MathMC: A Mathematica-Based Tool for CSL Model Checking of Deterministic and Stochastic Petri Nets

José M. Martínez Boudewijn R. Haverkort

Faculty of Electrical Engineering, Mathematics and Computer Science
University of Twente, the Netherlands
{martinez,brh}@cs.utwente.nl

Abstract

Deterministic and Stochastic Petri Nets (DSPNs) are a widely used high-level formalism for modeling discrete-event systems where events may occur either without consuming time, after a deterministic time, or after an exponentially distributed time. CSL (Continuous Stochastic Logic) is a (branching) temporal logic developed to express probabilistic properties in continuous time Markov chains (CTMCs). In this paper we present a Mathematica-based tool that implements recent developments for model checking CSL style properties on DSPNs. Furthermore, as a consequence of the type of process underlying DSPNs (a superset of Markovian processes), we are also able to check CSL properties of Generalized Stochastic Petri Nets (GSPNs) and labeled CTMCs.

Keywords*: DSPNs, CSL, model checking, Markov process, Markov regenerative process.*

1 Introduction

CSL is a widely used logic for the specification of system performance and dependability properties for Markovian models. DSPNs have been used widely for modeling and evaluation of all kinds of performance and dependability aspects of computer and communication systems.

MathMC is a Mathematica-based tool that implements recent developments [4] to extend the use of CSL in the context of so-called Markov regenerative processes (MRGPs), a class of stochastic processes that arises from DSPNs. This class of process is more general than CTMCs. Together with the CSL implementation for DSPNs, fully Markovian model checking techniques are also implemented, thus allowing to check properties for GSPNs and labeled CTMCs as well.

This paper is organized as follows. In Section 2 we give an overview of the tool and its capabilities. In Section 3 we present some information related to the algorithms implemented. Section 4 presents current and future work.

2 Tool overview

MathMC comprises two parts, one based on Mathematica [8], and another one implemented in C++. Mathematica was chosen since it provides a flexible model application, evaluation, and result presentation framework due to list processing, symbolic manipulation, functional programming, pattern matching, and graphics facilities.

Part of the numerical analysis of DSPNs, as well as the state space generation (reachability graph) is carried out by the algorithms implemented in SPNica, a prototype tool developed by German [2, 7] for the analysis of DSPNs. Besides the above implementation, algorithms for transient and steady-state analysis of CTMCs, as well as part of the analysis of DSPN are implemented in C++ (together with the data structures needed to store the relevant information, i.e., vectors and sparse matrices). The communication between Mathematica and the C++ compiled code is accomplished via MathLink, a toolkit for interfacing external code with Mathematica.

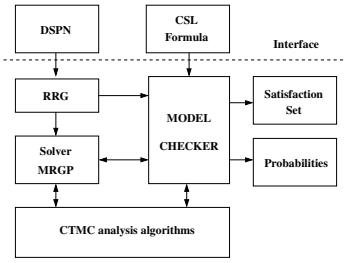

Figure 1. Block Diagram of the Tool

The structure of MathMC is summarized in the block diagram of Figure 1. **DSPN** is the specification of the model following the SPNica syntax, for a detailed explanation see [2]. **RRG** is the set of functions that generate the reduced reachability graph (RRG) of a Petri net. The exploration of states is carried out through a breadth-first-search algorithm. The information of the process is acquired via the *RRG*. **Solver MRGP** comprises the set of functions for the numerical evaluation of DSPNs, using either steady state or transient analysis algorithms. The logic formulas, in **CSL Formula**, are specified using the names of places defined in the DSPN model together with logical functions taken from Mathematica. **CTMC Analyisis Algorithms** implements the numerical methods involved in the analysis of Markovian processes, which are also part of the analysis of DSPNs. **MODEL CHECKER** is the set of functions for evaluating the operators involved in the model checking procedure. *MODEL CHECKER* functions interact with the

Third International Conference on the Quantitative Evaluation of Systems (QEST'06)
0-7695-2665-9/06 $20.00 © 2006 IEEE

Solver MRGP and the *CTMC Analysis Algorithms* in order to calculate those probabilities needed to determine the logical value of the expression being checked. **Satisfaction Set** is the set of states that satisfies the formula being checked. **Probabilities** corresponds to the probabilities associated to each state for which the CSL formula is evaluated.

Besides the description given above, two other formalism are implemented: GSPNs and labeled CTMCs. Efficient implementation of the algorithms, as well as the data structures involved in model checking of these formalisms is accomplished via compiled code linked to MathMC. Importing functions are also available for exchanging model information with other tools, specifically, MRMC [6], ETMCC [1], and Möbius [5].

3 CSL model checking of DSPNs

CSL expressions are defined in terms of atomic properties which refer to the number of tokens in the places of the DSPN. CSL model checking algorithms (originally intended for continuous-time Markov processes) have been extended to Markov regenerative processes, specifically those underlying a DSPN. The following examples illustrate the type of properties that can be expressed:

- $\mathcal{S}_{>0}[\texttt{num\_processors\_up} > 2]$: "in the long-run, the probability there are more than two processors working properly is greater than 0".
- $\mathcal{P}_{[0.5,0.8]}[(\texttt{buffer} < 10)\ \mathcal{U}^{\leq 5}\ (\texttt{buffer} = 10)]$: "the probability of the system reaching a buffer occupancy of 10 jobs within 5 time units when the buffer fills up lies in the interval $[0.5, 0.8]$".

The analysis of DSPNs is based on the method of supplementary variables [2], where a state of the underlying process (MRGP) is characterized by both a marking and the elapsed time of the deterministic transition that is enabled in that marking (if there is any).

The evaluation techniques implemented in our tool require some restrictions with respect to deterministic transitions: (1) at most one deterministic transition is enabled in each marking and (2) the firing policy is preemptive repeat different, i.e., when a timed transition is preempted, its already performed work is lost.

With respect to CSL, MathMC implements the steady-state and time-bounded until operator, following the procedure described in [4]. In the case of obtaining the satisfaction set for a steady-state formula, $\mathcal{S}_{\trianglelefteq p}(\phi)$, we compute the steady-state probabilities (or long-term proportion of time, if there are no limiting probabilities) for those states satisfying the formula ϕ. Steady-state probabilities are calculated according to the procedure described in [2].

In the case of the satisfaction set for a time-bounded until formula, $\mathcal{P}_{\trianglelefteq p}(\phi_1\ \mathcal{U}^{\leq t}\ \phi_2)$, we follow the same procedure that for the case of CTMCs, i.e., making absorbing those states that satisfy $(\neg\phi_1 \vee \phi_2)$ and considering the transient probability of being in states that satisfy ϕ_2 at time t, given we start in a state that satisfies ϕ_1. In order to calculate the transient probabilities we use the methodology developed in [2, 3]. Special attention must be taken when we make some states absorbing, since, it may be the case that a transition changes from being a non-preemtive one to be a preemtive one. For more details on this issue we refer to [4]. Furthermore, for the verification of the time-bounded until operator we had to assume that a deterministic transition may be initially enabled, but its elapsed time must be assumed zero. This assumption is required in order to be able to use the well-established transient algorithms.

From a practical point of view, the applicability of model checking techniques in DSPNs lags behind the CTMC case. Verifying a CSL formula for DSPNs is more time/space consuming than for CTMCs due to the fact we are dealing with non-memoryless transitions. The time-bounded next operator has been left out for efficiency reasons, as discussed in [4].

4 Future developments

Further work must concentrate on overcoming some limitations imposed by the assumptions for the time-bounded until operator, i.e., information about the elapsed time that a deterministic transition has been enabled must be considered.

Furthermore, the implementation of some numerical algorithms for the analysis of DSPNs must be changed from Mathematica code to C++ compiled code to increase efficiency.

References

[1] ETMCC. Erlangen-Twente Markov Chain Checker. `http://www7.informatik.uni-erlangen.de/etmcc`.

[2] R. German. *Performance Analysis of Communication Systems: Modeling with Non-Markovian Stochastic Petri Nets*. John Wiley & Sons Ltd., 2000.

[3] A. Heindl and R. German. A fourth-order algorithm with automatic stepsize control for the transient analysis of DSPNs. *IEEE Transactions on Software Engineering*, 25(2):194–206, March 1999.

[4] J. M. Martínez and B. R. Haverkort. CSL model checking of deterministic and stochastic Petri nets. In *Proceedings of the 13th GI/ITG Conference in Measuring, Modelling and Evaluation of Computer and Communication Systems*, pages 265–282, Nürnberg, March 2006. VDE Verlag.

[5] Möbius. `http://www.mobius.uiuc.edu`.

[6] MRMC. The Markov Reward Model Checker. `http://www.cs.utwente.nl/~zapreev/mrmc`.

[7] SPNica. `ftp://ftp.wiley.co.uk/pub/books/german/`.

[8] Wolfram Research. Mathematica 5.2. `http://www.wolfram.com`.