Great-Nsolve: a tool integration for (Markov Regenerative) stochastic Petri nets

E.G. Amparore[1], P. Buchholz[2], and S. Donatelli[1]

[1] Dipartimento di Informatica, Università degli Studi di Torino, Torino, Italy
[2] Department of Computer Science, TU Dortmund, Germany,
`amparore@di.unito.it, donatelli@di.unito.it,`
`peter.buchholz@cs.tu-dortmund.de`

Abstract. This paper presents *Great-Nsolve*, the integration of GreatSPN (with its user-friendly graphical interface and its numerous possibilities of stochastic Petri net analysis) and *Nsolve* (with its very efficient numerical solution methods) aimed at solving large Markov Regenerative Stochastic Petri Nets (MRSPN). The support for general distribution is provided by the alphaFactory library.

1 The baseline

Generalized stochastic stochastic Petri nets (GSPN) [1] are a stochastic extension of place/transition to associate exponentially distributed delays to transitions. The stochastic process described by a GSPN is a continuous time Markov chain (CTMC). Markov Regenerative Stochastic Petri Nets (MRSPN) [16] are an extension of GSPNs to allow transitions to have a generally distributed delay, given that, in each state, at most one non-exponential transition is enabled. The solution of a MRSPN is based on the solution of its underlying Markov Regenerative Processes (MRgP) which is typically based on the construction and solution of the embedded discrete time Markov chain (DTMC) at regeneration points (the so-called global kernel) and of the CTMC that describes the stochastic behavior of the net in-between regeneration points (the so-called local kernel). It is well known that the embedded DTMC matrix can be very dense (even if the MRgP transition matrix is sparse) and therefore it can be built, stored and solved only for small systems (thousands of states). This approach is called explicit, in contrast to the implicit, matrix-free technique [18,6] which does not require to build and store the embedded DTMC, but works with the (usually sparse) transition matrix.

When a net model is formulated as a set of components, the state space and transition matrix (and even the solution vector [14]) can be effectively represented in Kronecker form [20,11] allowing us to treat much larger state spaces.

A previous paper [5] has introduced the combination of matrix-free, as in [18,6], and Kronecker representation to solve MRSPN. This demo tool paper describes how the matrix-free solutions provided by GreatSPN [4] and the advanced Kronecker-based techniques provided by *Nsolve* [12] have been integrated to implement the theory presented in [5] and have been made accessible in an easy manner through the graphical interface of GreatSPN [3]. The support for general distributions is provided by the alphaFactory library [8]. The resulting *Great-Nsolve* tool allows the user to experiment and compare solutions of MRSPN based on a large variety of techniques: implicit and matrix-free approaches [2], already present in GreatSPN, and Kronecker-based approaches, thanks to the tool integration presented in this paper.

The *Nsolve* tool. *Nsolve* [12] is a collection of advanced numerical solution methods for the computation of stationary distributions of large structured Markov chains. The tool is written in C and uses a two level hierarchical Kronecker structure to represent generator matrices which can be built from a flat description of a model as a network of synchronized components [13]. The matrix structure is described in [15]. Due to the modular structure and the availability of basic functions for numerical operations it is possible to easily integrate new numerical solution methods at the level of C functions. Different interfaces exist to combine the numerical methods provided by *Nsolve* with other tools as front- and/or back-ends. First, a file interface using small component matrices stored in sparse format has been defined [12], then the internal C data structures for sparse matrices can be used and the solution functions can be called directly. Finally, models can be provided in APNN format [10], an XML based format to describe extended stochastic Petri nets. At the back-end *Nsolve* provides the stationary vector and results computed from an appropriate reward structure. *Nsolve* cannot deal with MRSPN models directly.

The GreatSPN tool. GreatSPN [4] is a tool for the qualitative and stochastic analysis of (stochastic) Petri nets and various extensions of them. Through the support on a Java-based, highly portable graphical interface (GUI) [3] that allows one to draw and compose nets and to play the (colored) token game, GreatSPN offers a qualitative analysis that includes state space construction, model-checking of CTL properties based on decision diagrams, various structural analysis techniques (like P- and T- invariants) and a stochastic analysis with a rich variety of numerical solutions for ergodic and non-ergodic models, as well as stochastic model checking of the CSL$^{\text{TA}}$ logic [17]. GreatSPN has been tailored for teaching, but it is also a tool with advanced solution techniques that regularly participates in the Petri net model checking competition [19]. For the integration with *Nsolve* the most relevant features of the tool are the solvers [2] for MRSPNs, that include an explicit and a matrix-free solver as well as a component-based method [7] for non-ergodic MRSPN.

The alphaFactory library This library [8] supports the definition of general distributions through their PDF $f(x)$, and the computation of the α factors to be used inside any uniformization method implementation. Examples of $f(x)$ are $I[\delta]$ for a Dirac impulse at time δ, i.e. a deterministic event, $R(a, b)$ for a uniform rectangular signal in the $[a, b]$ range, or more sophisticated functions like $f(x) = \frac{\lambda^r}{(r-1)!} \cdot x^{r-1} \cdot \mathrm{e}^{-\lambda x}$ for the Erlang distribution with r phases of rate λ. Syntactic sugar for common distributions (like *Uniform*, *Erlang* or *Pareto*) is available too.

2 *Great-Nsolve*: the integration

Changes to the GreatSPN GUI GreatSPN already has support for MRSPN: the GUI supports the labelling of general transitions with a PDF function $f(x)$ in *alphaFac-*

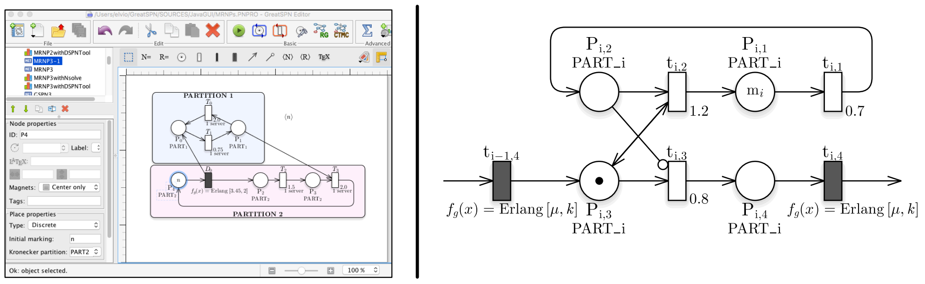

Fig. 1: Net partition in the GUI and a component of a Moving Server System model.

tory form, and the execution of different forms of MRSPN solutions, including a matrix-free approach. In *Great-Nsolve* the net model of the GUI has been extended to include the module information (a partition of places) to be exploited by *Nsolve*. The same net model is then shared by both the GreatSPN solvers (that ignores it) and the Kronecker-based solvers of *Nsolve*. The extensions of the GUI include: 1) Places can be *partitioned* by the use of labels: places with the same label are in the same partition; Such separation is compulsory for *Nsolve*, and it is ignored by other solvers. 2) The GUI outputs the model in the internal representation of *Nsolve*, i.e. the APNN model. This format requires the place bounds to be known in advance, hence place bounds are first computed from the place invariants. Nets not fully covered by P-invariants are currently not supported. 3) the results computed by *Nsolve* are shown back in the GUI (directly on the net or in tabular form), as for any other GreatSPN solver. These changes allow any user to exploit the efficient techniques presented in [5] in a fully transparent, 1-click approach manner. As a by-product of the integration, Kronecker based solution for GSPNs are also 1-click available.

Extension of *Nsolve* for MRP solutions Two main modifications were needed: 1) Computation of the hierarchical state space structure cannot be done with the standard algorithm from [13] since the hierarchical matrix structure for MRgPs differs from the matrix structure of CTMCs and requires the implementation of the algorithm in [5]. 2) The steady-state solution for MRgPs has been implemented to work using matrix in Kronecker form inside a matrix-free iterative method, similar to the one in [6]. Currently the iterative methods supported are the Power method, GMRES and BiCG-Stab. The matrix-free solution requires the computation of the local kernels using Uniformization w.r.t. the PDF $f(x)$ of the general transition, and the computation of the Uniformization coefficients is delegated to the alphaFactory library.

3 *Great-Nsolve* in action

Great-Nsolve is available as a virtual machine with all software preinstalled and ready to use that can be downloaded at: `http://www.di.unito.it/~greatspn/VBox/Ubuntu64_with_Great-nsolve.ova`. The code of GreatSPN with the *Nsolve* integration can be found at `https://github.com/greatspn/SOURCES`.
AlphaFactory can be found at `https://github.com/amparore/alphaFactory`.

The Kronecker-based solution of MRSPN in *Great-Nsolve* is limited to ergodic systems. If the system is non-ergodic, but finite, the model can still be solved using the matrix-free techniques and the component-based method [7,2] of GreatSPN.

To show the use of the integrated tool consider the Moving Server System (MSS) obtained by concatenating 4 stations (components) modeled by the net in Figure 1(right). Places are partitioned according to stations. The black and thick transitions have a general distribution. This is an Erlang-3 with rate 1. Experiments are performed for a variable number m_i of requests arriving to the station's queues (in the reported experiments the same value for all queues). The number of macro states (top states in the hierarchical structure of the MRSPN) is constant, and equal to 12. Macro states are built automatically. Table 1 summarizes the comparison in space based on the number of states and of non-zeros of the matrices used. The following cases are considered (left to right): 1) structured matrix-free representation of the MRgP matrices for the MRSPN in Figure 1, 2) structured representation of the CTMC of the GSPN obtained from the MRSPN in Figure 1 through phase-expansion (each Erlang-3 transition is substituted by a sequence of three exponential transitions) 3) matrix-free representation of the MRgP matrices for the MRSPN in Figure 1 and 4) explicit solution for the same matrices as 3). The first two cases are the new solvers of *Great-Nsolve* , while the last two cases are the best solutions available in GreatSPN. The explicit solution (last column) is not able to cope with large state spaces, while the matrix-free unstructured, although better than explicit, occupies much more space than the corresponding structured case proposed in this paper, as an example: 840 entries against more than 7 millions for the $m_i = 20$ case. Note also that the structured approach does not suffer much for the substitution of general distribution with their phase expansion.

	Structured representation (*Great-Nsolve*)						Unstructured representation				
	General (MRgP)			Phase-expansion (CTMC)			General (MRgP)				
	Matrix-free			-			Matrix-free			Explicit	
m_i	states	nnz	Time	states	nnz	Time	states	nnz	Time	nnz	Time
1	128	80	0.1	320	416	0.2	128	384	0.6	544	1.1
10	117128	440	1.8	244904	1856	2.5	117128	543048	11.8	7452116	67.7
20	1555848	840	47.5	3185784	3456	64.3	1555848	7482888	606.8	88318068	12202.1
25	3655808	1040	140.5	7452224	4256	208.5	3655808	17716608	1687.8	-	-
30	7388168	1240	369.3	15014664	5056	613.4	7388168	35987528	4048.4	-	-

Table 1: MSS: State space sizes and memory occupations.

Future work From a theoretical point of view, we are currently working on an optimization of the solution presented in [5]. We are also considering whether the component-based MRgP solution techniques presented in [7] can profit of a Kronecker-based approach for the solution of single components. From a tool point of view we plan to lift the requirement that all places are covered by a p-invariant. If this is not the case, indeed the place bounds cannot be computed through structural analysis, but, if the state space is finite, they can be computed on the actual state space, that can be efficiently generated in GreatSPN using decision-diagrams [9].

References

1. Ajmone Marsan, M., Conte, G., Balbo, G.: A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems. ACM Transactions on Computer Systems 2, 93–122 (May 1984)
2. Amparore, E.G., Donatelli, S.: DSPN-Tool: a new DSPN and GSPN solver for GreatSPN. QEST 2010 pp. 79–80 (2010)
3. Amparore, E.: A new GreatSPN GUI for GSPN editing and CSL$^{\text{TA}}$ model checking. In: Proca of the 11th QEST Conference. vol. 8657 LNCS, pp. 170–173. Springer (2014)
4. Amparore, E.G., Balbo, G., Beccuti, M., Donatelli, S., Franceschinis, G.: 30 Years of GreatSPN, chap. In: Principles of Performance and Reliability Modeling and Evaluation: Essays in Honor of Kishor Trivedi, pp. 227–254. Springer (2016)
5. Amparore, E.G., Buchholz, P., Donatelli, S.: A structured solution approach for Markov Regenerative Processes. In: Procs. of the 11th QEST Conference. pp. 9–24. Springer (2014)
6. Amparore, E.G., Donatelli, S.: Revisiting the matrix-free solution of Markov regenerative processes. Numerical Linear Algebra with Applications 18, 1067–1083 (2011)
7. Amparore, E.G., Donatelli, S.: A component-based solution for reducible Markov regenerative processes. Performance Evaluation 70(6), 400 – 422 (2013)
8. Amparore, E.G., Donatelli, S.: alphaFactory: A tool for generating the alpha factors of general distributions. In: Proceedings of the 14th QEST Conference,. pp. 36–51. Springer (2017)
9. Babar, J., Beccuti, M., Donatelli, S., Miner, A.: GreatSPN enhanced with decision diagram data structures. In: Proceedings of the 31st International Conference on Application and Theory of Petri Nets and Concurrency, June 21-25, 2010. vol. 6128 LNCS, pp. 416–425
10. Bause, F., Buchholz, P., Kemper, P.: A toolbox for functional and quantitative analysis of DEDS. In: Procs. of the 10th Int. Tools Conf. LNCS, vol. 1469, pp. 356–359. Springer (1998)
11. Buchholz, P., Ciardo, G., Donatelli, S., Kemper, P.: Complexity of memory-efficient Kronecker operations with applications to the solution of Markov models. INFORMS Journal on Computing 12(3), 203–222 (2000)
12. Buchholz, P.: Markov matrix market. `http://ls4-www.cs.tu-dortmund.de/download/buchholz/struct-matrix-market.html`
13. Buchholz, P.: Hierarchical Structuring of Superposed GSPNs. IEEE Transactions on Software Engineering 25(2), 166–181 (1999)
14. Buchholz, P., Dayar, T., Kriege, J., Orhan, M.C.: On compact solution vectors in Kronecker-based Markovian analysis. Perform. Eval. 115, 132–149 (2017)
15. Buchholz, P., Kemper, P.: Kronecker based matrix representations for large Markov models. In: Validation of Stochastic Systems. LNCS, vol. 2925, pp. 256–295. Springer (2004)
16. Choi, H., Kulkarni, V.G., Trivedi, K.S.: Markov regenerative stochastic Petri nets. Performance Evaluation 20(1-3), 337–357 (1994)
17. Donatelli, S., Haddad, S., Sproston, J.: Model checking timed and stochastic properties with CSL$^{\text{TA}}$. IEEE Transactions on Software Engineering 35(2), 224–240 (2009)
18. German, R.: Iterative analysis of Markov regenerative models. Performance Evaluation 44, 51–72 (April 2001)
19. Kordon, F., & all: Complete results for the 2019 edition of the Model Checking Contest (2019), `http://mcc.lip6.fr/2019/results.php`
20. Plateau, B., Fourneau, J.M.: A methodology for solving Markov models of parallel systems. J. Parallel Distrib. Comput. 12(4), 370–387 (1991)