

# Hybrid Automata Stochastic Logic: expressive statistical verification of stochastic models

Paolo Ballarini

INRIA, Bretagne Atlantique

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

# what is this presentation about?

- we introduce a new methodology for the **automatic verification of stochastic models**
- principal features are:
  - highly **expressive formalism** which allows for capturing **sophisticated dynamics/measures** of a model
  - making **stochastic model checking** leaning towards **performance evaluation**
  - formalism uses Linear Hybrid Automata as a means to express the properties/measures of interest
  - the **verification/estimation** procedure is **statistical** (simulation based), thus not affected by state-space-explosion problem

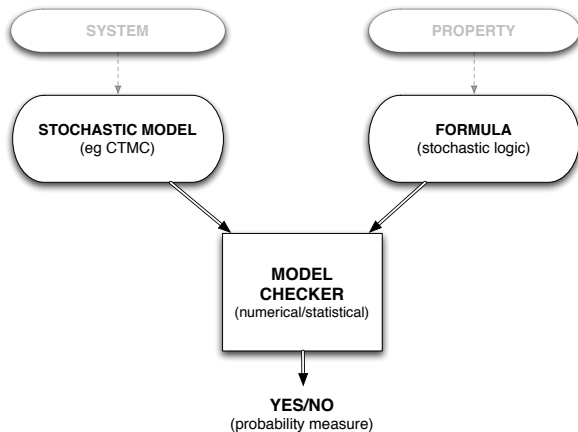
# Outline

Stochastic Logics

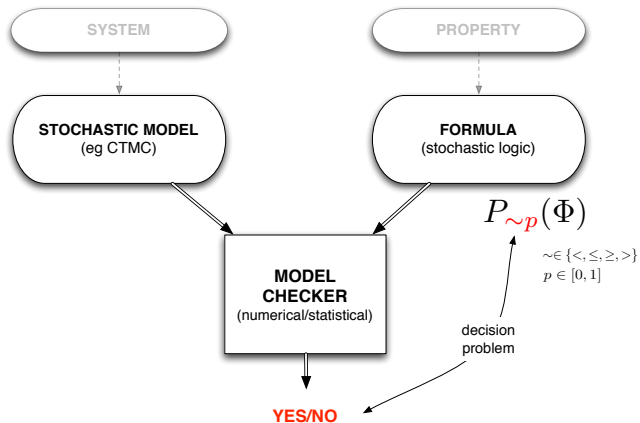
Hybrid Automata Stochastic Logic

COSMOS: a statistical model checker for HASL

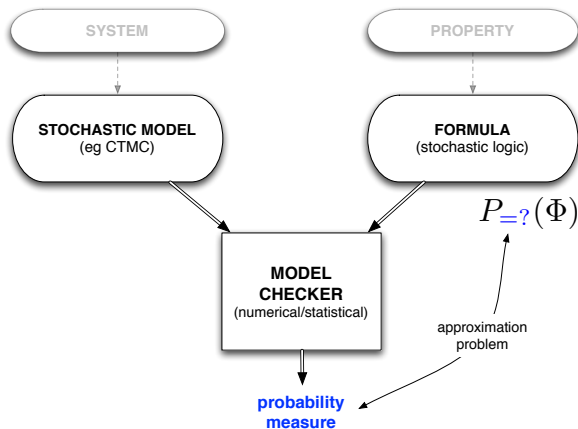
# Stochastic Model Checking



# Stochastic Model Checking



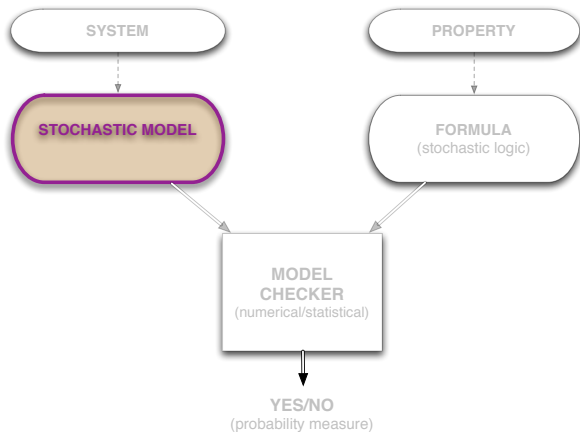
# Stochastic Model Checking



# 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

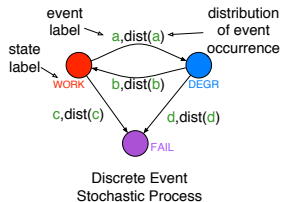
# Stochastic Model Checking



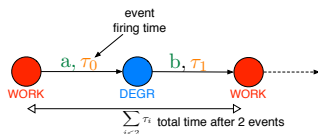
# Discrete Event Stochastic Process

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

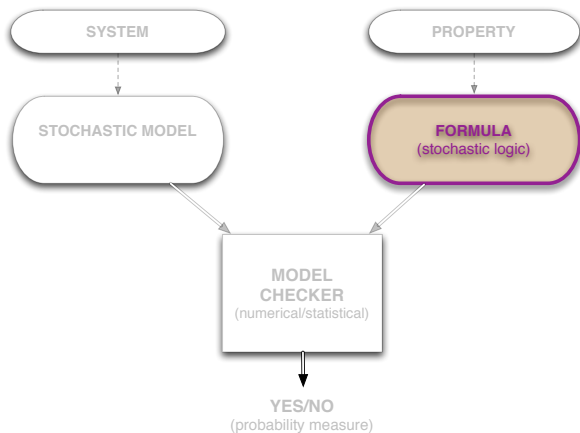
- **states:** enumerable set of states  $S = \{s_0, \dots, s_n \dots\}$ ,
  - for  $N$ -dimensional model  $\Rightarrow s_i = (y_1^i, y_2^i, \dots, y_N^i)$
  - state labels: conditions referring to state variables (e.g.)  
 $WORK \equiv (y_1 < 3) \wedge (y_2 \geq 5)$        $DEGR \equiv (y_1 \geq 4) \wedge (y_2 \geq 5)$
- **events (actions):** finite set of Actions  $Act = \{a, b, c \dots\}$ ;
  - **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



# Stochastic Model Checking



# 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<sup>TA</sup>); [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<sup>TA</sup>); [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 branching time logic

$$\phi := a \mid \neg\phi \mid \phi \wedge \phi \mid \mathcal{S}_{\sim p}(\phi) \mid \mathcal{P}_{\sim p}(\varphi)$$

$$\varphi := X^I \phi \mid \phi U^I \phi$$

- $a \in AP$  (Atomic Prop.),  $\sim \in \{<, \leq, \geq, >, =, \neq\}$ ,  $I = [\alpha, \beta] \subseteq \mathbb{R}_{\geq 0}$  (time-bound)
  
- CSL is branching-time: each temporal operator must be preceded by a probability bound  $\sim p$

$$\phi \equiv \mathcal{P}_{\leq 0.9}(\mathbf{WORK} U^{[\alpha', \beta']} [\mathcal{P}_{< 0.5}(\mathbf{DEGR} U^{[\alpha'', \beta'']} \mathbf{FAIL})]) \quad \text{CSL reasoning}$$

# Stochastic Temporal reasoning

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

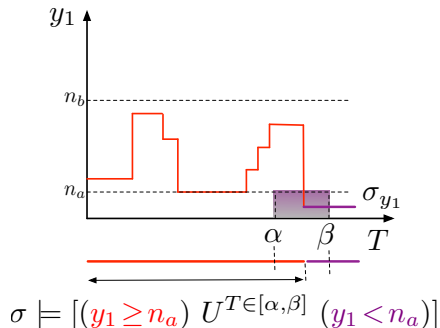
- **state labels**:  $L(s_i)$
- **transition labels** (actions):  $a_i$
- **transition durations**:  $\tau_i$
- **state/transition rewards** (if supported)

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

- **reachability** (CSL)
- **sequential reachability** (asCSL)
- **multiple-bounded sequential reachability** (1-clock automata - CSL<sup>TA</sup>)
- **conjunction of multiple-bounded sequential reachability** (n-clocks automata CSL<sup>n-TA</sup>)

# CSL- reachability

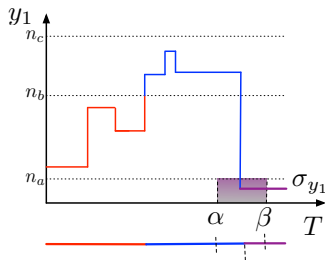
- reasoning about **state-labels** + **single-time-bound**



$\Rightarrow$  “reaching a **FAIL**  $\equiv (y_1 < n_a)$  state at time point  $t \in [\alpha, \beta]$  remaining in **WORK**  $\equiv (y_1 \geq n_a)$  states until that point”

# asCSL - sequential reachability

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

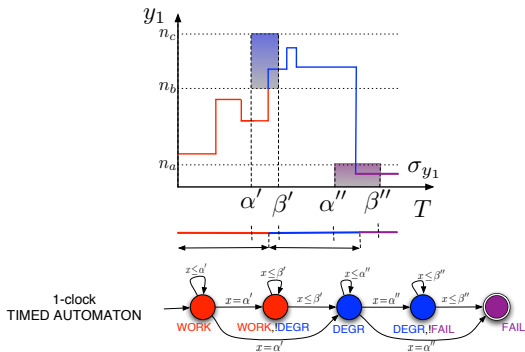


$$\sigma \models ((n_a \leq y_1 \leq n_b), Act)^*; ((n_b \leq y_1 \leq n_c), Act)^*; (y_1 \leq n_a, \sqrt{ })^{T \in [\alpha, \beta]}$$

$\Rightarrow$  "reach a **FAIL** state within  $T \in [\alpha, \beta]$  passing through a sequence of **WORK** states followed by **DEGR** states"

# CSL<sup>TA</sup> - multiple-bounded sequential reachability

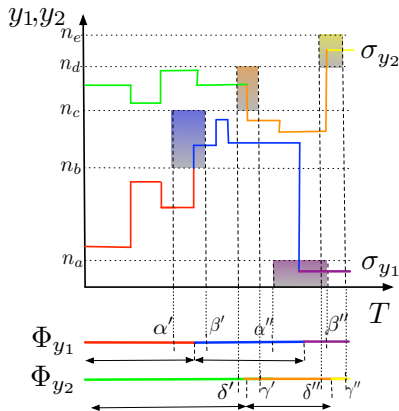
- reasoning about **state/action labels** + **multiple-time-bounds**



$\Rightarrow$  "pass from **WORK** states to **DEGR** within  $[\alpha', \beta']$  time units and then from **DEGR** to **FAIL** states within  $[\alpha'', \beta'']$  time units"

# CSL<sup>n-TA</sup> - conjunction of multiple-bounded sequential reachability

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



Acceptance condition:  $\Phi_{y_1} \wedge \Phi_{y_2}$



n-clocks  
TIMED AUTOMATON

# Reward-based reasoning

- **Rewards:** real-valued functions of a model
  - **state-rewards:**  $\rho : S \rightarrow \mathbb{R}^+$
  - **transition-rewards:**  $\iota : S \times S \rightarrow \mathbb{R}^+$
- **reward-based reasoning:** temporal-reasoning + conditions on the **reward accumulated**
- **CSRL [Baier et al. 2000]: CTMC + state-reward structure** ( $\rho : S \rightarrow \mathbb{R}^+$ )
  - CSL path-operators with time-bounds + reward-bounds
    - $(\Phi U_J^I \Psi)$  : reach a  $\Psi$ -state (through  $\Phi$ -states) within  $t \in I$  and so that average reward cumulated spending time in  $\Phi$ -states is in  $J$ .
  - $\rightarrow$  reward-analysis based on CTMC transient-analysis and steady-state analysis (thus affected by state-space explosion)

# Taxonomy of stochastic logics

	<b>CSL</b>	<b>CSRL</b>	<b>asCSL</b>	<b>CSL<sup>TA</sup></b>	<b>CSL<sup>n-TA</sup></b>
formalism	syntax	syntax	reg. expr. on action/state	1-clock timed automata	N-clock timed automata
reachability	YES	YES	YES	YES	YES
reward-bounded reachability	NO	YES	NO	NO	NO
sequential reachability	NO	NO	YES	YES	YES
multiple-bounded sequential reachability	NO	NO	NO	YES	YES
nested-UNTIL and conjunction of multiple sequential reachability	NO	NO	NO	NO	YES
rewards	NO	YES	NO	NO	NO
type of model	CTMC	CTMC + state reward	CTMC	CTMC	CTMC
numerical solution	YES	YES	YES	YES	YES
statistical solution	YES <sup>1</sup>	YES	N/A	N/A	N/A

---

<sup>1</sup> only on sub-logic with no-nested path-operators

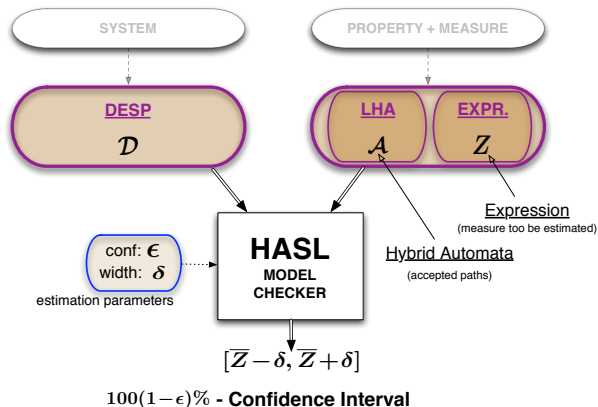
# Outline

Stochastic Logics

Hybrid Automata Stochastic Logic

COSMOS: a statistical model checker for HASL

# HASL model checking



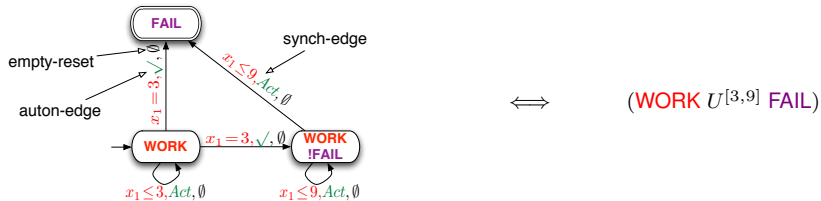
- **input:**

- **model:** a DESP  $\mathcal{D}$
- **formula:** an LHA  $\mathcal{A}$
- **expression:**  $Z$

- **output:**

- confidence interval
- estimation of expression  $Z$

# Deterministic Timed Automata



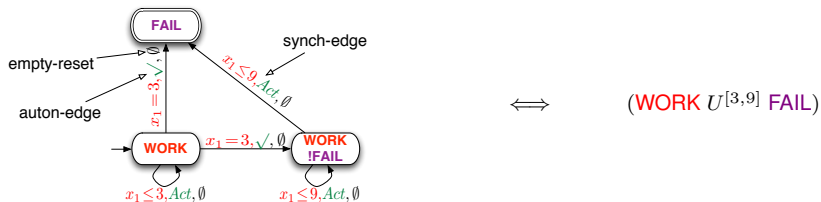
- DTA: a machine that reads in paths  $\sigma = s_0 \xrightarrow{a_0, \tau_0} s_1 \xrightarrow{a_1, \tau_1} \dots$
- **locations** labeled with state formula ;  $n$ -tuple  $X = (x_1, \dots, x_n)$  of *clock variables*
- **transitions**:  $l \xrightarrow{\gamma, A, r} l'$ 
  - $\gamma$  : a **clocks' constraint** e.g.  $\bigwedge_i x_i \leq c_i$
  - $A$  : a **set of actions** or  $\surd$
  - a (possibly empty) set of **clock resets**

autonomous edge  $\Leftrightarrow A = \surd$

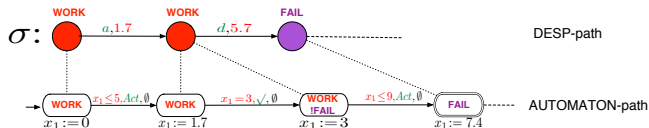
synchronizing edge  $\Leftrightarrow A \neq \surd$

auton-edges have precedence on synch-edges

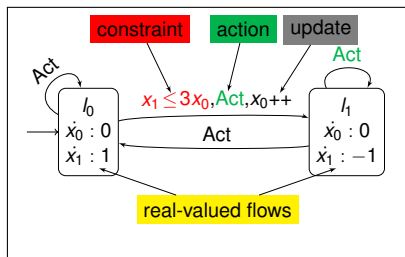
# Deterministic Timed Automata



- reading of a DESP-path through synchronization with the automaton
  - path  $\sigma$  is **accepted** if it leads to a final location of the automata
  - path  $\sigma$  is **rejected** if synchronization blocks without reaching a final location



# From Timed Automata to Hybrid Automata



- **real-valued data variables**  $X = (x_1, \dots, x_n)$  (in place of DTA's clock variables)
- **flow of variables:**
  - depends on the location
  - can be a **real-valued constant** or a **real-valued function of a DESP state**
- **transitions:**  $l \xrightarrow{\gamma, A, r} l'$

- $\gamma$  : a **constraint** -> **linear function of data-variables**

$$\bigwedge_j (\sum_{1 \leq i \leq n} \alpha_i x_i \sim c)$$

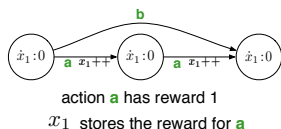
$\sim \in \{<, \leq, \geq, >\}, =$

- $u$  : **updates** -> **linear function of data-variables**

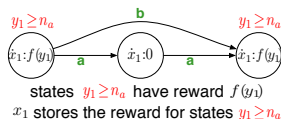
$$u_k(X) = \sum_{1 \leq i \leq n} \alpha_i x_i + c$$

# Modelling with LHA variables

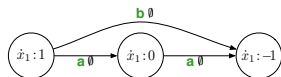
LHA variables can be used to model several things including: **timers**, **state-rewards**, **transition-rewards**



- $x_1$  : **transition-reward**
- zero-flow in every location
- update: function of synch-action (increment if event-counter)

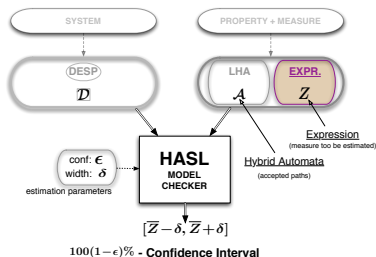


- $x_1$  : **state-reward**
- flow: function of a DESP-state-indicator



- $x_1$  : **timers**
- flow: normally in  $\{1, 0, -1\}$
- no update

# HASL Expressions



- expression  $Z$ : based on *moments of path-random-variables*  $Y$

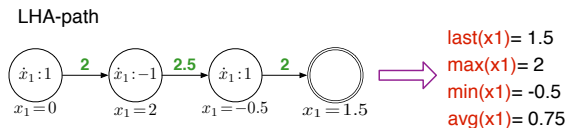
$Z ::= E(Y) \mid Z + Z \mid Z \times Z$

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

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

# LHA Path-Variables

- a path-variable  $Y$  is a random variable corresponding to a synchronizing-path of an LHA
  - $Y \equiv \text{last}(x_i)$  the last value of  $x_i$  along the synchronizing path
  - $Y \equiv \text{min}(x_i)$  the minimal value of  $x_i$  along the synchronizing path
  - $Y \equiv \text{avg}(x_i)$  the average value of  $x_i$  along the synchronizing path
  - $Y \equiv \text{var}(x_i)$  the variability of the value of  $x$  along the synchronizing path
  - $Y \equiv \text{corel}(x_i, x_j)$  the correlation between the value of  $x_i$  and  $x_j$  along the synchronizing path.

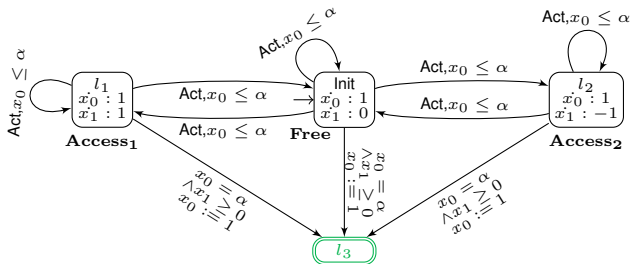


- path-variables are evaluated *on-the-fly* on generation of a  $(\mathcal{D} \times \mathcal{A})$ -path



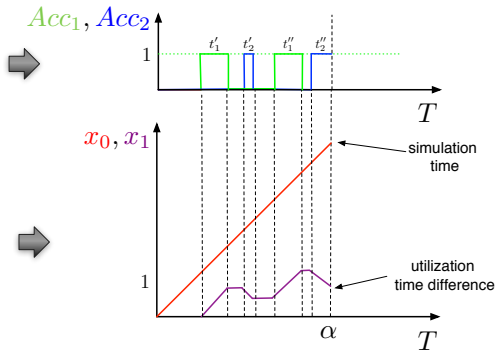
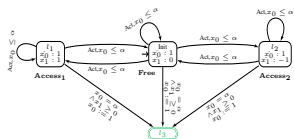
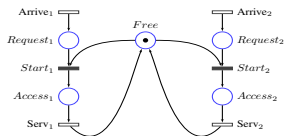
# Example 1: utilization difference measures

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



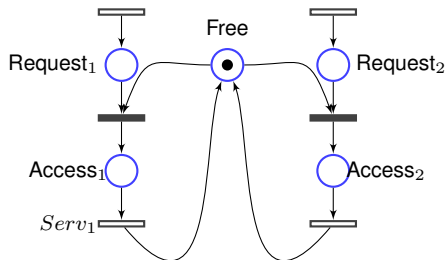
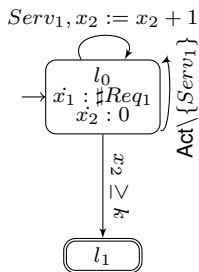
- $x_0$  (overloaded variable): global timer (before acceptance); bernoulli variable (on acceptance)
- $x_1$  : timer, difference between occupation time by  $P_1$  and  $P_2$
- $Z \equiv E[\text{last}(x_0)]$  : probability that the shared resource is used longer by  $P_1$ -processes than by  $P_2$ 's.
- $Z \equiv E[\text{last}(x_1)]$  : 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

# example



accepting condition :  $x_1 > 0 \equiv t'_1 + t''_1 > t'_2 + t''_2$

## Example 2: average waiting time til $k$ departures



- $x_1$  :  $P_1$ -processes cumulative waiting time (state-reward variable)
- $x_2$  : number of  $P_1$ -processes that have used the resource (transition-reward variable)
- $last(x_1/x_2)$  : (Sup of the) average waiting time (until  $k$   $P_1$ -departures)

# Taxonomy of stochastic logics

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

---

<sup>2</sup> only on sub-logic with no-nested path-operators

# Outline

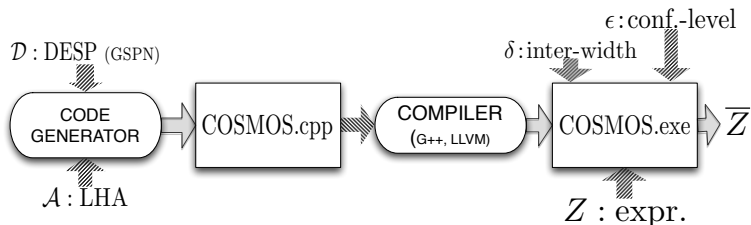
Stochastic Logics

Hybrid Automata Stochastic Logic

COSMOS: a statistical model checker for HASL

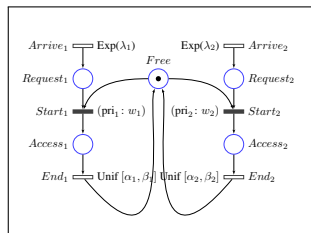
# COSMOS: a software tool for HASL analysis

- implemented in C++, uses Boost libraries for random number generation
- two versions currently available: G++ compiled, LLVM optimized compilation
- implemented according to a *model driven code generation* scheme



# Some experiments with COSMOS

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



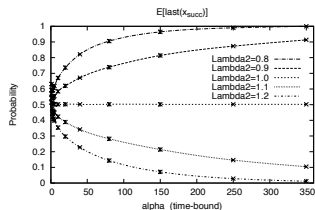
time bound  $I = [\alpha, \alpha]$ ,

$x_0$ : real-time;

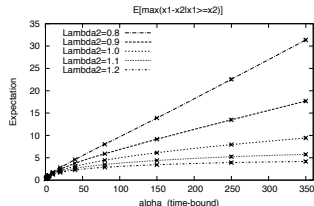
$x_1$ : occupation divergence timer;

acceptance condition:  $x_1 > 0 \wedge (x_0 \in [\alpha, 2\alpha])$

- $\Phi_1 \equiv E[\text{success} | (x_1 > 0) \wedge (x_0 \in [\alpha, \alpha])]$   
probability that *Res* used longer by  $P_1$ -processes
- $\Phi_2 \equiv E[\max(x_1) | (x_1 > 0) \wedge (x_0 \in [\alpha, \alpha])]$   
maximum occupation time divergence
- $\Phi_3 \equiv E[\text{avg}(x_1) | (x_1 > 0) \wedge (x_0 \in [\alpha, \alpha])]$   
average occupation time divergence



(a)  $\Phi_1$ : probability of  $x_1 \geq x_2$



(b)  $\Phi_2$ : maximum positive difference  
 $\max(x_1 - x_2)$

# COSMOS vs PRISM-stat

- we considered 2 benchmark CTMC models available at PRISM home page  
Tandem Queuing System (TQS) and Cyclic Server Polling System (CSPS)
  
- we assessed time-bounded properties with
  - PRISM-num (numerical engine of PRISM)
  - PRISM-stat (statistical engine of PRISM)
  - COSMOS
  
- **accuracy**: comparison of the estimates obtained with COSMOS and PRISM-stat  
(using PRISM-num as reference)
  
- **runtime**: comparison COSMOS vs PRISM-stat runtime

# COSMOS vs PRISM-stat

- estimating a measure of probability

PROBABILITY first queue is full (Tandem Queuing System, bounded queue length $C = 5$ )											
T	probability measure			generated paths			runtime (sec)			runtime-gain	
	PRISM-num	PRISM-stat	COSMOS	PRISM-stat	COSMOS	paths-gain	PRISM-stat	COS-G++	COS-LLVM	COS-G++	COS-LLVM
20	0.33574	+0.00137	+0.00010	92042	59000	+35.89%	18.64	51.61	14.63	-176.87%	+21.51%
40	0.56931	+0.00061	+0.00126	92042	64800	+29.59%	30.03	89.12	24.59	-196.76%	+18.11%
80	0.82229	-0.00351	-0.00640	92042	38800	+57.84%	42.54	73.73	20.22	-73.31%	+52.46%

- estimating a generic measure

AVERAGE WAITING-TIME of station 1 (Cyclic Server Polling System, number of stations $N = 4$ )											
T	waiting-time measure			generated paths			runtime (sec)			runtime-gain	
	PRISM-num	PRISM-stat	COSMOS	PRISM-stat	COSMOS	paths-gain	PRISM-stat	COS-G++	COS-LLVM	COS-G++	COS-LLVM
10	1.08484	-0.01778	-0.00197	92042	20400	77.83%	243.67	259.66	62.15	-6.56%	+74.49%
20	2.49446	+0.0634	0.01604	92042	13200	85.65%	434.18	302.37	73.06	+30.35%	+83.17%
50	6.73012	+0.20522	-0.00962	92042	36400	60.45%	1007.71	1937.75	467.22	-92.29%	+53.63%

# Conclusion

HASL a logic for stating complex temporal properties of stochastic models

- it unifies expressive temporal reasoning with reward based analysis (concept of **conditional expectation**)
- it naturally leans towards a **Performance Evaluation model checking** approach
- it targets DESP models (not only Markovian)
- it uses a statistical (simulation based) approach to estimate measure of interest
- it does not suffer of state-space-explosion
- software support available (prototype)

On going work

- understanding better the expressiveness of HASL
- applications: SYSTEMS BIOLOGY (to start)
- tool: code-optimization (on-going); GUI development (yet to start)
- optimization of the simulation process wrt rare-events (on-going)
- tool parallelization (COSMOS-GPU?)

# BIBLIOGRAPHY

to appear

**[BDDHP11a]:** P. Ballarini, H. Djafri, M. Dufлот, S. Haddad, N. Pekergin. "*HASL: an Expressive Language for Statistical Verification of Stochastic Models*" Proceeding 5th Int. Conference ValueTOOLS'11, to appear.

submitted

**[BDDHP11b]:** P. Ballarini, H. Djafri, M. Dufлот, S. Haddad, N. Pekergin. "*Petri Nets Compositional Modeling and Verification of Flexible Manufacturing Systems*" submitted to CASE'11

**[BDDHP11c]:** P. Ballarini, H. Djafri, M. Dufлот, S. Haddad, N. Pekergin. "*COSMOS: a Statistical Model Checker for Hybrid Automata Stochastic Logic*" tool paper, submitted to 8th Int. Conference QEST'11

in preparation

**[BDDHP11d]:** P. Ballarini, H. Djafri, M. Dufлот, S. Haddad, N. Pekergin. "*The Hybrid Automata Stochastic Logic*" Journal paper

**COSMOS web-page:** <http://www.lsv.ens-cachan.fr/~djafri/cosmos/> (under construction)

THANK YOU FOR YOUR ATTENTION

**acknowledgements:** this work has been partially funded by the **ANR Checkbound** project