

A framework to design and solve Markov Decision Well-formed Net models

M. Beccuti, D. Codetta-Raiteri, G. Franceschinis *
Dip. di Informatica, Univ. del Piemonte Orientale
Alessandria, Italy
{beccuti, raiteri, giuliana}@mf.n.unipmn.it

Serge Haddad †
LAMSADE CNRS, Univ. de Paris Dauphine
Paris, France
haddad@lamsade.dauphine.fr

1 Introduction

The *Markov Decision Process* (MDP) [7] formalism is widely used for modeling systems which exhibit both non deterministic and probabilistic behaviors (e.g. distributed systems, resource management systems, ...). Unfortunately, if the system is particularly complex then its modeling at the MDP level may be very hard; so in [6] a higher-level formalism called *Markov Decision Well-formed Net* (MDWN) was proposed. The MDWN allows to describe the system in terms of its components and their interactions, while the MDP describes directly the state space and the state transitions. The MDWN model is more compact and readable: in particular, it is possible to define a complex non deterministic or probabilistic behavior as a composition of simpler non deterministic or probabilistic steps. In the MDWN formalism, the probabilistic behavior of the system is clearly distinct from the non deterministic one; actually they are designed as two separate *Petri Nets* (PN): the probabilistic PN (N^{pr}) and the non deterministic PN (N^{nd}).

From the analysis point of view, the MDWN allows to derive efficient algorithms (e.g. taking advantage from the intrinsic symmetries of the system) reducing the analysis complexity. In fact the technique of *Symbolic Reachability Graph* (SRG) developed for the *Well-formed Nets* (WNs) [1] was adapted for the MDWN so that a reduced MDP (whose states represent aggregates of equivalent ordinary states) can be produced and directly used to compute the solution of the original problem. In this paper, we introduce the *MDWNSolver* framework for the system modeling and the optimization of performability measures based on the MDWN formalism. This framework combines together several tools: *Draw-Net* [3] for the model design, *algebra* to compose the non deterministic PN with the probabilistic one, *WN(S)RG* to build the (S)RG of the resulting com-

posed net, *RG2MDP* to generate the corresponding MDP from the (S)RG, and *MDP solver* based on the library *graphMDP* [5] to compute the optimal strategy and the optimal average reward. The tools *algebra* and *WN(S)RG* belong to the *GreatSPN* suite [2]. In the following section we are going to describe this architecture with more details.

2 Framework architecture

The architecture of our framework for the MDWN design and solution, is depicted in Fig. 1 where the framework components are indicated by rectangles, the component invocations are shown as thick arrows, while the exchange of models and of data is represented by thin arrows.

The *Draw-Net Modeling System* (*DMS*) [3] is exploited as graphical interface and as solution manager. The *DMS* model editor called *Draw-Net* [3] allows the user to design the N^{pr} and the N^{nd} and to define the reward function. To this aim, the *Draw-Net* tool has been adapted to draw such models by defining an ad-hoc formalism [3]. The *DMS* is able to save each PN in the *GreatSPN* file format (.net and .def file) [2], by means of an ad-hoc filter. The solution manager of the *DMS* is exploited to execute in the correct order the framework components, and to manage the exchange of models and of data between them. The solution process comprises four steps:

- (1) The N^{pr} and the N^{nd} models are composed by using the *algebra* tool: this requires to automatically generate a *Composition Net* and to perform a composition by place superposition [6]. The result of this step is a WN.
- (2) The WN is the input of *WN(S)RG* generating the (S)RG. After this, each probabilistic path in the (S)RG is substituted by a single transition. The resulting graph is stored in a file (.mdwn).
- (3) From the graph obtained in step 2 an MDP is derived by means of the *RG2MDP* converter using the reward definition. In case multi-step non deterministic decisions are present in the model, these are reduced to a single step complex decision.
- (4) The obtained MDP is stored in an XML file which

*The work of these authors was partially supported by Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), under the EU CRUTIAL project.

†The work of this author was supported in part by ANR-06-SETI-002 project Checkbound.

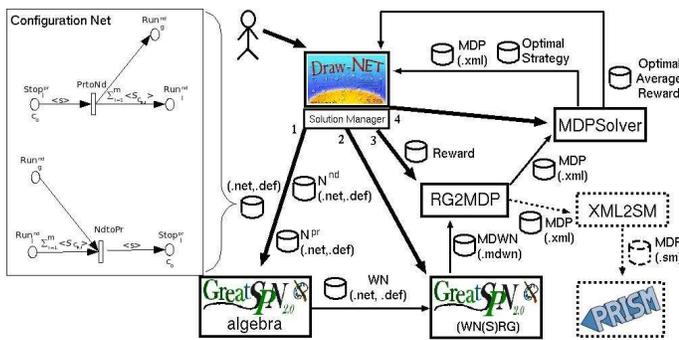


Figure 1. Framework architecture.

is in turn processed by the *MDPSolver* producing the *optimal strategy* and the *optimal average reward*. The MDP together with the *optimal strategy*, and the *optimal average reward* can be visualized by *Draw-Net*.

We are also planning the integration of *PRISM* [4] in our framework, with the role of alternative solver or model checker of the MDP. This is represented by the dashed part in the architecture shown in Fig. 1.

3 Application example

An implementation of such framework is available and some experimental results are now available: for instance in [6] a model representing a multiprocessor system was presented. The system is composed by several couples of processor and local memory with the addition of one global memory which may fail and can be repaired. A processor can use the global memory when its local memory is faulty. The probabilistic part represents the component faults while the non deterministic part represents the component repair strategy. A cost is associated with every component repair and a penalty is defined when the system is not available; the goal is to find an optimal strategy minimizing the system costs (repair + penalty).

The memory and time requirements for the solution of this MDWN are summarized in Table 1. The table shows

	Proc.Mem=2		Proc.Mem=3		Proc.Mem=4	
	#States	Time	#States	Time	#States	Time
RG	40.088	6s	1.619.392	1.367s	58.741.760	out of memory
MDP	441	7s	4.078	5.183s	out of mem	out of mem
SRG	20.316	5s	283.128	284s	2.734.826	2.477s
MDP'	219	4s	831	534s	2.360	48.722s

Table 1. Number of states and computation time of the example described in [6].

the number of states and the computation time of the RG,

the corresponding MDP, the SRG and the corresponding MDP' for different numbers of processors and memories, performed with an AMD Athlon 64 2.4Ghz with 4Gb memory capacity. The computation time of MDP/MDP' is the sum of their generation and solution time.

By comparing the RG line with the SRG line it is possible to appreciate the reduction of memory and time requirements due to the state aggregation obtained through the SRG technique. The difference between the (S)RG size and the corresponding MDP size, suggests that the modeler has exploited the possibility to decompose a complex non deterministic/probabilistic state transition into simpler steps (referred to a single component rather than to a system as a whole): as the number of components grows, this may become a crucial aspect w.r.t. the usability of the formalism.

References

- [1] G. Chiola, C. Dutheillet, G. Franceschinis, and S. Haddad. Stochastic well-formed coloured nets for symmetric modelling applications. *IEEE Transactions on Computers*, 42(11):1343–1360, November 1993.
- [2] G. Chiola, G. Franceschinis, R. Gaeta, and M. Ribaud. GreatSPN 1.7: Graphical editor and analyzer for timed and stochastic petri nets. *Performance Evaluation, special issue on Performance Modeling Tools*, 24(1-2):47–68, November 1995.
- [3] D. Codetta-Raiteri, G. Franceschinis, and M. Gribaudo. Defining formalisms and models in the Draw-Net Modelling System. In *4th Int. Workshop on Modelling of Objects, Components and Agents (MOCA)*, pages 123–144, Turku, Finland, June 2006.
- [4] A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker. PRISM: A tool for automatic verification of probabilistic systems. In *12th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems*, volume 3920 of LNCS, pages 441–444, Vienna, Austria, 2006. Springer.
- [5] P. Laroche. Graphmdp: A new decomposition tool for solving markov decision processes. *Int. Journal on Artificial Intelligence Tools*, 10(3):325–343, 2001.
- [6] M. Beccuti, G. Franceschinis, and S. Haddad. Markov Decision Petri Net and Markov Decision Well-Formed Net Formalisms. In *28th Int. Conf. on application and theory of Petri nets and other models of concurrency (ICATPN) 2007*, volume 4546 of LNCS, pages 43–62. Springer-Verlag Berlin Heidelberg, 2007.
- [7] M. L. Puterman. *Markov Decision Processes. Discrete Stochastic Dynamic Programming*. Wiley, 2005.