Cosmos: Evolution of a Statistical Model Checking Platform

Paolo Ballarini, Benoît Barbot

Cosmos: Evolution of a Statistical Model Checking Platform

Paolo Ballarini
CentraleSupélec
Université Paris Saclay
4 Rue Joliot-Curie, Gif-sur-Yvette, 91195, France
paolo.ballarini@centralesupelec.fr

Benoît Barbot
LACL
Univ Paris Est Creteil,
F-94010 Creteil, France
benoit.barbot@lacl.fr

ABSTRACT

Cosmos is a statistical model checker for Hybrid Automata Stochastic Logic (HASL). HASL uses Linear Hybrid Automata (LHA), a generalization of Deterministic Timed Automata (DTA), to describe accepting execution paths of a Discrete Event Stochastic Process (DESP), a class of stochastic models which includes, but is not limited to, Markov chains. As a result, HASL verification turns out to be a unifying framework where sophisticated temporal reasoning is naturally blended with elaborate reward-based analysis. Cosmos takes as input a DESP (described in terms of a Generalized Stochastic Petri Net (GSPN)), a LHA and an expression Z representing the quantity to be estimated. It returns a confidence interval estimation of Z. Cosmos is written in C++ and is freely available to the research community. It is jointly developed by researchers of the *Institute National de Recherche en Informatique et Automatque* (INRIA) and of the *Laboratoire Algorithmique Complexité et Logique* (LACL) of the *Université Paris-Est Créteil*. Since its introduction [8] the tool has evolved with the addition of a number of new features including support for rare-events systems and for hybrid systems.

Keywords

Cosmos, performance evaluation, statistical model checking, rare events, stochastic simulation, Petri nets.

1. INTRODUCTION

Within the realm of software tools for performance modeling of stochastic systems the academic community has proved very active over the last few decades. In this respect tools may be distinguished according to different aspects such as the supported input formalism (i.e., general purpose high-level modelling formalism v. customised modelling language), the kind of mathematical approaches they rely upon (e.g. numerical methods v. stochastic simulation), the expressiveness of the property language they feature (if any) and finally the domains of application they have been demonstrated on. GreatSPN [19] and Möbius [17], two pivotal tools using extensions of Petri nets as input formalism and supporting both numerical as well as simulation engines, have undergone constant extensions since their introduction, leading e.g., to the addition of quantitative [3] as well as qualitative [4] model checking functionalities. Amongst native model checking tools targeting Markov models PRISM [24] and MRMC [22] opened the way with the former rapidly becoming the tool of reference throughout academics worldwide. If PRISM provides the user with a syntax-based language for specifying Markov models, MRMC requires a low-level representation in terms of the corresponding probability/rate matrix as input. Initially equipped with numerical engines for solving probabilistic model checking problems in more recent times they both have been extended with simulation-based engines for statistical model checking. Properties languages include the probabilistic extensions of CTL, i.e. PCTL and CSL, (although probabilistic LTL has also been added to PRISM) enriched with reward-based properties. Amongst native model checking tools Marcie [20] uses Petri Nets as input formalism and supports several verification engines including one for qualitative (CTL) properties and others targeting quantitative (CSL and its reward-based extension CSRL) properties (through either exact numerical analysis, approximated numerical analysis or stochastic simulation). To overcome the limits of Markov models model checking tools that rely on resource-greedy exact numerical analysis engines, in more recent times, statistical model checking (SMC) became increasingly prominent, starting with tools like Ymer [26], VESTA [25] up to more recent ones such as e.g. PLASMA-Lab [21] and SBIP 2.0 [23] and it is to this category of tools that Cosmos belongs. The most eminent characteristic of Cosmos is the expressiveness of the property language, which being based on hybrid automata allows for naturally blending state, event and reward-conditions in the formulation of the property of interest, hence letting Cosmos a tool naturally suited for (highly expressive) performance analysis.

2. OVERVIEW OF COSMOS

The first version of the Cosmos tool [8] supported basic statistical model checking functionalities for the HASL formalism. Since then the tool has evolved in a number of directions. The three main new features are the support of colored Petri Nets, facilities to estimate rare event using variance reduction see (section 4.1) and handling of hybrid systems (see section 4.3). Many small features have been added: various new input file format for GSPN making easy to use graphical Petri Net editors like "GreatSPN Editor". Output options allow to easily choose which data to export and to plot graphs to visualize them. There were also many optimizations and many bug fixes.

Below we summarise the basic structure of the tool and

TOSME2021 – Tools for stochastic modelling and evaluation

briefly resume the extensions introduced as of the first version of the tool.

At its current state COSMOS tool consists of about 22000 lines of C++ code. The tool relies on code generation to perform efficient simulation. Figure 1 illustrates the architecture of COSMOS which consists of three main parts:

- **Code generation**. This part is in charge for parsing of the input files (and of the command line options) so to create the necessary data structures for storing the Petri net representation of the DESP model $\mathcal{M}$ and of the considered property expressed in terms of an LHA $\mathcal{A}$. Then optimised C++ code for the simulator of the product process $\mathcal{M} \times \mathcal{A}$ is generated. The resulting code is compiled by a C++ compiler and linked with the simulator library yielding a corresponding binary of the actual $\mathcal{M} \times \mathcal{A}$ simulator program.
- **Cosmos runtime**. Cosmos simulator library implements the core algorithm for synchronisation. This includes 1) the implementation of the stochastic generation of events based on the pseudo random number generator provided by the C++ Boost library as well as 2) the handling of the generated stochastic events in an event queue.
- **Statistical Engine**. This part is in charge for launching several copies of the simulator process and for aggregating the results issued by each simulator copy. Based on the chosen statistical parameters (e.g., confidence-level, interval-width), a procedure decides whether enough trajectories have been simulated and stops all simulators when needed. Then, HASL expressions are evaluated and several output files are produced according to options. The computation of confidence intervals uses the boost library for the computation of quantiles of the normal distribution function and binomial distribution.

Statistical methods. COSMOS supports a family of statistical methods that are applied depending on the nature of the formula and on the assumptions on the model. For boolean-valued formulae (that correspond to the comparison of a probability measure with a given threshold) COSMOS applies *sequential hypothesis testing*, a very efficient method where the *sample size* (i.e., the number of generated trajectories) is (sequentially) established at *runtime* depending on the successive outputs. For *quantitative* formulae (those that correspond to estimating the first moment of a random variable) the applied method depends on the nature of the random variable. For Bernoulli random variables (i.e., formulae that evaluates to a probability measure) COSMOS uses Clopper-Pearson bounds method, whereas for formulae that involve a generic bounded random variable, it uses the Chernoff-Hoeffding bounds method, both of which employ a *sequential* sampling scheme. In the general case, it is based on the standard Gaussian approximation related to the average of independent and identically distributed random variables.

Cosmos extensions. Since its introduction COSMOS has been extended with a number of functionalities so to address the challenges raised by different projects. These include:

1) Rare events. The simulation engine has been specialised to handle simulation of rare events. By altering the sampling of distribution in the simulation algorithm efficient estimations of rare event probabilities have been computed [14].

2) Colored Petri Nets. The simulation engine has been extended to Stochastic Symmetric Nets (SSNs) [2], a coloured extension of GSPN, i.e., the original input formalism of the tool. SSNs allow for a much more compact model representation hence effectively permit COSMOS to treat much more complex systems has demonstrated in a study of vehicular wireless networks protocols [7].

3) Cosimulation. Thank to the very small footprint of the generated simulator, COSMOS has been used for the cosimulation of a pacemaker software with a model of the human heart. The generated simulator was small enough to fit in the memory of microcontroller on which live power consumption was measured [15].

4) Custom probability distribution defined by polynomials have been used to sample uniform trajectory for timed automata [10, 11].

5) Synchronised simulation. Facilities for the synchronised simulation of several models have implemented as well as a simulator for hybrid systems given as simulink. The outcome is the ability to simulate an hybrid model inside a stochastic environment which has been applied to the verification of autonomous vehicles [5].

Input formalisms. COSMOS takes as input a Petri Net and an LHA. For the Petri Net, althought the main file format `.GrML` which is required for very specialized model (Colored, Hybrid, rare events), a other file format are available namely `.PNML` the standard for Petri net, `.prism` through an encoding of the prism language to Petri Net. The LHA can be provided in the `.GrML` file format as well or as a custom textual file format for ease of use. COSMOS provide facility to generate LHA on the fly for common indices or simple logical formula.

Targets. COSMOS has been mainly used as a research tool to investigate new statistical model checking techniques and logic and its application to different domains. In recent times it has been used in an industrial context to model autonomous vehicle controllers within a joint research project in partnership with `https://www.irt-systemx.fr`. Further to research COSMOS is used for teaching to students of the *"Université Paris Est Créteil"* as well as to young researchers from the Real Time Systems community at the *"Ecole d'été temps-réel"* (ETR) summer school (`https://etr2021.ensma.fr/`).

3. AVAILABILITY

COSMOS is an open source tool distributed under the GPLv3 licence. It can be freely downloaded from its homepage [18] where one can find: the source code of the tool, prebuild binaries for Mac OS and Linux, the tool documentation, and a step by step guide to run the tool. As COSMOS relies on code generation and on the libexpat and libboost C++ libraries, in order to run the binary version one will also need a C++ compiler installed as well as the libexpat and libboost libraries (both of which are freely available on the internet). In order to build the tool from the source code version a further few dependencies (which are listed on COSMOS homepage) must be taken into account.

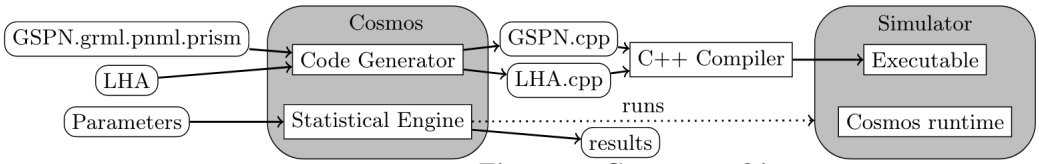

Figure 1: Cosmos architecture

4. CASE STUDIES

The capability of Cosmos performance analysis has been demonstrated through a number of case studies in a variety of different domains, including computerised systems, biological systems, hybrid systems, systems characterised by rare events and manufacturing systems. In the remainder we briefly report about a some of them.

4.1 Systems with rare events

In a probabilistic reachability problem, one wants to compute the probability p to reach a final state, from the initial state of a given model. When the size of the model becomes too large for numerical analysis and when the probability of interest p is too small for direct statistical simulation, rare event acceleration methods become necessary. A possible approach to tackle this problem is to build an importance sampling scheme for which a statistical simulation becomes possible. In order to be useful, this importance sampling has to be built carefully to ensure a reduction of the variance. In [14] a method to to produce such an importance sampling is described. This approach have been implemented in Cosmos and have been successfully applied to waiting queue example as well as a biological case study [12, 13].

4.2 Systems biology

Analysis of biological systems call for expressive tools for performance analysis. In this respect the potential of the Cosmos tool has been demonstrated in different case studies.

Protein synthesis performances. In gene expression proteins are synthesized through a 2-steps process consisting of *transcription* followed by *translation* phase both of which are time consuming events driven by stochasticity. Cosmos has been applied to the analysis of perfromances of stochastically delayed models of gene expression [9].

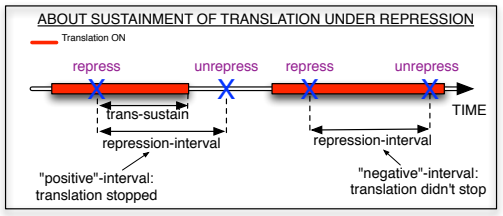

Figure 2: Sustainabilty of translation once gene is reprerssed

A number of expressive indicators related to the performances of the protein syntheis dynamics have been identified and assessed, including, some related to the *sustainability* of translation (i.e., how long synthesis goes on once transcription is repressed, see Figure 2) and a few concerned with assessing how the translation process compares with the transcription process (e.g. the average throughput of translation within a transcription interval, see Figure 3).

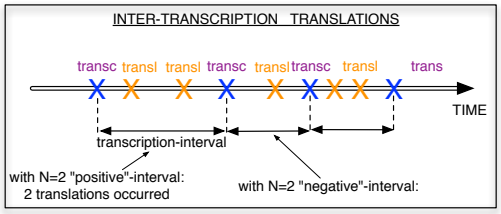

Figure 3: Inter-transcription translations

Stochastic oscillators. Many cellular circuits are characterised by periodicity which combined with low-population numbers (i.e., genetic circuits) is of a stochastic nature. Performance analysis of stochastic oscillators entails the definition of suitable indicators capable of capturing meaningful features related to noisy periodic behaviour (Figure 4), like, for example, the moments of the oscillation period and amplitude. Classical temporal logic reasoning featured by languages such as CSL and its reward-based extensions, turned out to be insufficiently expressive for measuring oscillations.

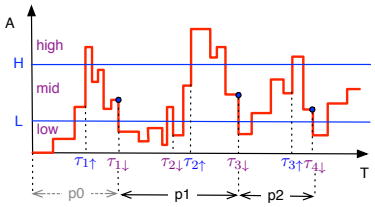

Figure 4: Noisy periodicity of stochastic oscillators

A number of performance indicators capturing relevant aspects of oscillation, such as, e.g., the moments of the oscillation period and amplitude, have been characterised and assessed through Cosmos on a few examples of biological oscillators [6].

4.3 Hybrid system

Cosmos can perform the synchronised simulation of a hybrid system modelled with differential equation and a stochastic environment moddelled as a Petri Net. We can illustrate this framework with the well known toy (but still relevant) example of a device with two heaters prone to faults and using bang-bang controllers to keep the temperature in a room between $20^{\circ}C$ and $25^{\circ}C$. The system is

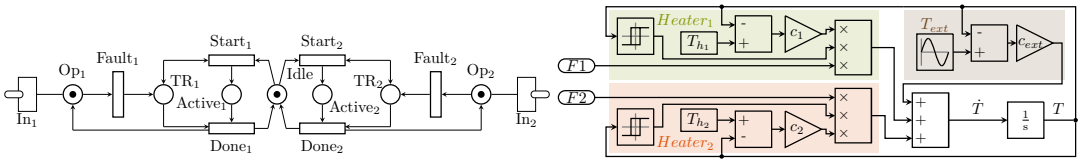

Figure 5: A Petri net and A Simulink model computing differential equations for the double heater system with fault.

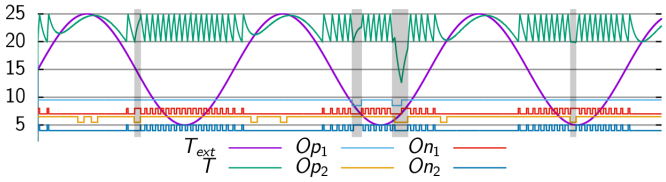

Figure 6: A trace of simulation: T and T_{ext} represent the inside and outside temperatures, Op_i corresponds to heater i being operational and On_i corresponds to heater i being switched on. Gray areas highlight failure of at least one heater when $T_{ext} < T_{min}$.

modeled by a stochastic Petri net (Figure 5) with randomized faults and repairs. The evolution of room temperature and heater behaviours are hybrid and thus are modeled in Simulink (Figure 5). The fault transitions of the net have an exponential time distribution (with different rates). The repairman, initially at the Idle state, randomly chooses which (faulty) heater he will repair, then proceeds in fixed time and goes back to the Idle state. By default, both heaters are working (places Op_1 and Op_2 have a token).

The Simulink model handles the differential equations for both heaters, and for the outside temperature which is modelled by a sine wave (T_{ext}). A bang-bang controller is a very simple hysteresis controller where the heater is switched on ($On_i = 1$) when the temperature decreases to T_{min} and switched off ($On_i = 0$) when the temperature increases to T_{max}. The inputs $F1$ and $F2$ receive respectively the content of places Op_1 and Op_2.

Figure 6 shows a simulation of the system. In the first period there is only a small failure of heater 2, and we can observe the bang-bang behaviours of the system. In the second period both heaters fail at the same time while the outside temperature is low, thus the temperature quickly drops to $13°C$ before the first heater is repaired.

4.4 Computerised systems

COSMOS has proved effective in a number of applications concerned with computerised systems.

Passage time distributions. The passage time distribution is a specific type of performance index concerned with the probability of the delay between a starting event and the corresponding ending event. It is particularly useful when reasoning about properties related with Service Level Agreements (SLA) or safety requirements. In [1] COSMOS has been used to assess a variety of *passage time distributions* for an order-handling business process. Further to "classical" passage time measures COSMOS has been shown to be readily suitable for assessing relevant conditional passage time distribution such as, e.g. *"the passage-time for an ordered good to be delivered given that it was out-of-stock"* or also *"the passage-time for an ordered good to be delivered given that it is not out-of-stock and that the total delay for checking its availability and shipping it does not exceed a given time bound K"*.

Wireless protocols for vehicular network The IEEE 802.11 family of medium access control (MAC) protocols are in charge for regulating the access to a shared medium in wireless networks. They employ randomised delay procedures as a means for minimising traffic collisions aiming at optimising the overall network performances. Intelligent Transportation Systems (ITS) entail hybrid communication scenarios where both Inter-Vehicle-Communication (IVC), based on ad-hoc connections between moving vehicles, and Roadside-Vehicle-Communication (RVC), concerned with the exchanging of information between moving vehicles and fixed roadside infrastructures, co-exist. In order to cope with specific needs of such hybrid scenarios, an adaptation of the IEEE 802.11 MAC layer, named 802.11p Wireless Access in Vehicular Environments (WAVE) standard, has been introduced for Vehicular Ad Hoc Networks (VANETs). The 802.11p MAC is an adaptation of the CSMA/CA scheme, to the case of a network whereby traffic with 4 different levels of priority. COSMOS have been shown an effective tool to tackle the complexity of modelling networks controlled by the 802.11p protocol. In [7] a reasonably sized, fully configurable, colored Petri net model of 802.11p networks of arbitrary size is illustrated and its performances are analysed through COSMOS.

5. CONCLUSION

Future developments of COSMOS include equipping it with functionalities for parameter-estimation driven by spatio-temporal constraints based on the recently introduced Automata-ABC appproach [16].

6. REFERENCES

[1] E. G. Amparore, P. Ballarini, M. Beccuti, S. Donatelli, and G. Franceschinis. Expressing and computing passage time measures of gspn models with hasl. In *Petri Nets*, volume 7927 of *LNCS*, pages 110–129. Springer, 2013.

[2] E. G. Amparore, B. Barbot, M. Beccuti, S. Donatelli, and G. Franceschinis. Simulation-based verification of hybrid automata stochastic logic formulas for

stochastic symmetric nets. In *SIGSIM-PADS '13, Montreal, QC, Canada, May 19-22*, pages 253–264. ACM, 2013.

[3] E. G. Amparore and S. Donatelli. MC4CSLTA: an efficient model checking tool for CSLTA. In *QEST 2010, Williamsburg, Virginia, USA, 15-18 September*, pages 153–154. IEEE, 2010.

[4] E. G. Amparore, S. Donatelli, and F. Gallà. A ctl* model checker for petri nets. In *PETRI NETS 2020, Paris, France, June 24-25, 2020, Proceedings*, volume 12152 of *LNCS*, pages 403–413. Springer, 2020.

[5] Y. D. B. Barbot, B. Bérard and S. Haddad. Integrating Simulink Models into the Model Checker Cosmos. In *Petri Nets 2018*, volume 10877 of *LNCS*, pages 363–373. Springer, 2018.

[6] P. Ballarini. Analysing oscillatory trends of discrete-state stochastic processes through HASL statistical model checking. *Int. J. Softw. Tools Technol. Transf.*, 17(4):505–526, 2015.

[7] P. Ballarini, B. Barbot, and N. Vasselin. Performance modelling of access control mechanisms for local and vehicular wireless networks. In *VALUETOOLS 2019, Palma de Mallorca, Spain, March 12-15*, pages 111–118. ACM, 2019.

[8] P. Ballarini, H. Djafri, M. Duflot, S. Haddad, and N. Pekergin. COSMOS: a statistical model checker for the hybrid automata stochastic logic. In *QEST'11*, pages 143–144. IEEE Computer Society Press, sep. 2011.

[9] P. Ballarini, J. Mäkelä, and A. S. Ribeiro. Expressive statistical model checking of genetic networks with delayed stochastic dynamics. In *CMSB*, pages 29–48, 2012.

[10] B. Barbot, N. Basset, M. Beunardeau, and M. Kwiatkowska. Uniform sampling for timed automata with application to language inclusion measurement. In *QEST16*, pages 175–190, Cham, 2016. Springer.

[11] B. Barbot, N. Basset, and T. Dang. Generation of Signals Under Temporal Constraints for CPS Testing. In *NFM 2019*, volume LNCS 11460, Houston, TX, United States, May 2019.

[12] B. Barbot, S. Haddad, M. Heiner, and C. Picaronny. Rare event handling in signalling cascades. In *SIMUL'14*, pages 126–131, Nice, France, Oct. 2014. XPS.

[13] B. Barbot, S. Haddad, M. Heiner, and C. Picaronny. Rare event handling in signalling cascades. *International Journal on Advances in Systems and Measurements*, 8(1-2):69–79, June 2015.

[14] B. Barbot, S. Haddad, and C. Picaronny. Coupling and importance sampling for statistical model checking. In *TACAS'12*, LNCS, Tallinn, Estonia, Mar. 2012. Springer. To appear.

[15] B. Barbot, M. Kwiatkowska, A. Mereacre, and N. Paoletti. Building Power Consumption Models from Executable Timed I/O Automata Specifications. In *HSCC 2016*, 2016.

[16] M. Bentriou, P. Ballarini, and P.-H. Cournède. Automaton-abc: a statistical method to estimate the probability of spatio-temporal properties for parametric markov population models. *Theoretical Computer Science*, 2021.

[17] G. Clark, T. Courtney, D. Daly, D. Deavours, S. Derisavi, J. Doyle, W. Sanders, and P. Webster. The mobius modeling tool. In *Proceedings 9th International Workshop on Petri Nets and Performance Models*, pages 241–250, 2001.

[18] Cosmos home page. https://cosmos.lacl.fr/.

[19] Greatspn home page. http://www.di.unito.it/greatspn/index.html.

[20] M. Heiner, C. Rohr, and M. Schwarick. Marcie – model checking and reachability analysis done efficiently. In *PETRI NETS 2013*, pages 389–399, Berlin, Heidelberg, 2013. Springer.

[21] C. Jegourel, A. Legay, and S. Sedwards. A platform for high performance statistical model checking – plasma. In *TACAS2021*, pages 498–503, Berlin, Heidelberg, 2012. Springer.

[22] J. P. Katoen and I. S. Zapreev. MRMC home page. http://www.mrmc-tool.org/trac/.

[23] B. L. Mediouni, A. Nouri, M. Bozga, M. Dellabani, A. Legay, and S. Bensalem. SBIP 2.0: Statistical Model Checking Stochastic Real-time Systems. In *ATVA 2018* , pages 536–542, Los Angeles, CA, United States, Oct. 2018. Springer.

[24] Prism home page. http://www.prismmodelchecker.org.

[25] K. Sen, M. Viswanathan, and G. Agha. Statistical model checking of black-box probabilistic systems. In *CAV04*, pages 202–215, Berlin, Heidelberg, 2004. Springer.

[26] H. L. S. Younes. Ymer: A statistical model checker. In *CAV05*, pages 429–433. Springer, 2005.