Hybrid Automata Stochastic Logic

an expressive language for statistical verification of stochastic models

Paolo Ballarini

Ecole Centrale de Paris, MAS Lab

joint work with: H. Djafri, S. Haddad (LSV-ENS Cachan), M. Duflot, N. Pekergin (LACL-UPEC)
objectives

- design of statistical model checking method which:
  - addresses a large class of stochastic models (larger than Markovian)
  - employs an expressive verification language

- interest:
  - combining model-checking with performance evaluation
objectives

- design of statistical model checking method which:
  - addresses a large class of stochastic models (larger than Markovian)
  - employs an expressive verification language

- interest:
  - combining model-checking with performance evaluation

- outcome
  - addressed models: Discrete Event Stochastic Processes (DESP)
  - new logic: Hybrid Automata Stochastic Logic (HASL)
    - sophisticated temporal reasoning blended with reward-based analysis
Outline

1. Overview on Stochastic Logics
2. Hybrid Automata Stochastic Logic
3. COSMOS: software support for HASL
Stochastic Model Checking

SYSTEM

STOCHASTIC MODEL
(eg CTMC)

PROPERTY

FORMULA
(stochastic logic)

MODEL CHECKER
(numerical/statistical)

YES/NO
(probability measure)

Hybrid Automata Stochastic Logic

Ballerini, Djafrì, Duflot, Haddad, Pekergin
Stochastic Model Checking

Hybrid Automata Stochastic Logic

Ballarini, Djafri, Duflot, Haddad, Pekergin
Stochastic Model Checking

- **SYSTEM**
  - STOCHASTIC MODEL (eg CTMC)

- **PROPERTY**
  - FORMULA (stochastic logic)

- **MODEL CHECKER**
  - (numerical/statistical)

- $P = ? (\Phi)$
- approximation problem
- probability measure
Numerical vs Statistical

- **Numerical model checking**: uses numerical methods for approximating probability measures (i.e. transient analysis, steady-state analysis), the approximation is in form of truncation error
  - + : better when very accurate approximation is required
  - - : state-space explosion problem
  - - : many realistic models are numerically untreatable

- **Statistical model checking**: uses discrete-event simulation to sampling model’s executions and to produce an estimation of the measure of interest
  - + : small memory requirements (no need to store the state-space)
  - + : only option for verification of many realistic models
  - - : very accurate approximations may be computationally harder
Stochastic Model Checking

- SYSTEM
- PROPERTY

STOCHASTIC MODEL

FORMULA
(stochastic logic)

MODEL CHECKER
(numerical/statistical)

YES/NO
(probability measure)

Hybrid Automata Stochastic Logic  Ballarini, Djafri, Duflot, Haddad, Pekergin
Discrete Event Stochastic Process

an abstraction whereby a real system is represented in terms of:

- **states**: enumerable set of states $S = \{s_0, \ldots, s_n\}$,
  - for $N$-dimensional model $\Rightarrow s_i = (y^i_1, y^i_2, \ldots y^i_N)$
  - state labels: conditions referring to state variables
    
    ▶ $WORK \equiv (y_1 < 3) \land (y_2 \geq 5)$  
    ▶ $DEGR \equiv (y_1 \geq 4) \land (y_2 \geq 5)$
Discrete Event Stochastic Process

an abstraction whereby a real system is represented in terms of:

- **states**: enumerable set of states $S = \{s_0, \ldots, s_n \ldots\}$,
  - for $N$-dimensional model $\Rightarrow s_i = (y_1^i, y_2^i, \ldots y_N^i)$
  - state labels: conditions referring to state variables
    
    $\text{WORK} \equiv (y_1 < 3) \land (y_2 \geq 5)$
    
    $\text{DEGR} \equiv (y_1 \geq 4) \land (y_2 \geq 5)$

- **events (actions)**: finite set of Actions $\text{Act} = \{a, b, c \ldots\}$;
  - occurrence time is driven by a probability distribution
    - generic DESP: no restriction for prob. distribution
    - Markov chain: only exponentially distributed events
an abstraction whereby a real system is represented in terms of:

- **states**: enumerable set of states $S = \{s_0, \ldots, s_n \ldots\}$,
  - for $N$-dimensional model $\Rightarrow s_i = (y_1^i, y_2^i, \ldots y_N^i)$
  - state labels: conditions referring to state variables
    
    $WORK \equiv (y_1 < 3) \land (y_2 \geq 5)$  
    $DEGR \equiv (y_1 \geq 4) \land (y_2 \geq 5)$

- **events (actions)**: finite set of Actions $Act = \{a, b, c \ldots\}$;
  - occurrence time is driven by a probability distribution
    
    - generic DESP: no restriction for prob. distribution
    - Markov chain: only exponentially distributed events

- **path**: a (possibly infinite) sequence of events occurrences

Hybrid Automata Stochastic Logic

Ballarini, Djafri, Duflot, Haddad, Pekergin
Stochastic Model Checking

- **SYSTEM**
- **PROPERTY**
- **STOCHASTIC MODEL**
- **FORMULA** (stochastic logic)
- **MODEL CHECKER** (numerical/statistical)
- **YES/NO** (probability measure)

Hybrid Automata Stochastic Logic

Ballerini, Djafri, Duflot, Haddad, Pekergin
Stochastic Logic

**basic idea:** to extend reasoning of classical temporal logic (LTL/CTL) to **stochastic models**

- **syntax based:** properties are expressed in terms of a formula
  - **Continuous Stochastic Logic (CSL);** [Aziz 2000]
  - **Continuous Stochastic Reward Logic (CSRL);** [Baier et al. 2000]

**important points about above logics:**
- **limited to CTMCs models**
- **designed for application of numerical methods for CTMC analysis**
- **solution algorithms are affected by state-space explosion**
Stochastic Logic

**basic idea:** to extend reasoning of classical temporal logic (LTL/CTL) to stochastic models

▶ **syntax based:** properties are expressed in terms of a formula

  ▶ Continuous Stochastic Logic (**CSL**); [Aziz 2000]
  ▶ Continuous Stochastic Reward Logic (**CSRL**); [Baier *et al.* 2000]

▶ **regular expressions:** properties are expressed in terms of a regular expression

  ▶ action-state Continuous Stochastic Logic (**asCSL**); [Baier *et al.* 2004]
Stochastic Logic

**basic idea:** to extend reasoning of classical temporal logic (LTL/CTL) to **stochastic models**

- **syntax based:** properties are expressed in terms of a formula
  - **Continuous Stochastic Logic (CSL);** [Aziz 2000]
  - **Continuous Stochastic Reward Logic (CSRL);** [Baier et al. 2000]

- **regular expressions:** properties are expressed in terms of a regular expression
  - **action-state Continuous Stochastic Logic (asCSL);** [Baier et al. 2004]

- **automata-based:** properties are expressed in terms of an automaton
  - **Continuous Stochastic Logic - (1-clock) Timed Automata (CSL\textsuperscript{TA});** [Haddad et al. 2007]
  - **Continuous Stochastic Logic - (n-clocks) Timed Automata;** [Katoen et al. 2009]
Stochastic Logic

**basic idea:** to extend reasoning of classical temporal logic (LTL/CTL) to stochastic models

- **syntax based:** properties are expressed in terms of a formula
  - Continuous Stochastic Logic (**CSL**); [Aziz 2000]
  - Continuous Stochastic Reward Logic (**CSRL**); [Baier et al. 2000]

- **regular expressions:** properties are expressed in terms of a regular expression
  - action-state Continuous Stochastic Logic (**asCSL**); [Baier et al. 2004]

- **automata-based:** properties are expressed in terms of an automaton
  - Continuous Stochastic Logic - (1-clock) Timed Automata (**CSL^TA**); [Haddad et al. 2007]
  - Continuous Stochastic Logic - (n-clocks) Timed Automata; [Katoen et al. 2009]

- **important points about above logics:**
  - limited to CTMCs models
  - designed for application of numerical methods for CTMC analysis
    - solution algorithms are affected by state-space explosion
CSL (Continuous Stochastic Logic)

- CSL: a language to state properties wrt to Continuous Time Markov Chain models

\[ \phi := a | \neg \phi | \phi \land \phi | S_{\sim p}(\phi) | P_{\sim p}(\varphi) \]

\[ \varphi := X^l \phi | \phi U^l \phi \]

- \( a \in AP \) (Atomic Prop.), \( \sim \in \{<, \leq, \geq, >, =, \neq \} \), \( l = [\alpha, \beta] \subseteq \mathbb{R}_{\geq 0} \) (time-bound)

- CSL is branching-time: each temporal operator must be preceded by a probability bound \( \sim p \)

\[ \phi \equiv P_{\leq 0.9}(WORK U^{[\alpha', \beta']} [P_{< 0.5}(DEGR U^{[\alpha'', \beta'']} FAIL)]) \]

CSL reasoning
Paths probability measure (stochastic models)

Probability Measure of a finite-path

\[ Pr(s_0, s_1, \ldots, s_N) = p_1 \cdot p_2 \cdot p_3 \cdot \ldots \cdot p_N \]

Probability Measure of a UNTIL-path

\[ \phi \equiv (c U e) \]

\[ Pr(s_0, (c U e)) = p_1 \cdot p_2 \]

Discretely quantified Path-formulae (CTL) vs Probabilistically quantified Path-formulae (PCTL/CSL)

\[ AG(\phi) \equiv P_{\geq 1}(\phi) \]
\[ EF(\phi) \equiv P_{> 0}(\phi) \]
Stochastic Temporal reasoning

in general a logic formula \( \varphi \) shall allow to **reason about** any/all of the following:

- **state labels**: \( L(s_i) \)
- **transition labels** (actions): \( a_i \)
- **transition durations**: \( \tau_i \)
- **state/transition rewards** (if supported)

what type of properties can we characterize through stochastic temporal logics?

- **reachability** (**CSL**)
- **sequential reachability** (**asCSL**)
- **multiple-bounded sequential reachability** (1-clock automata - **CSL\(^{1TA}\)**)
- **conjunction of multiple-bounded sequential reachability** (n-clocks automata **CSL\(^{nTA}\)**)
CSL- reachability

- reasoning about state-labels + single-time-bound

\[
\sigma \models [(y_1 \geq n_a) \cup_{T \in [\alpha, \beta]} (y_1 < n_a)]
\]

⇒ “reaching a FAIL ≡ (y_1 < n_a) state at time point \( t \in [\alpha, \beta] \) remaining in WORK ≡ (y_1 \geq n_a) states until that point”
asCSL - sequential reachability

- reasoning about state/action labels + single-time-bound

\[ \sigma \models \left( (n_a \leq y_1 \leq n_b), \text{Act} \right)^*; (n_b \leq y_1 \leq n_c), \text{Act} \right)^*; (y_1 \leq n_a, \sqrt{\cdot}) T \in [\alpha, \beta] \]

⇒ “reach a FAIL state within \( T \in [\alpha, \beta] \) passing through a sequence of WORK states followed by DEGR states"
CSL$^{1TA}$: multiple-bound sequential reachability

- reasoning about state/action labels + multiple-time-bounds

$\Rightarrow$ “pass from WORK states to DEGR within $[\alpha', \beta']$ time units and then from DEGR to FAIL states within $[\alpha'', \beta'']$ time units"
**CSL\textsuperscript{nTA}: complex multiple-bound sequential reachability**

- reasoning about state/action labels + several independent-time-bounds

![Diagram of CSN TA with state transitions and acceptance condition](image-url)
### Taxonomy of stochastic logics

<table>
<thead>
<tr>
<th></th>
<th>CSL</th>
<th>CSRL</th>
<th>asCSL</th>
<th>CSL(^{1\text{TA}})</th>
<th>CSL(^{n\text{TA}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>formalism</td>
<td>syntax</td>
<td>syntax</td>
<td>reg. expr. on action/state</td>
<td>1-clock timed automata</td>
<td>n-clock timed automata</td>
</tr>
<tr>
<td>reachability</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>reward-bounded reachability</td>
<td>NO</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>sequential reachability</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>multiple-bounded sequential reachability</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>nested-UNTIL and conjunction of multiple sequential reachability</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>rewards</td>
<td>NO</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>type of model</td>
<td>CTMC</td>
<td>CTMC + state reward</td>
<td>CTMC</td>
<td>CTMC</td>
<td>CTMC</td>
</tr>
<tr>
<td>numerical solution</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>statistical solution</td>
<td>YES(^1)</td>
<td>YES</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

\(^1\) only on sub-logic with no-nested path-operators
Outline

1. Overview on Stochastic Logics

2. Hybrid Automata Stochastic Logic

3. COSMOS: software support for HASL
Hybrid Automata Stochastic Logic

- a formalism for assessing measures of a stochastic model
  - it addresses a large class of models: Discrete Event Stochastic Processes
  - the “property” to be verified consists of two elements:
    - a (Linear) Hybrid Automata for selecting a model's paths
    - an expression indicating the quantity to be measured
  - it is based on confidence-interval (statistical simulation)

- outcome: expressive verification language whereby sophisticated (linear-time) temporal reasoning is naturally blended with reward-based analysis
the model: Discrete Event Stochastic Processes

\[ \mathcal{D} = (S, \pi_0, E, \text{Ind}, \text{enabled}, \text{target}, \text{delay}, \text{choice}) \]

- **S**: set of states
- **\(\pi_0\)**: initial distribution
- **E**: set of events
- **Ind**: indicator function \((S \rightarrow \mathbb{R})\)
- **enabled**: enabled events \((S \rightarrow 2^E)\)
- **target**: successor function \((S \times E \rightarrow S)\)
- **delay**: delay function \((S \times E \rightarrow \text{dist}(\mathbb{R}^+))\)
- **choice**: choice policy \((S \times 2^E \rightarrow \text{dist}(E))\)

### DEQP model of a shared memory system in GSPN form

**Remarks:**

- **delay**: no restrictions on allowed distributions (discrete distributions included)
- **choice**: necessary to disambiguate simultaneous events
Synchronizing Linear Hybrid Automata

role: selector of DESP paths \[ A = \langle E, L, \Lambda, \text{Init}, \text{Final}, X, \text{flow}, \rightarrow \rangle \]

- \( E \): events
- \( L \): locations
- \( X \): real-valued vars

- \( \Lambda \): location labels (refer to DESP state)
- \( \text{flow}(x_i) \): real-valued function of DESP state
- \( \rightarrow \): edge labels \((l, A, u \leftarrow l')\)

\[ \begin{align*}
\dot{x}_1 & : 0 \\
\dot{x}_T & : \text{global time} \\
\end{align*} \]

\[ \begin{align*}
x_T & \leq \alpha \\
\dot{x}_1 & : \text{Access}_2 > 0 \\
\dot{x}_1 & : \text{req}_2 \\
\end{align*} \]

\[ \begin{align*}
\dot{x}_1 & : \text{Access}_1 > 0 \\
\dot{x}_T & : \text{global time} \\
\end{align*} \]

\[ \begin{align*}
x_T & = \alpha \land x_1 > 0, x_{\text{OK}} = 1 \\
\end{align*} \]

\( \text{A: Linear Hybrid Automaton} \)

Ballarini, Djafri, Duflot, Haddad, Pekergin
Synchronizing Linear Hybrid Automata

role: selector of DESP paths $\Rightarrow A = \langle E, L, \Lambda, \text{Init}, \text{Final}, X, \text{flow}, \rightarrow \rangle$

- $E$: events
- $L$: locations
- $X$: real-valued vars
- $\Lambda$: location labels (refer to DESP state)
- $\text{flow}(x_i)$: real-valued function of DESP state
- $\rightarrow$: edge labels ($l \xrightarrow{\gamma,A,u} l'$)

$D$: DESP

HASL verification scheme (intuitively):
- a timed-path $\sigma$ of $D$ is generated through simulation
- $A$ decides whether $\sigma$ is accepted or not, meanwhile collecting data about $\sigma$
- collected data is used for estimating relevant measures (wrt accepted paths)

$\phi_A$: accept all paths such that up to time $\alpha$ the memory has been occupied longer by Q1 users than Q2's
the quantity to be measured ($Z$)

$Z ::= E[Y] \mid Z + Z \mid Z \times Z$

$Y ::= c \mid Y + Y \mid Y \times Y \mid Y/Y \mid last(y) \mid min(y) \mid max(y) \mid int(y) \mid avg(y)$

$y ::= c \mid x \mid y + y \mid y \times y \mid y/y$

$\sigma \equiv \text{last}(x_i)$ the last value of $x_i$ along $\sigma$

$\sigma \equiv \text{min}(x_i)$ the minimal value of $x_i$ along $\sigma$

$\sigma \equiv \text{avg}(x_i)$ the average value of $x_i$ along $\sigma$

$\sigma \equiv \text{corel}(x_i, x_j)$ the correlation between the value of $x_i$ and $x_j$ along $\sigma$

remark: path-variables are evaluated on-the-fly on generation of a $(\mathcal{D} \times \mathcal{A})$-path

what $Z$ represents? $\Rightarrow$ the conditional “moments" of $Y$

conditional expectation: $E[\text{last}(x_1 \mid x_1 > 0, x_t \leq 10)]$

conditional variance: $\text{Var}[\text{max}(x_1 - x_2 \mid x_2 \geq k, x_t \leq 10)]$

remark: probabilistic model-checking corresponds to $Z \equiv E[\text{last}(x_{\text{OK}}) \mid \phi_{\mathcal{A}}]$
Example 1: utilization difference measures

**accepting condition:** at time $T = \alpha$, the $P_1$-processes have used the resource longer than the $P_2$-processes

- $x_1$ : timer, it measures difference between occupation time by $P_1$ and $P_2$
- $Z \equiv E[\text{last}(x_{\text{OK}})]$ : probability that the shared resource is used longer by $P_1$-processes than by $P_2$'s.
- $Z \equiv E[\text{last}(x_1)]$ : expected value of the difference between the utilization time of $P_1$ and $P_2$ processes
- $Z \equiv E[\text{max}(x_1)]$ : expected value of the maximum of such difference
Some experiments with COSMOS

assessing resource occupation based measures as a function of the arrival-rate $\lambda_2$ (arrival of $P_2$-processes)

\begin{itemize}
  \item $\Phi_1 \equiv E[success \mid (x_1 > 0) \land (x_0 \in [\alpha, \alpha])]$
  \item $\Phi_2 \equiv E[max(x_1) \mid (x_1 > 0) \land (x_0 \in [\alpha, \alpha])]$
  \item $\Phi_3 \equiv E[avg(x_1) \mid (x_1 > 0) \land (x_0 \in [\alpha, \alpha])]$
\end{itemize}

hybrid automata stochastic logic
Example 2: average waiting time til \( k \) departures

- \( x_1 : \text{\( P_1 \)-processes cumulative waiting time (state-reward variable) }\)
- \( x_2 : \text{number of \( P_1 \)-processes that have used the resource (transition-reward variable) }\)
- \( \text{last}(x_1/x_2) : \text{(Sup of the) average waiting time (until \( k \) \( P_1 \)-departures) }\)
# Taxonomy of stochastic logics

<table>
<thead>
<tr>
<th></th>
<th>CSL</th>
<th>CSRL</th>
<th>asCSL</th>
<th>CSL^{1TA}</th>
<th>CSL^{nTA}</th>
<th>HASL</th>
</tr>
</thead>
<tbody>
<tr>
<td>formalism</td>
<td>syntax</td>
<td>syntax</td>
<td>reg. expr. on action/state</td>
<td>1-clock DTA</td>
<td>n-clocks DTA</td>
<td>LHA</td>
</tr>
<tr>
<td>reachability</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>reward-bounded</td>
<td>NO</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>reachability</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sequential reachability</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>multiple-bounded</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>sequential reachability</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>nested-UNTIL and</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>conjunction of multiple sequential reachability</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rewards</td>
<td>NO</td>
<td>state-only</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>type of model</td>
<td>CTMC</td>
<td>CTMC +</td>
<td>CTMC</td>
<td>CTMC</td>
<td>CTMC</td>
<td>DESP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>state reward</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>numerical solution</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>statistical solution</td>
<td>YES^2</td>
<td>YES</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>YES</td>
</tr>
</tbody>
</table>

^2 only on sub-logic with no-nested path-operators

---

Ballarini, Djafri, Duflot, Haddad, Pekergin
Outline

1. Overview on Stochastic Logics
2. Hybrid Automata Stochastic Logic
3. COSMOS: software support for HASL
COSMOS: a software tool for HASL verification

- **COSMOS**: *Concepts et Outils Statistiques pour les MOdèles Stochastiques*
  - implemented in C++ (*BOOST* libraries)
  - inputs: DESP (GSPN) + LHA + Expression (textual files)
  - it features a command-line interpreter

- efficient code generation (model compilation architecture)

- two versions available:
  - **COSMOS-G++**: compiled with G++
  - **COSMOS-LLVM**: optimized compilation through LLVM Compiler Infrastructure
assessing COSMOS

- comparison with PRISM4.0 statistical engines
  - PRISM-CI: (Confidence Interval) ⇒ any measure
  - PRISM-ACI: (Asymptotic Confidence Interval) ⇒ any measure
  - PRISM-APMC: (Approx. Prob. Model Checking) ⇒ prob. measures only

- we considered 2 benchmarks CTMC models
  - tandem queueing network (TQN) ⇒ a measure of probability
  - cyclic server polling system (CSPS) ⇒ a measure of time

- we compare PRISM-stat solver against COSMOS in the following respects:
  - accuracy: average distance of estimated value from value obtained with PRISM-num solver
  - run-time: time employed to produce output
measure: probability of first queue to be full within T
average runtime comparison
- **COS-LLVM**: \( \sim \) **PRISM-ACI, PRISM-CI**
- **COS-G++**: \( \sim \) 3x slower than **PRISM-CI**, **PRISM-ACI**

measure: average waiting time of Queue 1 within T
average runtime comparison
- **COS-LLVM**: \( \sim \) 11x faster than **PRISM-ACI**
- **COS-LLVM**: \( \sim \) 7x faster than **PRISM-CI**
- **COS-G++**: \( \sim \) 2.8x faster than **PRISM-ACI**
- **COS-G++**: \( \sim \) 1.8x faster than **PRISM-CI**
ongoing/future work

▶ conceptual aspects:
  ▶ **rare-events**: assessing measures related to rare-events may require extremely large simulation time (ongoing)

  ▶ **time-unbounded formulae**: (open problem)

▶ practical aspects:
  ▶ **testing**: more needed
  ▶ **applications**:
    ▶ Flexible Manufacturing Systems (ongoing)
    ▶ Systems Biology

  ▶ **COSMOS GUI**: development (to be done)
BIBLIOGRAPHY


in preparation


_COSMOS web-page: http://www.lsv.ens-cachan.fr/~djafri/cosmos/  (under construction)_