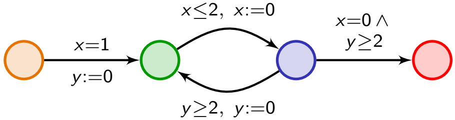
Analyzing timed automata

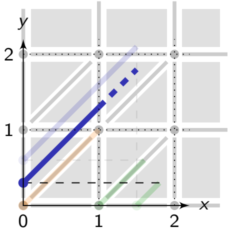




[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

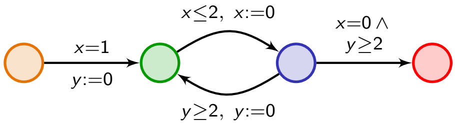
Analyzing timed automata

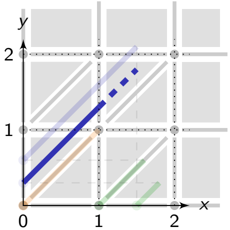




[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

Analyzing timed automata





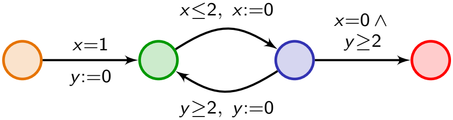
Theorem [AD94]

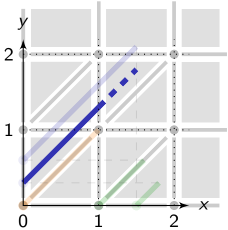
Reachability in timed automata is decidable (as well as many other important properties). It is PSPACE-complete.

- Technical tool: region abstraction

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

Analyzing timed automata





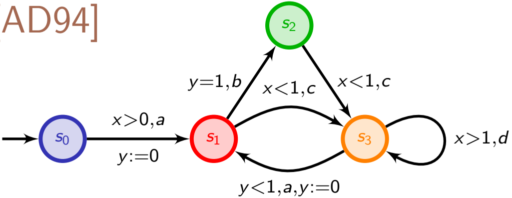
Theorem [AD94]

Reachability in timed automata is decidable (as well as many other important properties). It is PSPACE-complete.

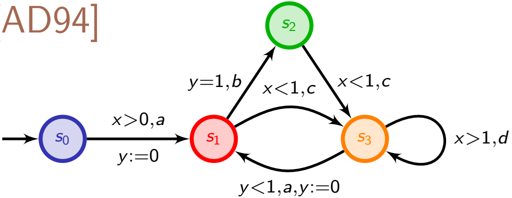
- Technical tool: region abstraction
- Efficient symbolic technics based on zones, implemented in tools

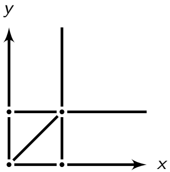
[AD90] Alur, Dill. Automata for modeling real-time systems *(ICALP'90)*.
[AD94] Alur, Dill. A theory of timed automata *(Theoretical Computer Science)*.

An example [AD94]

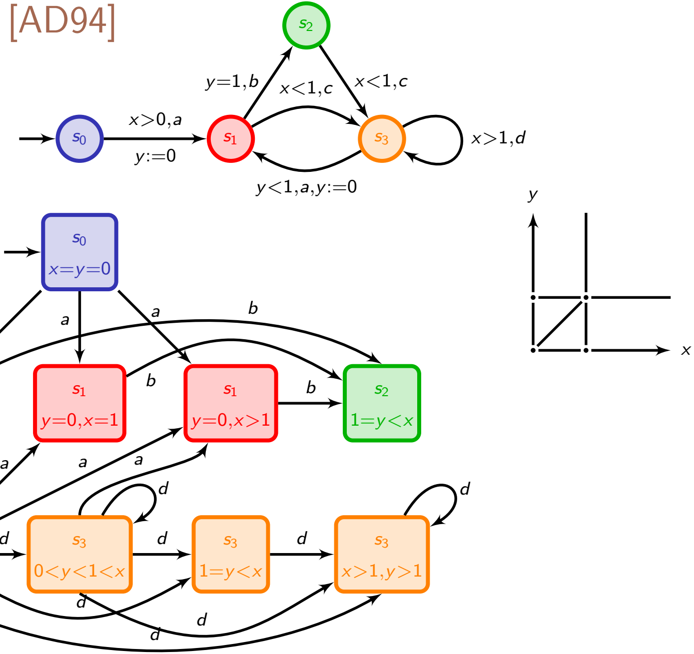


An example [AD94]



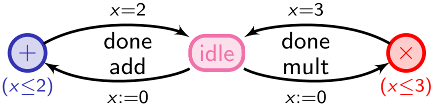


An example [AD94]



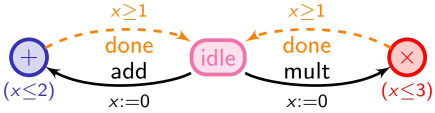
How to model uncertainty over delays?

- Using timed games



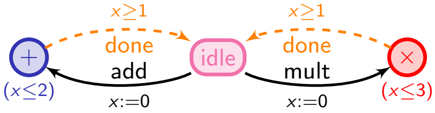
How to model uncertainty over delays?

- Using timed games

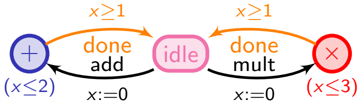


How to model uncertainty over delays?

- Using timed games

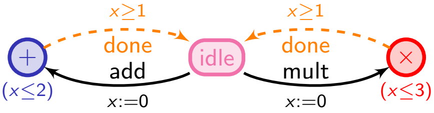


- Using stochastic delays

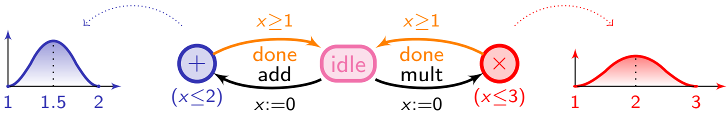


How to model uncertainty over delays?

- Using timed games



- Using stochastic delays



Outline

Existing models?

Models based on timed automata

- Probabilistic timed automata [KNSS99]
  $\rightsquigarrow$ only discrete probabilities over edges
- Continuous probabilistic timed automata [KNSS00]
  $\rightsquigarrow$ resets of clocks are randomized, but only few results

[KNSS99] Kwiatkowska, Norman, Segala, Sproston. Automatic verification of real-time systems with discrete probability distributions *(ARTS'99)*.
[KNSS00] Kwiatkowska, Norman, Segala, Sproston. Verifying quantitative properties of continuous probabilistic timed automata *(CONCUR'00)*.

Existing models?

Models based on timed automata

- Probabilistic timed automata [KNSS99]
  $\rightsquigarrow$ only discrete probabilities over edges
- Continuous probabilistic timed automata [KNSS00]
  $\rightsquigarrow$ resets of clocks are randomized, but only few results

Other related models I was not familiar with in 2006

- Continuous-time Markov chains (CTMCs)
- Generalized semi-Markov processes (GSMPs)
- Process algebras (like Modest) [DK05,BDHK06]

[KNSS99] Kwiatkowska, Norman, Segala, Sproston. Automatic verification of real-time systems with discrete probability distributions *(ARTS'99)*.
[KNSS00] Kwiatkowska, Norman, Segala, Sproston. Verifying quantitative properties of continuous probabilistic timed automata *(CONCUR'00)*.
[DK05] D'Argenio, Katoen. Stochastic timed automata, Part I and Part II *(Information and Computation)*.
[BDHK06] Bohnenkamp, D'Argenio, Hermanns, Katoen. MODEST: A compositional modeling formalism for hard and softly timed systems *(IEEE Trans. Software Engineering)*.

Our choice of stochastic timed automata

- Based on the standard timed automata model
  - Model largely adopted for real-time systems
  - Enjoys efficient verification algorithms and corresponding implementations

Our choice of stochastic timed automata

- Based on the standard timed automata model
  - Model largely adopted for real-time systems
  - Enjoys efficient verification algorithms and corresponding implementations
  - We understand it well!

Our choice of stochastic timed automata

- Based on the standard timed automata model
  - Model largely adopted for real-time systems
  - Enjoys efficient verification algorithms and corresponding implementations
  - We understand it well!

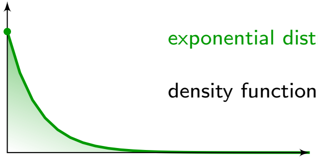
- Randomize delays

Our choice of stochastic timed automata

- Based on the standard timed automata model
  - Model largely adopted for real-time systems
  - Enjoys efficient verification algorithms and corresponding implementations
  - We understand it well!
- Randomize delays
- We believe this is an interesting model for systems integrating both:
  - real-time constraints
  - randomized aspects

How can we attach probabilities to delays?

- The example of continuous-time Markov chains

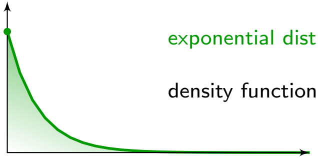


exponential distribution

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



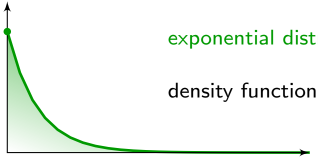
exponential distribution

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

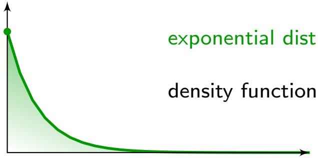
density function $t \mapsto \begin{cases} \lambda \cdot \exp(-\lambda t) & \text{if } t \geq 0 \\ 0 & \text{otherwise} \end{cases}$

$\rightsquigarrow$ 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

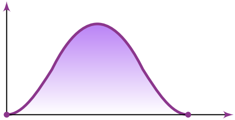


exponential distribution

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 $[0, +\infty)$

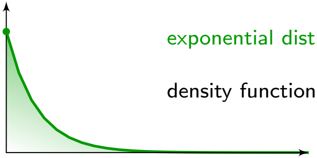
- But what if bounded intervals?



truncated normal distribution

How can we attach probabilities to delays?

- The example of continuous-time Markov chains

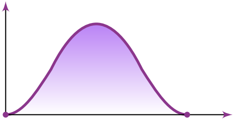


exponential distribution

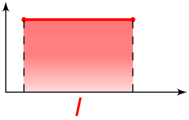
density function $t \mapsto \begin{cases} \lambda \cdot \exp(-\lambda t) & \text{if } t \geq 0 \\ 0 & \text{otherwise} \end{cases}$

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

- But what if bounded intervals?

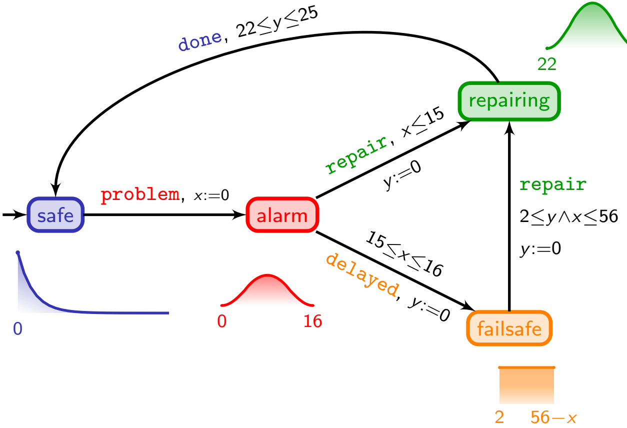


truncated normal distribution



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

How does a STA look like?

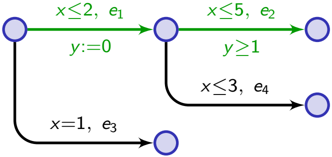


Formalization of the semantics

- $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$: symbolic path from s firing edges $e_1, \dots, e_n$

Formalization of the semantics

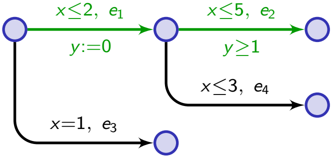
- $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$: symbolic path from s firing edges $e_1, \dots, e_n$
- Example:



$$\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2}) = \{s_0 \xrightarrow{\tau_1, e_1} s_1 \xrightarrow{\tau_2, e_2} s_2 \mid \tau_1 \leq 2,\ \tau_1 + \tau_2 \leq 5,\ \tau_2 \geq 1\}$$

Formalization of the semantics

- $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$: symbolic path from s firing edges $e_1, \dots, e_n$
- Example:



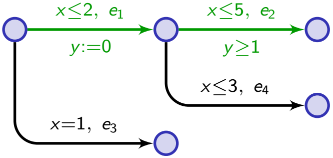
$$\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2}) = \{s_0 \xrightarrow{\tau_1, e_1} s_1 \xrightarrow{\tau_2, e_2} s_2 \mid \tau_1 \leq 2,\ \tau_1 + \tau_2 \leq 5,\ \tau_2 \geq 1\}$$

- Idea: compute the probability of a symbolic path

From state s:

Formalization of the semantics

- $\pi(s \xrightarrow{e_1} \ldots \xrightarrow{e_n})$: symbolic path from s firing edges $e_1, \ldots, e_n$
- Example:

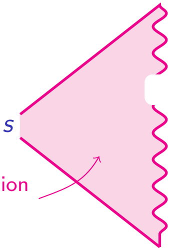


$$\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2}) = \{s_0 \xrightarrow{\tau_1, e_1} s_1 \xrightarrow{\tau_2, e_2} s_2 \mid \tau_1 \leq 2,\ \tau_1 + \tau_2 \leq 5,\ \tau_2 \geq 1\}$$

- Idea: compute the probability of a symbolic path

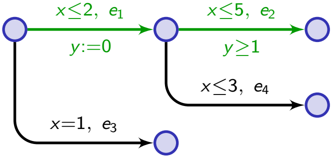
From state s:

- randomly choose a delay



Formalization of the semantics

- $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$: symbolic path from s firing edges $e_1, \dots, e_n$
- Example:

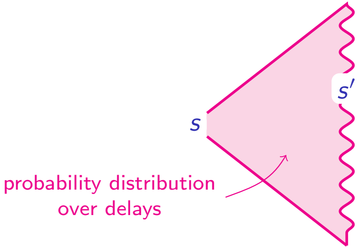


$$\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2}) = \{s_0 \xrightarrow{\tau_1, e_1} s_1 \xrightarrow{\tau_2, e_2} s_2 \mid \tau_1 \leq 2,\ \tau_1 + \tau_2 \leq 5,\ \tau_2 \geq 1\}$$

- Idea: compute the probability of a symbolic path

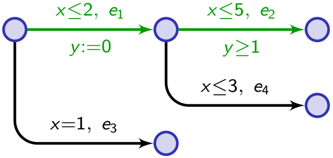
From state s:

- randomly choose a delay



Formalization of the semantics

- $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$: symbolic path from s firing edges $e_1, \dots, e_n$
- Example:

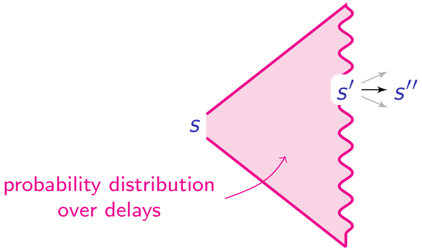


$$\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2}) = \{s_0 \xrightarrow{\tau_1, e_1} s_1 \xrightarrow{\tau_2, e_2} s_2 \mid \tau_1 \leq 2,\ \tau_1 + \tau_2 \leq 5,\ \tau_2 \geq 1\}$$

- Idea: compute the probability of a symbolic path

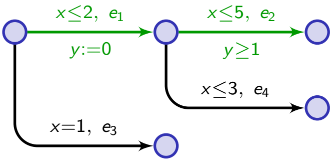
From state s:

- randomly choose a delay
- then randomly select an edge



Formalization of the semantics

- $\pi(s \xrightarrow{e_1} \dots \xrightarrow{e_n})$: symbolic path from s firing edges $e_1, \dots, e_n$
- Example:

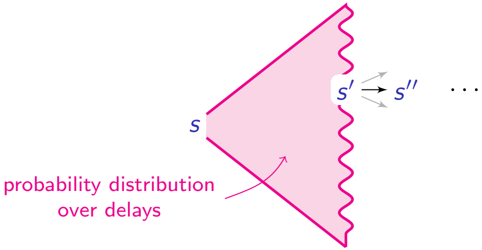


$$\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2}) = \{s_0 \xrightarrow{\tau_1, e_1} s_1 \xrightarrow{\tau_2, e_2} s_2 \mid \tau_1 \leq 2,\ \tau_1 + \tau_2 \leq 5,\ \tau_2 \geq 1\}$$

- Idea: compute the probability of a symbolic path

From state s:

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



Formalization of the semantics

symbolic path: $\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n\}$

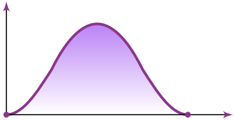
$$\mathbb{P}\big(\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\big) = \int_{t \in I(s,e_1)} p_{s+t}(e_1)\,\mathbb{P}\big(\pi(s_t \xrightarrow{e_2} \cdots \xrightarrow{e_n})\big)\,\mathrm{d}\mu_s(t)$$

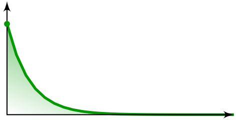
Formalization of the semantics

symbolic path: $\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n\}$

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

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





Formalization of the semantics

symbolic path: $\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n\}$

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

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

Formalization of the semantics

symbolic path: $\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n\}$

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

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

Formalization of the semantics

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

Formalization of the semantics

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

- Can be viewed as an *n*-dimensional integral

Formalization of the semantics

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

- Can be viewed as an *n-dimensional integral*
- Easy extension to constrained symbolic paths

$$\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n \mid (\tau_1, \cdots, \tau_n) \models \mathcal{C}\}$$

Formalization of the semantics

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

- Can be viewed as an n-dimensional integral
- Easy extension to constrained symbolic paths

$$\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n \mid (\tau_1, \cdots, \tau_n) \models \mathcal{C}\}$$

- Definition over sets of infinite runs:

Formalization of the semantics

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

- Can be viewed as an n-dimensional integral
- Easy extension to constrained symbolic paths

$$\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n \mid (\tau_1, \cdots, \tau_n) \models \mathcal{C}\}$$

- Definition over sets of infinite runs:
  - $\mathsf{Cyl}(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})) = \{\varrho \cdot \varrho' \mid \varrho \in \pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\}$

Formalization of the semantics

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

- Can be viewed as an *n*-dimensional integral
- Easy extension to constrained symbolic paths

$$\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n \mid (\tau_1, \cdots, \tau_n) \models \mathcal{C}\}$$

- Definition over sets of infinite runs:
  - $\mathrm{Cyl}(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})) = \{\varrho \cdot \varrho' \mid \varrho \in \pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\}$
  - $\mathbb{P}\big(\mathrm{Cyl}(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}))\big) = \mathbb{P}\big(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\big)$

Formalization of the semantics

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

- Can be viewed as an n-dimensional integral
- Easy extension to constrained symbolic paths

$$\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n \mid (\tau_1, \cdots, \tau_n) \models \mathcal{C}\}$$

- Definition over sets of infinite runs:
  - $\mathrm{Cyl}(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})) = \{\varrho \cdot \varrho' \mid \varrho \in \pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\}$
  - $\mathbb{P}\big(\mathrm{Cyl}(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}))\big) = \mathbb{P}\big(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\big)$
  - unique extension of $\mathbb{P}$ to the generated σ-algebra

Formalization of the semantics

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

- Can be viewed as an n-dimensional integral
- Easy extension to constrained symbolic paths

$$\pi_{\mathcal{C}}\big(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}\big) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n \mid (\tau_1, \cdots, \tau_n) \models \mathcal{C}\}$$

- Definition over sets of infinite runs:
  - $\mathsf{Cyl}(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})) = \{\varrho \cdot \varrho' \mid \varrho \in \pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\}$
  - $\mathbb{P}\Big(\mathsf{Cyl}(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}))\Big) = \mathbb{P}\Big(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\Big)$
  - unique extension of $\mathbb{P}$ to the generated σ-algebra
- Property: $\mathbb{P}$ is a probability measure over sets of infinite runs

Formalization of the semantics

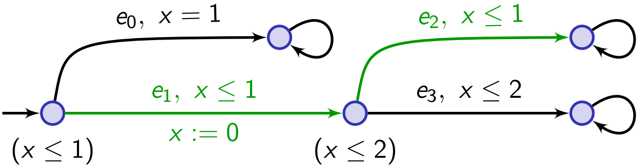
$$\mathbb{P}\big(\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\big) = \int_{t \in I(s,e_1)} p_{s+t}(e_1)\,\mathbb{P}\big(\pi(s_t \xrightarrow{e_2} \cdots \xrightarrow{e_n})\big)\,\mathrm{d}\mu_s(t)$$

- Can be viewed as an *n*-dimensional integral
- Easy extension to constrained symbolic paths

$$\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}) = \{s \xrightarrow{\tau_1, e_1} s_1 \cdots \xrightarrow{\tau_n, e_n} s_n \mid (\tau_1, \cdots, \tau_n) \models \mathcal{C}\}$$

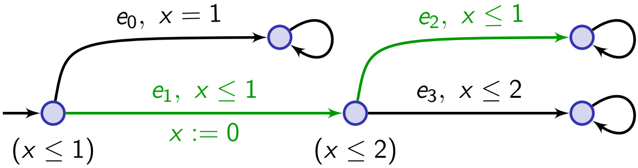
- Definition over sets of infinite runs:
  - $\mathsf{Cyl}(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})) = \{\varrho \cdot \varrho' \mid \varrho \in \pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\}$
  - $\mathbb{P}\big(\mathsf{Cyl}(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}))\big) = \mathbb{P}\big(\pi_{\mathcal{C}}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n})\big)$
  - unique extension of $\mathbb{P}$ to the generated σ-algebra
- Property: $\mathbb{P}$ is a probability measure over sets of infinite runs
- Example:
  - $\mathsf{Zeno}(s) \;=\; \displaystyle\bigcup_{M \in \mathbb{N}} \bigcap_{n \in \mathbb{N}} \bigcup_{(e_1, \cdots, e_n) \in E^n} \mathsf{Cyl}(\pi_{\sum_i \tau_i \leq M}(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}))$

An example of computation (with uniform distributions)



The probability of the symbolic path $\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2})$ is $\frac{1}{4}$.

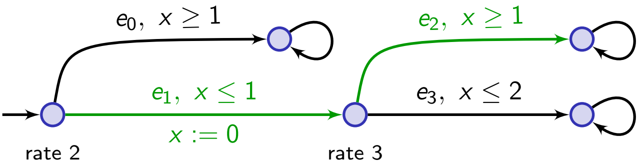
An example of computation (with uniform distributions)



The probability of the symbolic path $\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2})$ is $\frac{1}{4}$.

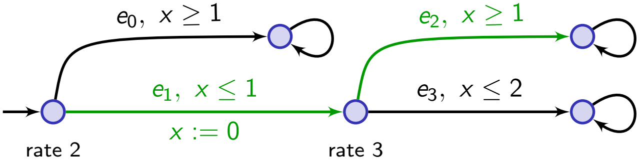
$$\begin{aligned}
\mathbb{P}\big(\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2})\big) &= \int_0^1 \mathbb{P}\big(\pi(s_1 \xrightarrow{e_2})\big)\mathrm{d}\mu_{s_0}(t) + \int_1^1 \frac{\mathbb{P}\big(\pi(s_1 \xrightarrow{e_2})\big)}{2}\mathrm{d}\mu_{s_0}(t) \\
&= \int_0^1 \int_0^1 \left(\frac{\mathbb{P}\big(\pi(s_2)\big)}{2}\mathrm{d}\mu_{s_1}(u)\right)\mathrm{d}\mu_{s_0}(t) \\
&= \int_0^1 \int_0^1 \left(\frac{1}{2}\frac{\mathrm{d}u}{2}\right)\mathrm{d}t \quad = \frac{1}{4}
\end{aligned}$$

An example of computation (with exponential distrib.)



The probability of the symbolic path $\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2})$ is $e^{-3} - e^{-5} \approx 0.043$

An example of computation (with exponential distrib.)



The probability of the symbolic path $\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2})$ is $e^{-3} - e^{-5} \approx 0.043$

$$\begin{aligned}
\mathbb{P}\big(\pi(s_0 \xrightarrow{e_1}\xrightarrow{e_2})\big) &= \int_0^1 \mathbb{P}\big(\pi(s_1 \xrightarrow{e_2})\big)\mathrm{d}\mu_{s_0}(t) = \int_0^1 \mathbb{P}\big(\pi(s_1 \xrightarrow{e_2})\big) 2\exp(-2t)\mathrm{d}t \\
&= \int_0^1 \left(\int_1^{+\infty} 3\exp(-3u)\mathrm{d}u\right) 2\exp(-2t)\mathrm{d}t \\
&= [-\exp(-2t)]_{t=0}^{1} \cdot [-\exp(-3u)]_{u=1}^{+\infty} \\
&= (1-e^{-2}) \cdot e^{-3} = e^{-3} - e^{-5}
\end{aligned}$$

Some remarks

- This defines a purely stochastic process

Some remarks

- This defines a purely stochastic process
- Continuous-time Markov chains = STA 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

- This defines a purely stochastic process
- Continuous-time Markov chains = STA with a single "useless" clock which is reset on all transitions. The distributions on delays are exponential distributions with a rate per location
- Finite-state generalized semi-Markov processes (residual-lifetime semantics) are STAs (if no fixed-delay events)

Some remarks

- This defines a purely stochastic process
- Continuous-time Markov chains = STA with a single “useless” clock which is reset on all transitions. The distributions on delays are exponential distributions with a rate per location
- Finite-state generalized semi-Markov processes (residual-lifetime semantics) are STAs (if no fixed-delay events)
- Allows to express richer timing constraints

Outline

Almost-sure model-checking

We are interested in (automatic) model-checking algorithms!

- Qualitative model-checking: decide whether

$$\mathbb{P}\big(\{\varrho \in \mathsf{Runs}(s) \mid \varrho \models \varphi\}\big) = 1$$

  We write $s \models\!\!\!\approx \varphi$ whenever it is the case.
  This is the almost-sure model-checking problem.

- Quantitative model-checking: compute (or approximate) the value

$$\mathbb{P}\big(\{\varrho \in \mathsf{Runs}(s) \mid \varrho \models \varphi\}\big)$$

Almost-sure model-checking

We are interested in (automatic) model-checking algorithms!

- Qualitative model-checking: decide whether

$$\mathbb{P}\big(\{\varrho \in \mathsf{Runs}(s) \mid \varrho \models \varphi\}\big) = 1$$

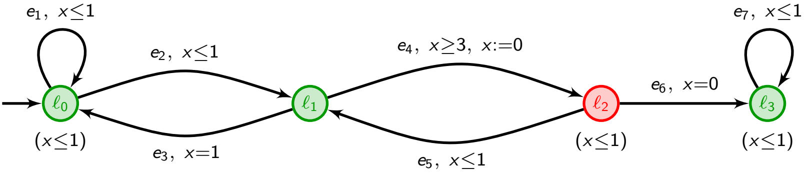
We write $s \models\!\!\!\approx \varphi$ whenever it is the case.
This is the almost-sure model-checking problem.

- Quantitative model-checking: compute (or approximate) the value

$$\mathbb{P}\big(\{\varrho \in \mathsf{Runs}(s) \mid \varrho \models \varphi\}\big)$$

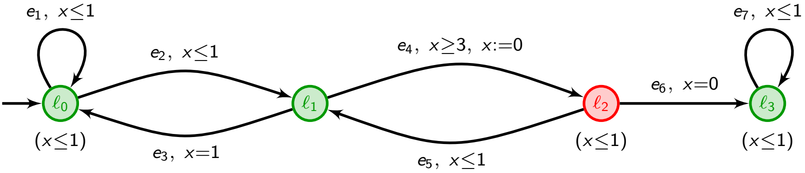
In this talk we focus on
the almost-sure model-checking problem.

An example



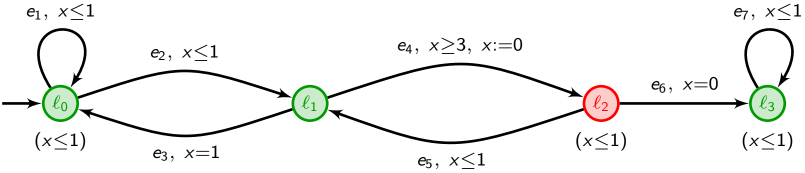
$e_1,\ x\leq 1$
$e_2,\ x\leq 1$
$e_3,\ x=1$
$e_4,\ x\geq 3,\ x:=0$
$e_5,\ x\leq 1$
$e_6,\ x=0$
$e_7,\ x\leq 1$
ℓ_0
ℓ_1
ℓ_2
ℓ_3
$(x\leq 1)$
$(x\leq 1)$
$(x\leq 1)$

An example



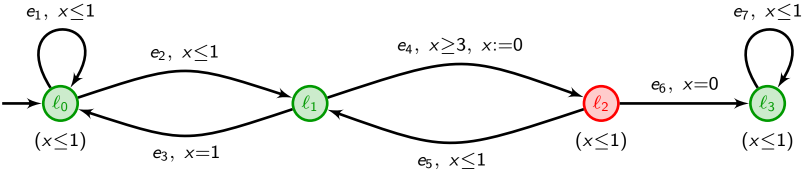
$\mathcal{A} \not\models \mathbf{G}(\text{green} \Rightarrow \mathbf{F}\ \text{red})$

An example



$$\mathcal{A} \not\models \mathbf{G}(\text{green} \Rightarrow \mathbf{F}\,\text{red}) \qquad \text{but} \qquad \mathbb{P}\big(\mathcal{A} \models \mathbf{G}(\text{green} \Rightarrow \mathbf{F}\,\text{red})\big) = 1$$

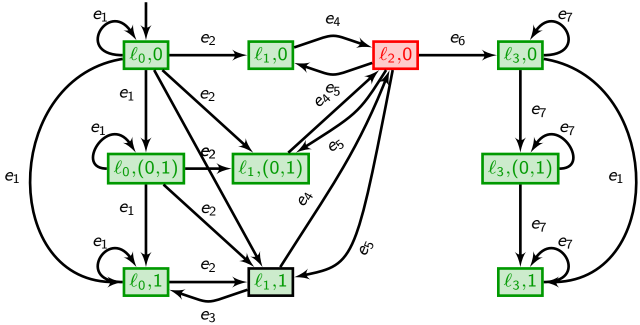
An example



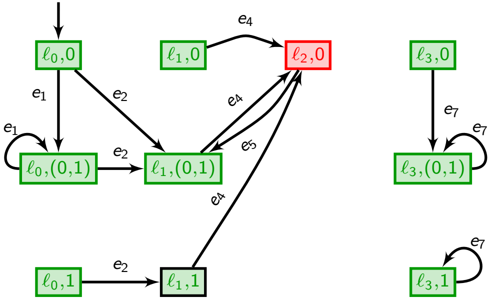
$\mathcal{A} \not\models \mathbf{G}(\text{green} \Rightarrow \mathbf{F}\,\text{red})$ but $\mathbb{P}\big(\mathcal{A} \models \mathbf{G}(\text{green} \Rightarrow \mathbf{F}\,\text{red})\big) = 1$

Indeed, almost surely, paths are of the form $e_1^* e_2 \big(e_4 e_5\big)^\omega$

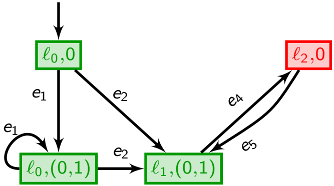
The classical region automaton



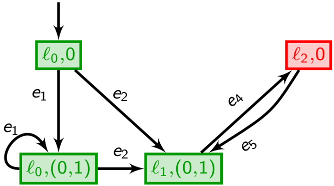
The pruned region automaton



The pruned region automaton

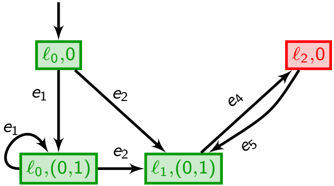


The pruned region automaton



... viewed as a finite Markov chain $MC(\mathcal{A})$

The pruned region automaton

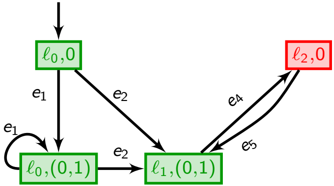


... viewed as a finite Markov chain $MC(\mathcal{A})$

It holds as well that:

$$\mathbb{P}\big(MC(\mathcal{A}) \models \mathbf{G}(\text{green} \Rightarrow \mathbf{F}\,\text{red})\big) = 1$$

The pruned region automaton



... viewed as a finite Markov chain $MC(\mathcal{A})$

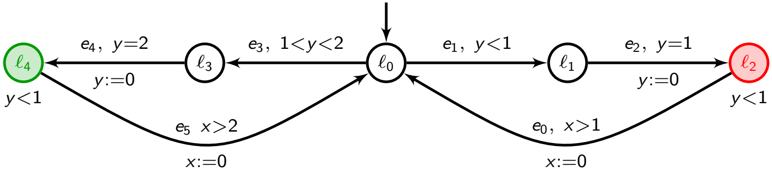
It holds as well that:

$$\mathbb{P}\big(MC(\mathcal{A}) \models \mathbf{G}(\text{green} \Rightarrow \mathbf{F}\ \text{red})\big) = 1$$

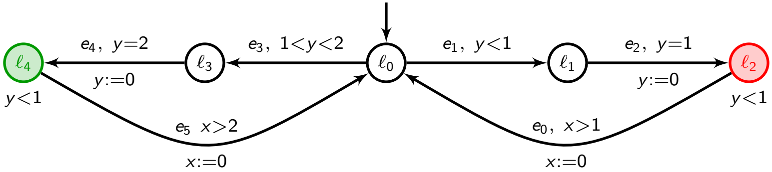
When is that the case that

$$\mathbb{P}\big(\mathcal{A} \models \varphi\big) = 1 \quad \text{iff} \quad \mathbb{P}\big(MC(\mathcal{A}) \models \varphi\big) = 1\ ?$$

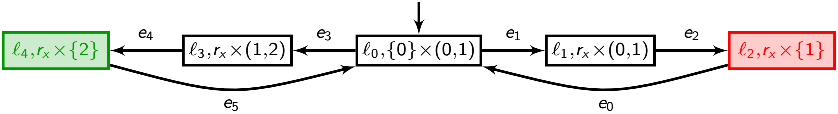
A counter-example



A counter-example

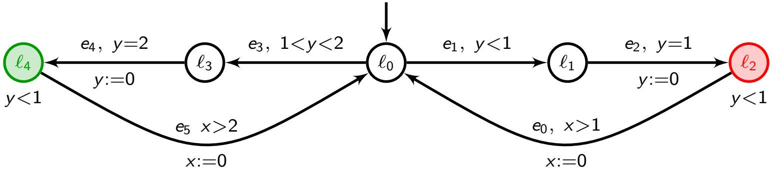


The pruned region automaton viewed as a finite Markov chain $MC(\mathcal{A})$:

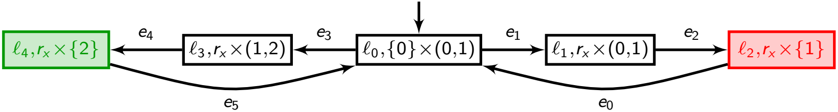


$$\varphi \equiv (\mathbf{GF}\ \text{green}) \wedge (\mathbf{GF}\ \text{red})$$

A counter-example



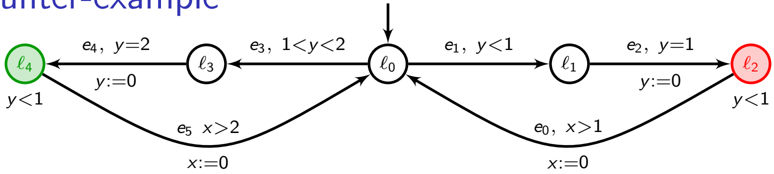
The pruned region automaton viewed as a finite Markov chain $MC(\mathcal{A})$:



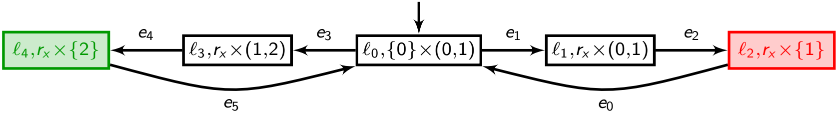
$$\varphi \equiv (\mathbf{GF}\ \text{green}) \wedge (\mathbf{GF}\ \text{red})$$

We clearly have that $\mathbb{P}(MC(\mathcal{A}) \models \varphi) = 1$ BUT $\mathbb{P}(\mathcal{A} \models \varphi) < 1$.

A counter-example



The pruned region automaton viewed as a finite Markov chain $MC(\mathcal{A})$:



$$\varphi \ \equiv\ (\mathbf{GF}\ \text{green})\ \wedge\ (\mathbf{GF}\ \text{red})$$

Let y_n be the value of y at the n^{th} arrival in ℓ_0

$$y_n < 1 \qquad \text{and} \qquad y_n < y_{n+1}$$

Main decidability results [BBB+14]

Theorem

Let $\mathcal{A}$ be a STA and φ a safety property. Then:

$$\mathbb{P}(\mathcal{A} \models \varphi) = 1 \quad \text{iff} \quad \mathbb{P}(MC(\mathcal{A}) \models \varphi) = 1$$

[BBB+14] Bertrand, Bouyer, Brihaye, Menet, Baier, Größer, Jurdziński. Stochastic timed automata *(Logical Methods in Computer Science)*.

Main decidability results [BBB+14]

Theorem

Let $\mathcal{A}$ be a STA and φ a safety property. Then:

$$\mathbb{P}(\mathcal{A} \models \varphi) = 1 \quad \text{iff} \quad \mathbb{P}(MC(\mathcal{A}) \models \varphi) = 1$$

Theorem

Let $\mathcal{A}$ be a STA and φ an ω-regular property.
If $\mathbb{P}(\mathcal{A} \models \text{fair}) = 1$ then

$$\mathbb{P}(\mathcal{A} \models \varphi) = 1 \quad \text{iff} \quad \mathbb{P}(MC(\mathcal{A}) \models \varphi) = 1$$

[BBB+14] Bertrand, Bouyer, Brihaye, Menet, Baier, Größer, Jurdziński. Stochastic timed automata *(Logical Methods in Computer Science)*.

Main decidability results [BBB+14]

Theorem

Let $\mathcal{A}$ be a STA and φ a safety property. Then:

$$\mathbb{P}(\mathcal{A} \models \varphi) = 1 \quad \text{iff} \quad \mathbb{P}(MC(\mathcal{A}) \models \varphi) = 1$$

Theorem

Let $\mathcal{A}$ be a STA and φ an ω-regular property.
If $\mathbb{P}(\mathcal{A} \models \text{fair}) = 1$ then

$$\mathbb{P}(\mathcal{A} \models \varphi) = 1 \quad \text{iff} \quad \mathbb{P}(MC(\mathcal{A}) \models \varphi) = 1$$

Fairness is a semantic condition:

every edge which is enabled infinitely often is taken infinitely often

[BBB+14] Bertrand, Bouyer, Brihaye, Menet, Baier, Größer, Jurdziński. Stochastic timed automata *(Logical Methods in Computer Science)*.

Main decidability results [BBB+14]

Theorem

Let $\mathcal{A}$ be a STA and φ a safety property. Then:

$$\mathbb{P}(\mathcal{A} \models \varphi) = 1 \quad \text{iff} \quad \mathbb{P}(MC(\mathcal{A}) \models \varphi) = 1$$

Theorem

Let $\mathcal{A}$ be a STA and φ an ω-regular property.
If $\mathbb{P}(\mathcal{A} \models \mathsf{fair}) = 1$ then

$$\mathbb{P}(\mathcal{A} \models \varphi) = 1 \quad \text{iff} \quad \mathbb{P}(MC(\mathcal{A}) \models \varphi) = 1$$

Fairness is a semantic condition:

every thick edge of the region graph which is enabled infinitely often is taken infinitely often

[BBB+14] Bertrand, Bouyer, Brihaye, Menet, Baier, Größer, Jurdziński. Stochastic timed automata *(Logical Methods in Computer Science)*.

Main decidability results [BBB+14]

Theorem

Let $\mathcal{A}$ be a STA and φ a safety property. Then:

$$\mathbb{P}(\mathcal{A} \models \varphi) = 1 \quad \text{iff} \quad \mathbb{P}(MC(\mathcal{A}) \models \varphi) = 1$$

Theorem

Let $\mathcal{A}$ be a STA and φ an ω-regular property.
If $\mathbb{P}(\mathcal{A} \models \text{fair}) = 1$ then

$$\mathbb{P}(\mathcal{A} \models \varphi) = 1 \quad \text{iff} \quad \mathbb{P}(MC(\mathcal{A}) \models \varphi) = 1$$

- Proof based on a topology over the set of paths
- Notions of largeness (for proba 1) and meagerness (for proba 0)
- Link between probabilities and topology thanks to the topological games called Banach-Mazur games

[BBB+14] Bertrand, Bouyer, Brihaye, Menet, Baier, Größer, Jurdziński. Stochastic timed automata *(Logical Methods in Computer Science)*.

Almost-sure fairness?

- Finite Markov chains are almost-surely fair

Almost-sure fairness?

- Finite Markov chains are almost-surely fair
  Decisive Markov chains are almost-surely fair

$$\mathbb{P}\big(\mathbf{F}\ T \vee \mathbf{F}\ \widetilde{T}\big) = 1$$

[ABM07] Abdulla, Ben Henda, Mayr. Decisive Markov chains *(Logical Methods in Computer Science)*.

Almost-sure fairness?

- Finite Markov chains are almost-surely fair
  Decisive Markov chains are almost-surely fair

$$\mathbb{P}\Big(\mathbf{F}\ T \vee \mathbf{F}\ \widetilde{T}\Big) = 1$$

- Are STA almost-surely fair?

Almost-sure fairness?

- Finite Markov chains are almost-surely fair
  Decisive Markov chains are almost-surely fair

$$\mathbb{P}\Big(\mathbf{F}\ T \vee \mathbf{F}\ \widetilde{T}\Big) = 1$$

- Are STA almost-surely fair? No!

Results [BBB+14]

Theorem

The following classes of STAs are almost-surely fair:

- single-clock STAs
- (weak-)reactive STAs

[BBB+14] Bertrand, Bouyer, Brihaye, Menet, Baier, Größer, Jurdziński. Stochastic timed automata *(Logical Methods in Computer Science)*.

Results [BBB+14]

Theorem

The following classes of STAs are almost-surely fair:

- single-clock STAs
- (weak-)reactive STAs

Reactive: for every $s = (\ell, v)$, $I(s) = \mathbb{R}_+$, and constant distributions within a location

[BBB+14] Bertrand, Bouyer, Brihaye, Menet, Baier, Größer, Jurdziński. Stochastic timed automata *(Logical Methods in Computer Science)*.

Results [BBB+14]

Theorem

The following classes of STAs are almost-surely fair:

- single-clock STAs
- (weak-)reactive STAs

(Note: CTMCs are reactive STAs)

Reactive: for every $s = (\ell, v)$, $I(s) = \mathbb{R}_+$, and constant distributions within a location

[BBB+14] Bertrand, Bouyer, Brihaye, Menet, Baier, Größer, Jurdziński. Stochastic timed automata *(Logical Methods in Computer Science)*.

Results [BBB+14]

Theorem

The following classes of STAs are almost-surely fair:

- single-clock STAs
- (weak-)reactive STAs

(Note: CTMCs are reactive STAs)

Reactive: for every $s = (\ell, v)$, $I(s) = \mathbb{R}_+$, and constant distributions within a location

Corollary

The almost-sure model-checking of ω-regular properties in single-clock (resp. reactive) STAs can be decided in NLOGSPACE (resp. PSPACE).
The almost-sure model-checking of of LTL properties in single-clock or reactive STAs can be decided in PSPACE.

[BBB+14] Bertrand, Bouyer, Brihaye, Menet, Baier, Größer, Jurdziński. Stochastic timed automata *(Logical Methods in Computer Science)*.

Ingredients of the proofs

- Proof for single-clock STAs:
  - Technical analysis of single-clock STAs
  - Fairness over compact subsets of $\mathbb{R}_+$

Ingredients of the proofs

- Proof for single-clock STAs:
  - Technical analysis of single-clock STAs
  - Fairness over compact subsets of $\mathbb{R}_+$
- Reactive STAs:
  - There exists $\epsilon > 0$ such that for all s, $\mu_s(]M, +\infty[) > \epsilon$
  - Notion of memoryless region: for every x, either $x = 0$ or $x > M$
  - Borel-Cantelli lemma

Assume $(\mathcal{E}, \mathbb{P})$ is a probabilistic space, and that the measurable events $(E_k)_{k \in \mathbb{N}}$ are independent. If $\sum_{k \in \mathbb{N}} \mathbb{P}(E_k) = +\infty$, then

$$\mathbb{P}\left(\bigcap_{n \in \mathbb{N}} \bigcup_{k \geq n} E_k\right) = 1\,.$$

A note on Zeno behaviours

- The set of Zeno behaviours is measurable:

$$\mathrm{Zeno}(s) \;=\; \bigcup_{M\in\mathbb{N}} \bigcap_{n\in\mathbb{N}} \bigcup_{(e_1,\cdots,e_n)\in E^n} \mathrm{Cyl}(\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}))$$

A note on Zeno behaviours

- The set of Zeno behaviours is measurable:

$$\mathrm{Zeno}(s) = \bigcup_{M \in \mathbb{N}} \bigcap_{n \in \mathbb{N}} \bigcup_{(e_1, \cdots, e_n) \in E^n} \mathrm{Cyl}(\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}))$$

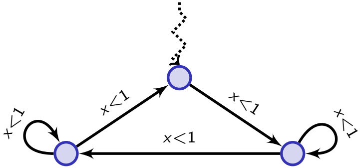
- In single-clock timed automata, we can decide in NLOGSPACE whether $\mathbb{P}(\mathrm{Zeno}(s)) = 0$:

A note on Zeno behaviours

- The set of Zeno behaviours is measurable:

$$\mathrm{Zeno}(s) = \bigcup_{M\in\mathbb{N}} \bigcap_{n\in\mathbb{N}} \bigcup_{(e_1,\cdots,e_n)\in E^n} \mathrm{Cyl}\Big(\pi\big(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}\big)\Big)$$

- In single-clock timed automata, we can decide in NLOGSPACE whether $\mathbb{P}\big(\mathrm{Zeno}(s)\big) = 0$:
  - check whether there is a purely Zeno BSCC in $MC(\mathcal{A})$

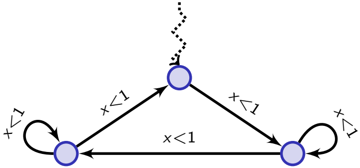


A note on Zeno behaviours

- The set of Zeno behaviours is measurable:

$$\mathrm{Zeno}(s) \;=\; \bigcup_{M\in\mathbb{N}} \bigcap_{n\in\mathbb{N}} \bigcup_{(e_1,\cdots,e_n)\in E^n} \mathrm{Cyl}(\pi(s \xrightarrow{e_1} \cdots \xrightarrow{e_n}))$$

- In single-clock timed automata, we can decide in NLOGSPACE whether $\mathbb{P}(\mathrm{Zeno}(s)) = 0$:
  - check whether there is a purely Zeno BSCC in $MC(\mathcal{A})$



  - an interesting notion of non-Zeno timed automata

- In reactive STAs, Zeno behaviours have probability 0

Outline

Challenge: compositional design of STA

Problematic

componentwise description of systems involving timed constraints and stochastic uncertainties

Challenge: compositional design of STA

Problematic

componentwise description of systems involving timed constraints and stochastic uncertainties

How can we compose STAs?

- First step: a parallel composition operator
- Second step: add interaction
  - Game extensions studied so far not adequate [BF09,BS12]
  - Planned solution: interaction *à la* Interactive Markov Chains [Her02]
    Note: inspiring discussion in [HK09]

[BF09] Bouyer, Forejt. Reachability in stochastic timed games *(ICALP'09)*.
[BS12] Bertrand, Schewe. Playing optimally on timed automata with random delays *(FORMATS'12)*.
[Her02] Hermanns. Interactive Markov chains: The quest for quantified quality *(LNCS 2428)*.
[HK09] Hermanns, Katoen. The how and why of interactive Markov chains *(FMCO'09)*.

Challenge: compositional design of STA

Problematic

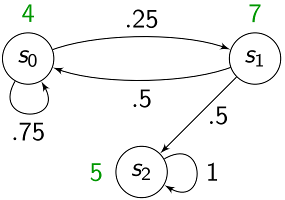
componentwise description of systems involving timed constraints and stochastic uncertainties

How can we compose STAs?

- First step: a parallel composition operator
- Second step: add interaction
  - Game extensions studied so far not adequate [BF09,BS12]
  - Planned solution: interaction *à la* Interactive Markov Chains [Her02]
    Note: inspiring discussion in [HK09]

[BF09] Bouyer, Forejt. Reachability in stochastic timed games *(ICALP'09)*.
[BS12] Bertrand, Schewe. Playing optimally on timed automata with random delays *(FORMATS'12)*.
[Her02] Hermanns. Interactive Markov chains: The quest for quantified quality *(LNCS 2428)*.
[HK09] Hermanns, Katoen. The how and why of interactive Markov chains *(FMCO'09)*.

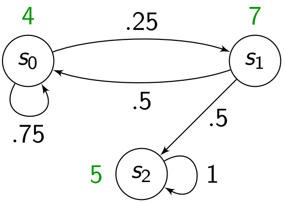
A quick look at CTMCs

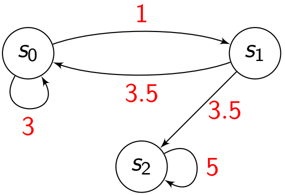


A dual representation of CTMCs...

- Exit rates $r(\cdot)$ of states
  (parameters of the exp. distributions)

A quick look at CTMCs



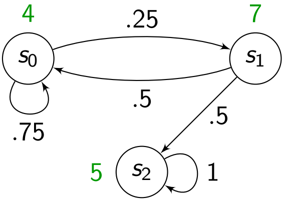


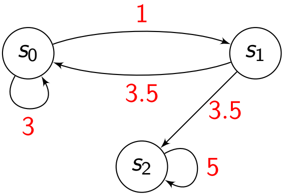
A dual representation of CTMCs...

- Exit rates $r(\cdot)$ of states
  (parameters of the exp. distributions)
- Time-abstract rates $R(\cdot,\cdot)$ of edges

$$R(s_0, s_1) = r(s_0) \cdot p(s_0, s_1)$$

A quick look at CTMCs

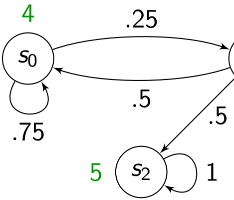


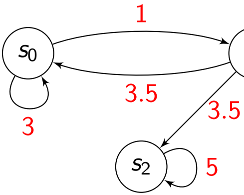


A dual representation of CTMCs...

- Exit rates $r(\cdot)$ of states (parameters of the exp. distributions)
- Time-abstract rates $R(\cdot,\cdot)$ of edges
$$R(s_0,s_1) = r(s_0)\cdot p(s_0,s_1)$$
- $r(s_0)$ is the rate of the min. distrib. of rates $R(s_0,s_1)$ and $R(s_0,s_0)$

A quick look at CTMCs





A dual representation of CTMCs...

- Exit rates $r(\cdot)$ of states
  (parameters of the exp. distributions)
- Time-abstract rates $R(\cdot,\cdot)$ of edges

$$R(s_0, s_1) = r(s_0) \cdot p(s_0, s_1)$$

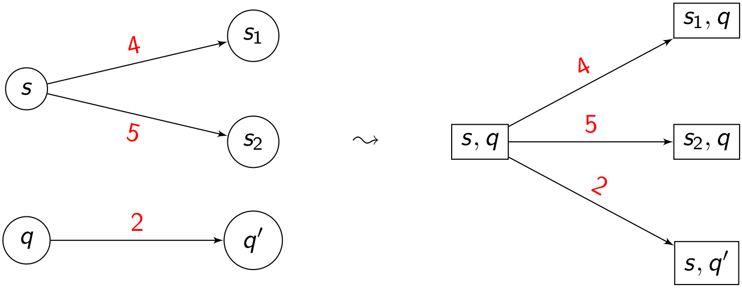
- $r(s_0)$ is the rate of the min. distrib. of rates $R(s_0, s_1)$ and $R(s_0, s_0)$

... which allows some computations

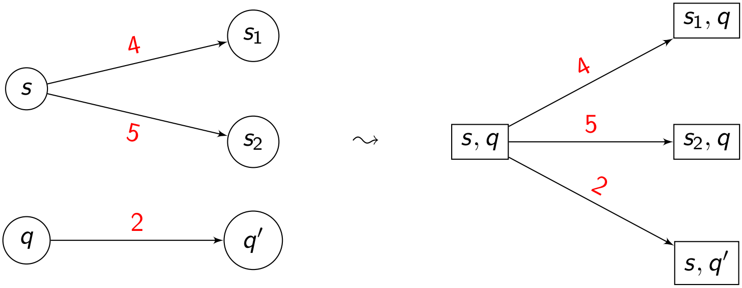
- The probability to move from s_0 to s_1 within $[0, t]$ is:

$$\frac{R(s_0, s_1)}{r(s_0)} \cdot (1 - \exp(-r(s_0) \cdot t))$$

Composition of CTMCs made easy

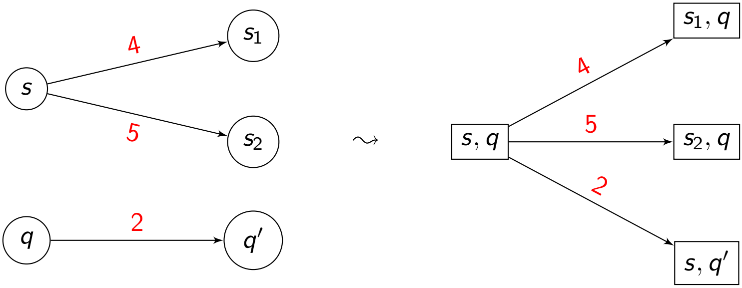


Composition of CTMCs made easy



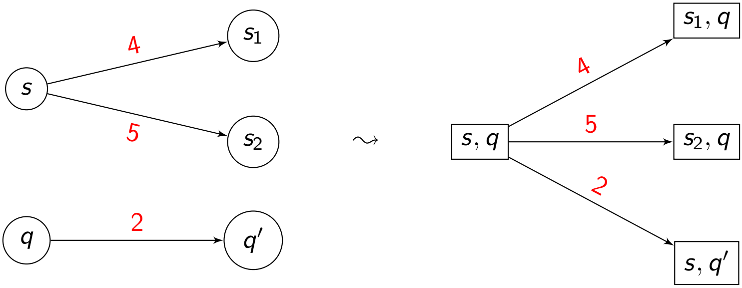
- Race between (s, s_1), (s, s_2) and (q, q')

Composition of CTMCs made easy



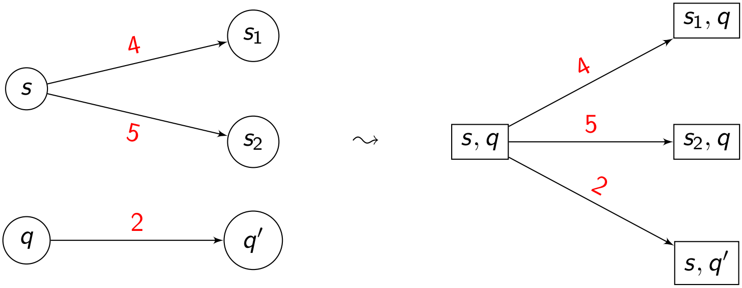
- Race between (s, s_1), (s, s_2) and (q, q')
- If (s, s_i) wins, the system moves to (s_i, q)

Composition of CTMCs made easy



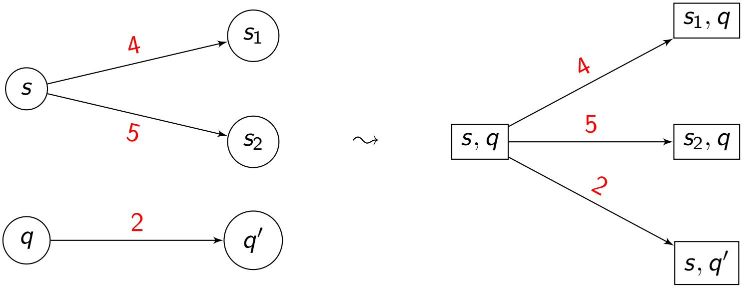
- Race between (s, s_1), (s, s_2) and (q, q')
- If (s, s_i) wins, the system moves to (s_i, q)
- There is a new race between edges from s_i and (q, q') again

Composition of CTMCs made easy



- Race between (s, s_1), (s, s_2) and (q, q')
- If (s, s_i) wins, the system moves to (s_i, q)
- There is a new race between edges from s_i and (q, q') again
- Correct since exp. distrib. are memoryless!

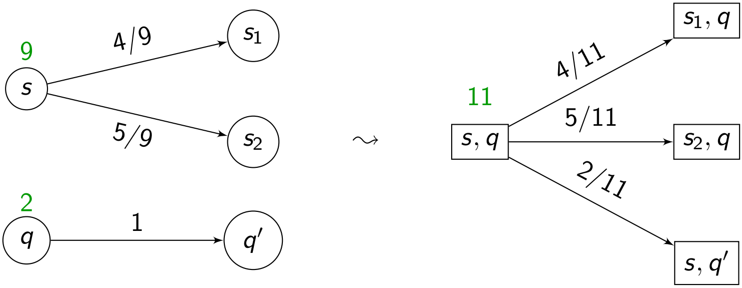
Composition of CTMCs made easy



- Race between (s, s_1), (s, s_2) and (q, q')
- If (s, s_i) wins, the system moves to (s_i, q)
- There is a new race between edges from s_i and (q, q') again
- Correct since exp. distrib. are memoryless!
  If X is a r.v. following an exp. distrib.

$$\text{Prob}(X \geq t + t' \mid X \geq t) = \text{Prob}(X \geq t')$$

Composition of CTMCs made easy



- Race between (s, s_1), (s, s_2) and (q, q')
- If (s, s_i) wins, the system moves to (s_i, q)
- There is a new race between edges from s_i and (q, q') again
- Correct since exp. distrib. are memoryless!
  If X is a r.v. following an exp. distrib.

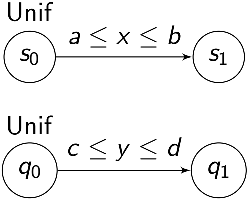
$$\mathrm{Prob}(X \geq t + t' \mid X \geq t) = \mathrm{Prob}(X \geq t')$$

How does that extend to STAs?

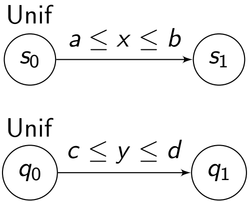
Difficulties

- we have to handle guards
- we have to compose more general continuous distributions
  $\rightsquigarrow$ should represent a race between the components
- we want to preserve the structure of the product automaton
- the product should be "interleaving"

An example where everything goes well



An example where everything goes well



Unif. distrib. over $[a, b]$

- Density function:

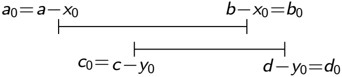
$$f(t) = \begin{cases} 0 & \text{if } t < a \text{ or } t > b \\ \frac{1}{b-a} & \text{if } a \leq t \leq b \end{cases}$$

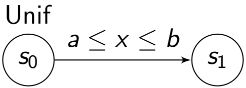
- Cumulative function:

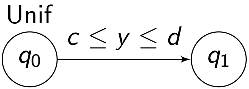
$$\begin{aligned} F(t) &= \int_{d \leq t} f(t)\,\mathrm{d}t \\ &= \begin{cases} 0 & \text{if } t < a \\ \frac{t-a}{b-a} & \text{if } a < t < b \\ 1 & \text{if } t > b \end{cases} \end{aligned}$$

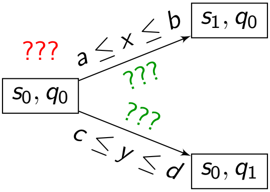
An example where everything goes well

Assume $x_0 \le a$ and $y_0 \le c$ are s.t.

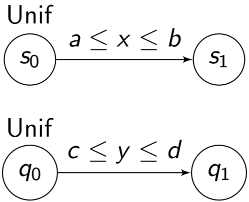


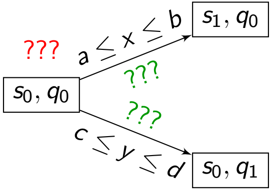




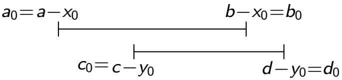


An example where everything goes well





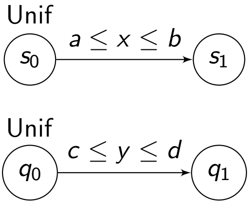
Assume $x_0 \leq a$ and $y_0 \leq c$ are s.t.

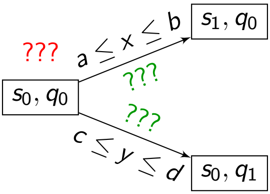


- Distrib. over delays: min. of the two distrib. over delays (race). Its density is:

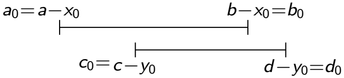
$$f_{x_0,y_0}(t) = \begin{cases} 0 & \text{if } t < a_0 \text{ or } t > b_0 \\ \frac{1}{b_0 - a_0} & \text{if } a_0 < t < c_0 \\ \frac{d_0 + b_0 - 2t}{(b_0 - a_0)\cdot(d_0 - c_0)} & \text{if } c_0 < t < b_0 \end{cases}$$

An example where everything goes well





Assume $x_0 \leq a$ and $y_0 \leq c$ are s.t.



- Distrib. over delays: min. of the two distrib. over delays (race). Its density is:

$$f_{x_0,y_0}(t) = \begin{cases} 0 & \text{if } t < a_0 \text{ or } t > b_0 \\ \frac{1}{b_0 - a_0} & \text{if } a_0 < t < c_0 \\ \frac{d_0 + b_0 - 2t}{(b_0 - a_0)\cdot(d_0 - c_0)} & \text{if } c_0 < t < b_0 \end{cases}$$

- Discrete proba. on edges within $[c_0, b_0]$: proba. that the component has won the race

$$\frac{b_0 - t}{d_0 + b_0 - 2t}, \text{ resp. } \frac{d_0 - t}{d_0 + b_0 - 2t}$$

for the bottom, resp. top, edge.

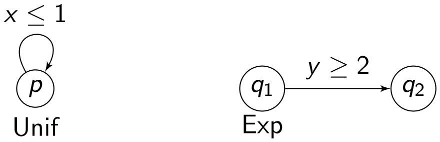
What could go wrong?

A component should not be impacted by the other's actions!

What could go wrong?

A component should not be impacted by the other's actions!

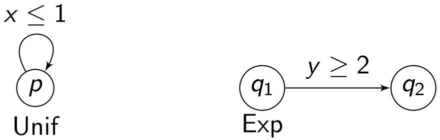
- The first automaton is blocking the second!



What could go wrong?

A component should not be impacted by the other's actions!

- The first automaton is blocking the second!

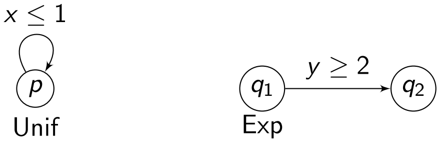


$\rightsquigarrow$ We should assume automata are almost-surely non-Zeno!!

What could go wrong?

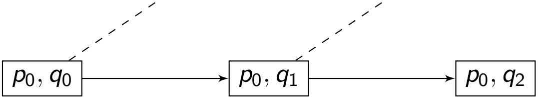
A component should not be impacted by the other's actions!

- The first automaton is blocking the second!



$\rightsquigarrow$ We should assume automata are almost-surely non-Zeno!!

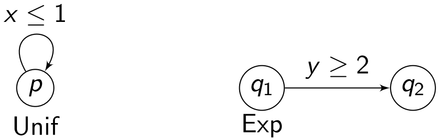
- In the product:



What could go wrong?

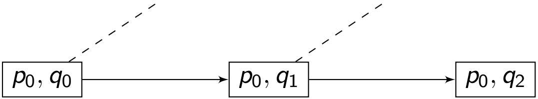
A component should not be impacted by the other's actions!

- The first automaton is blocking the second!



$\leadsto$ We should assume automata are almost-surely non-Zeno!!

- In the product:

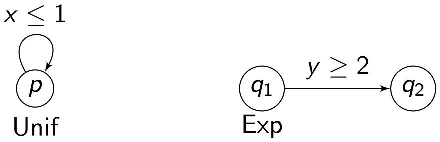


$\leadsto$ The sum of the delays in (p_0, q_0), (p_0, q_1) and (p_0, q_2) should be distributed (for the first component) as a single delay in p_0

What could go wrong?

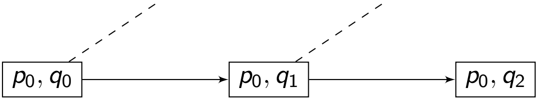
A component should not be impacted by the other's actions!

- The first automaton is blocking the second!



$\rightsquigarrow$ We should assume automata are almost-surely non-Zeno!!

- In the product:



$\rightsquigarrow$ The sum of the delays in (p_0, q_0), (p_0, q_1) and (p_0, q_2) should be distributed (for the first component) as a single delay in p_0

$\rightsquigarrow$ We impose some weak-memorylessness condition on distrib.

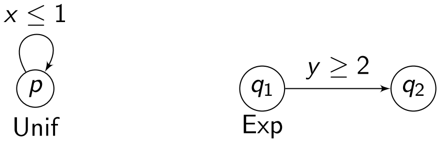
$$\mathrm{Prob}(X_{(\ell,v)} \geq t + t' \mid X_{(\ell,v)} \geq t) = \mathrm{Prob}(X_{(\ell,v+t)} \geq t')$$

($X_{(\ell,v)}$: r.v. for delays from config. (ℓ, v))

What could go wrong?

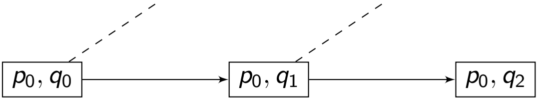
A component should not be impacted by the other's actions!

- The first automaton is blocking the second!



$\rightsquigarrow$ We should assume automata are almost-surely non-Zeno!!

- In the product:



$\rightsquigarrow$ The sum of the delays in (p_0, q_0), (p_0, q_1) and (p_0, q_2) should be distributed (for the first component) as a single delay in p_0

$\rightsquigarrow$ We impose some weak-memorylessness condition on distrib.

$$\text{Prob}(X_{(\ell,v)} \geq t + t' \mid X_{(\ell,v)} \geq t) = \text{Prob}(X_{(\ell,v+t)} \geq t')$$

What does the last weak-memorylessness condition mean?

$$\mathrm{Prob}(X_{(\ell,v)} \geq t + t' \mid X_{(\ell,v)} \geq t) = \mathrm{Prob}(X_{(\ell,v+t)} \geq t')$$

What does the last weak-memorylessness condition mean?

$$\mathrm{Prob}(X_{(\ell,v)} \geq t + t' \mid X_{(\ell,v)} \geq t) = \mathrm{Prob}(X_{(\ell,v+t)} \geq t')$$

- It is satisfied by CTMCs, which are memoryless: $X_{(\ell,v)} = X_\ell$

What does the last weak-memorylessness condition mean?

$$\mathrm{Prob}(X_{(\ell,v)} \geq t + t' \mid X_{(\ell,v)} \geq t) = \mathrm{Prob}(X_{(\ell,v+t)} \geq t')$$

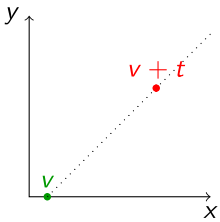
- It is satisfied by CTMCs, which are memoryless: $X_{(\ell,v)} = X_\ell$
- It is satisfied by GSMPs, (almost) by definition

What does the last weak-memorylessness condition mean?

$$\mathrm{Prob}(X_{(\ell,v)} \geq t + t' \mid X_{(\ell,v)} \geq t) = \mathrm{Prob}(X_{(\ell,v+t)} \geq t')$$

- It is satisfied by CTMCs, which are memoryless: $X_{(\ell,v)} = X_\ell$
- It is satisfied by GSMPs, (almost) by definition
- A constraint on a time "fiber":

> Distrib. from $v + t$ is that from v, under the condition that t t.u. have already elapsed.

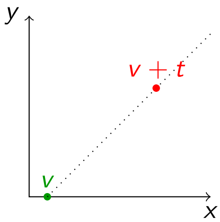


What does the last weak-memorylessness condition mean?

$$\mathrm{Prob}(X_{(\ell,v)} \geq t + t' \mid X_{(\ell,v)} \geq t) = \mathrm{Prob}(X_{(\ell,v+t)} \geq t')$$

- It is satisfied by CTMCs, which are memoryless: $X_{(\ell,v)} = X_\ell$
- It is satisfied by GSMPs, (almost) by definition
- A constraint on a time "fiber":

Distrib. from $v+t$ is that from v, under the condition that t t.u. have already elapsed.



Examples

- $\mathrm{Exp}(\lambda)$ distrib. from v implies $\mathrm{Exp}(\lambda)$ distrib. from $v+t$
- Unif distrib. from v implies Unif distrib. from $v+t$
- Any μ in v can be transferred to some μ' in $v+t$

Composing STAs: the recipe

Composing STAs: the recipe

- Define distrib. over delays as a race between the components, that is, use the law for the minimal delay

$$f_{\min}(t) = f_1(t) \cdot (1 - F_2(t)) + f_2(t) \cdot (1 - F_1(t))$$

Composing STAs: the recipe

- Define distrib. over delays as a race between the components, that is, use the law for the minimal delay

$$f_{\min}(t) = f_1(t) \cdot (1 - F_2(t)) + f_2(t) \cdot (1 - F_1(t))$$

- Compute discrete proba. as a combination of original discrete proba. and of the likelihood that the given component wins the race

$$w_i(t) = \frac{f_i(t) \cdot (1 - F_{3-i}(t))}{f_{\min}(t)}$$

Composing STAs: the recipe

- Define distrib. over delays as a race between the components, that is, use the law for the minimal delay

$$f_{\min}(t) = f_1(t) \cdot (1 - F_2(t)) + f_2(t) \cdot (1 - F_1(t))$$

- Compute discrete proba. as a combination of original discrete proba. and of the likelihood that the given component wins the race

$$w_i(t) = \frac{f_i(t) \cdot (1 - F_{3-i}(t))}{f_{\min}(t)}$$

Theorem

When STAs are weak-memoryless and almost-surely non-Zeno, the above recipe defines an *internal* parallel composition operator such that:

$$\mathbb{P}_{\mathcal{A}_1 \| \mathcal{A}_2}(\varphi_1 \wedge \varphi_2) = \mathbb{P}_{\mathcal{A}_1}(\varphi_1) \cdot \mathbb{P}_{\mathcal{A}_2}(\varphi_2)$$

Further nice and useful properties

Bisimulation (inspired by [DP03])

An extension of that for CTMCs can be defined: $(\ell, v) \equiv (\ell', v')$ if and only if for every (measurable and reasonable) $\equiv$-closed set C, for every measurable set of delays I,

$$\mathbb{P}\big((\ell, v) \xrightarrow{I,E} C\big) = \mathbb{P}\big((\ell', v') \xrightarrow{I,E} C\big)$$

[DP03] Desharnais, Panangaden. Continuous stochastic logic characterizes bisimulation of continuous-time Markov processes *(Journal of Logic and Algebraic Programming)*.

Further nice and useful properties

Bisimulation (inspired by [DP03])

An extension of that for CTMCs can be defined: $(\ell, v) \equiv (\ell', v')$ if and only if for every (measurable and reasonable) $\equiv$-closed set C, for every measurable set of delays I,

$$\mathbb{P}\big((\ell, v) \xrightarrow{I,E} C\big) = \mathbb{P}\big((\ell', v') \xrightarrow{I,E} C\big)$$

Theorem

The bisimulation is a congruence w.r.t. parallel composition:

$$\mathcal{A}_1 \equiv \mathcal{A}_2 \text{ implies } \mathcal{A}_1 \parallel \mathcal{B} \equiv \mathcal{A}_2 \parallel \mathcal{B}$$

[DP03] Desharnais, Panangaden. Continuous stochastic logic characterizes bisimulation of continuous-time Markov processes *(Journal of Logic and Algebraic Programming)*.

Outline

Challenges

- Algorithmics:

Challenges

- Algorithmics:
  - More approximation algorithms are to come

Challenges

- Algorithmics:
  - More approximation algorithms are to come
  - Almost-sure model-checking of unfair STAs by analyzing non-forgetful cycles

Challenges

- Algorithmics:
  - More approximation algorithms are to come
  - Almost-sure model-checking of unfair STAs by analyzing non-forgetful cycles
- Componentwise modelling:

Challenges

- Algorithmics:
  - More approximation algorithms are to come
  - Almost-sure model-checking of unfair STAs by analyzing non-forgetful cycles
- Componentwise modelling:
  - Add synchronization to the composition operator while preserving the nice properties (e.g. congruence)

Challenges

- Algorithmics:
  - More approximation algorithms are to come
  - Almost-sure model-checking of unfair STAs by analyzing non-forgetful cycles
- Componentwise modelling:
  - Add synchronization to the composition operator while preserving the nice properties (e.g. congruence)
  - Models with non-determinism studied so far not amenable to composition [BF09,BS12]

[BF09] Bouyer, Forejt. Reachability in stochastic timed games *(ICALP'09)*.
[BS12] Bertrand, Schewe. Playing optimally on timed automata with random delays *(FORMATS'12)*.

Challenges

- Algorithmics:
  - More approximation algorithms are to come
  - Almost-sure model-checking of unfair STAs by analyzing non-forgetful cycles
- Componentwise modelling:
  - Add synchronization to the composition operator while preserving the nice properties (e.g. congruence)
  - Models with non-determinism studied so far not amenable to composition [BF09,BS12]
  - Use interaction, as done in interactive Markov chains [Her02,HK09]

[BF09] Bouyer, Forejt. Reachability in stochastic timed games *(ICALP'09)*.
[BS12] Bertrand, Schewe. Playing optimally on timed automata with random delays *(FORMATS'12)*.
[Her02] Hermanns. Interactive Markov chains: The quest for quantified quality *(LNCS 2428)*.
[HK09] Hermanns, Katoen. The how and why of interactive Markov chains *(FMCO'09)*.

Challenges

- Algorithmics:
  - More approximation algorithms are to come
  - Almost-sure model-checking of unfair STAs by analyzing non-forgetful cycles
- Componentwise modelling:
  - Add synchronization to the composition operator while preserving the nice properties (e.g. congruence)
  - Models with non-determinism studied so far not amenable to composition [BF09,BS12]
  - Use interaction, as done in interactive Markov chains [Her02,HK09]
- Compositional verification

[BF09] Bouyer, Forejt. Reachability in stochastic timed games *(ICALP'09)*.
[BS12] Bertrand, Schewe. Playing optimally on timed automata with random delays *(FORMATS'12)*.
[Her02] Hermanns. Interactive Markov chains: The quest for quantified quality *(LNCS 2428)*.
[HK09] Hermanns, Katoen. The how and why of interactive Markov chains *(FMCO'09)*.