Available online at www.sciencedirect.com

SCIENCE DIRECT®

Electronic Notes in Theoretical Computer Science

Electronic Notes in Theoretical Computer Science 154 (2006) 73–78

www.elsevier.com/locate/entcs

A Framework for Stochastic System Modelling and Analysis

Work in Progress

Sebastian Menge[1] Georgios Lajios[2]

Software Technology
University of Dortmund
Germany

Abstract

Stochastic Graph Transformation combines the benefits of graphical modelling with stochastic analysis techniques. In this paper we report on our framework SMA for Stochastic Modelling and Analysis, and SGT⋆, a tool which uses the framework for Stochastic Graph Transformation.

Keywords: graph transformation, stochastic analysis, model checking, tool support

1 Introduction

In distributed and mobile systems with volatile bandwidth and fragile connectivity, non-functional aspects such as performance and reliability become more and more important. To analyse such properties, stochastic methods are required. At the same time such systems are characterized by a high degree of architectural reconfiguration. This gave rise to the notion of *Stochastic Graph Transformation* [5], which combines the benefits of using graph transformation for system modelling with the power of stochastic analysis as known from areas such as queueing theory, Markov theory, or recently probabilistic model checking.

[1] Email: sebastian.menge@uni-dortmund.de
[2] Email: georgios.lajios@uni-dortmund.de

1571-0661
doi:10.1016/j.entcs.2005.03.033

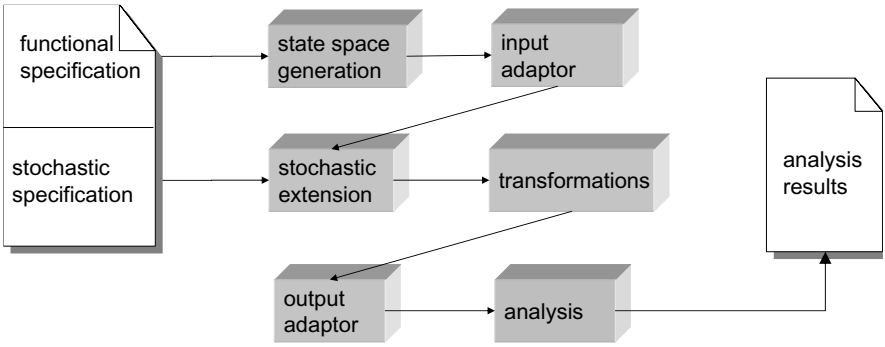

Fig. 1. the architecture of the framework

While this combination is conceptually beneficial, it is still difficult to integrate graphical modelling with stochastic analysis when it comes to tool support. Though many tools are available to meet the requirements of either graphical modelling or stochastic analysis, there is still lack of tool support to combine both aspects. Thus, to analyse case studies for stochastic graph transformation, there was the need to develop something to bridge the gap between intuitive modelling and good analysis capabilities.

When developing the tool, we did not want to restrict ourselves to a specific approach, but wanted to retain flexibility both in the modelling and the analysis paradigm. Therefore, we decided to build a framework to accomodate the integration of different approaches to stochastic modelling and analysis.

In this paper, we present such a framework. Section 2 presents the overall architecture and main ideas, while Section 3 discusses how we used the framework for stochastic graph transformation. Section 4 concludes the paper and presents further ideas.

2 A Framework for Stochastic Modelling and Analysis

The main aim of the Sma framework is to keep modelling and analysis of stochastic systems separate but to allow tight integration of specific tools at the same time.

Since we want to reuse the existing powerful tools when investigating stochastic systems, we have to adapt to these tools and map between the different kinds of models they deal with. Furthermore, we want to perform non-trivial transformations on the models, such as merging modular specifications [6], minimizing the state space etc. To meet these requirements, we use *pipes and filters* [14] as main architectural style: Each pipe represents a model and each filter transforms data from one representation into another.

Because the filters are independent of each other, we are able to reuse the adaptors and transformations in many different combinations.

Figure 1 depicts the architecture of the SMA framework. The system specification is twofold: we separate functional and stochastic specification. Using the functional specification, the first step is *state space generation*. Because this is done by external tools, the *input adaptor* transforms the state space representation into our own data structures. In the next step, we extend the state space with the stochastic parts of the specification. The resulting model could subsequently be *transformed*. This is an important step, because most often, the input transition system is not in a form that is easy to analyse. At last, the model is exported through an *output adaptor* and eventually *analysed*.

Thus, we focus on two key aspects: The extension of a state-based system to obtain some kind of stochastic model (such as a Discrete or Continuous Time Markov Chain) and the transformation of this model such that it is easier to analyse. For the modelling and analysis part we are able to reuse existing tools.

The separation of functional and stochastic aspects in the specification is most reasonable since many of the existing approaches to stochastic system modelling extend existing formalisms with stochastic information. Examples are stochastic Petri nets [8] or stochastic process algebras [1]. Since both rely on labelled transition systems it would be possible to build up a tool chain to investigate such systems using our framework.

3 Stochastic Graph Transformation with SGT*

Since our focus is on stochastic graph transformation (SGT), we now illustrate the framework by discussing SGT*, our tool to analyse SGT systems along with a simple example.

A SGT System consists of a Graph Transformation System [13] together with a mapping which associates with each rule a positive real number, the rate of the exponentially distributed application delay of the transformation. We showed in [5] how this leads to a Continuous Time Markov Chain (CTMC).

As a proof of concept, we modelled and analysed an example situation in mobile communications with SGT*. Given a fixed network of base stations, a mobile device can connect to one of them in order to make a call, and disconnect afterwards. A station may be broken with a certain probability, and will then be repaired. The actual state of a station (broken or not broken) is expressed by a boolean attribute, whose value switches between true and false, stochastically triggered by rules *fail* and *repair*. The configuration of such a mobile network can easily and intuitively be represented as an attributed

Fig. 2. *connect* and *disconnect*, resp. *fail* and *repair*

rule name p	rate $\rho(p)$	rule name p	rate $\rho(p)$
repair	500	connect	10000
fail	1	disconnect	10000

Fig. 3. Rates associated with the rules

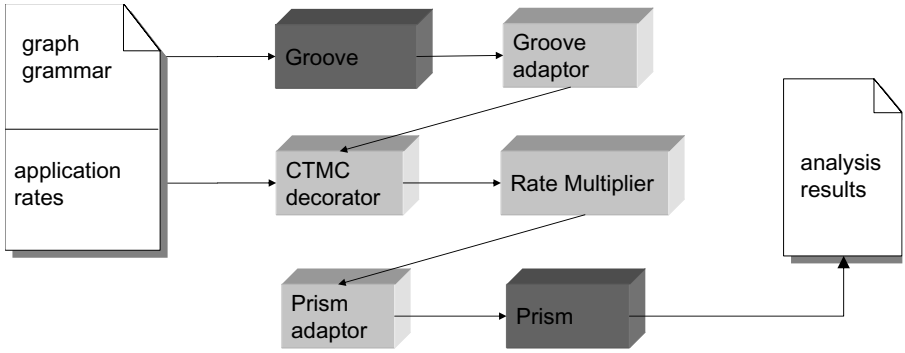

Fig. 4. SGT*

graph. The corresponding rules are shown in Fig. 2.

Apart from the graph grammar, the specification of a SGT System consists of a table containing the rates of the exponential distribution[3] associated with the rules (Fig. 3). We generate the labelled transition system defined by the graph grammar with GROOVE [12] and use SGT* to combine the GROOVE output with the rates, yielding a CTMC, exported in PRISM [10] format. PRISM is a stochastic model checking tool which allows for a variety of stochastic models and logics, including CTMCs and Continuous Stochastic Logic (CSL).

As Fig. 4 shows, SGT* consists of different components: The GROOVE

[3] We assume all tranformations to be exponentially distributed. While this is standard for the reliability of hardware components [9], also call attempt rates und call holding times are often modelled in this manner (see the discussion in [3]).

adaptor to understand GROOVE's representation of a labelled transition system, the *CTMC decorator* which maps the application rates of the transformation rules to the corresponding transitions, a so-called *Rate Multiplier* to replace multiple transitions with identical label, source and target with one transition and the multiplied rate (see [7]), and the PRISM adaptor to generate a CTMC in the PRISM language.

We emphasize, that the transformation-step (multiplication of the rates), can be easily extended with additional transformations. This could be used to cope with parameterized rules, prioritized rules, typing information (GROOVE has no typing-concept)

4 Conclusion and Perspectives

In this paper, we presented the SMA framework and SGT*, our tool for stochastic graph transformation. By using the pipes and filters architecture we gain a lot of flexibility. First, we can easily replace modelling and analysis tools, for example there are other probabilistic model checkers besides PRISM or interesting specification languages like PEPA[4]. Second, we could also model more complex systems. For example, if a rule is associated with an Erlang distribution, SGT* can be extended by a filter which introduces virtual states into the CTMC in order to simulate the Erlang distribution. Every probability density function with rational Laplace transform can be treated in that manner [2]. At last, apart from CTMCs, we plan to support other stochastic models like Discrete Time Markov Chains or Hybrid Systems [11].

References

[1] Ed Brinksma and Holger Hermanns. Process algebra and markov chains. In J.-P. Katoen E. Brinksma, H. Hermanns, editor, *FMPA 2000*, number 2090 in LNCS, pages 183–231. Springer, 2001.

[2] D.R. Cox. A use of complex probabilities in the theory of stochastic processes. *Proc. Camb. Phil. Soc., 51, 1955, pp. 313-319*, 51:313–319, 1955.

[3] Yuguang Fang, Imrich Chlamtac, and Yi-Bing Lin. Channel occupancy times and handoff rate for mobile computing and pcs networks. *IEEE Trans. Comput.*, 47(6):679–692, 1998.

[4] Stephen Gilmore and Jane Hillston. The PEPA workbench: A tool to support a process algebra-based approach to performance modelling. In *Computer Performance Evaluation*, pages 353–368, 1994.

[5] Reiko Heckel, Georgios Lajios, and Sebastian Menge. Stochastic graph transformation systems. In Hartmut Ehrig, Gregor Engels, and Francesco Parisi-Presicce, editors, *Graph Transformations: Second International Conference, ICGT 2004, Rome, Italy*, volume 3256 of *LNCS*, pages 210–225. Springer, 2004.

[6] Reiko Heckel, Georgios Lajios, and Sebastian Menge. Modulare Analyse Stochastischer Graphtransformationssysteme. In Peter Liggesmeyer, Klaus Pohl, and Michael Goedicke,

editors, *Software Engineering*, volume 64 of *LNI*, pages 141–152. GI, 2005. ISBN 3-88579-393-8.

[7] Reiko Heckel, Georgios Lajios, and Sebastian Menge. Stochastic Graph Transformation Systems. Technical Report 154, Lehrstuhl fr Softwaretechnologie, Uni-Dortmund, Dortmund, Germany, March 2005.

[8] D. Kartson, G. Balbo, S. Donatelli, G. Franceschinis, and Giuseppe Conte. *Modelling with Generalized Stochastic Petri Nets*. John Wiley & Sons, Inc., New York, NY, USA, 1994.

[9] Dimitri Kececioglu. *Reliability Engineering Handbook*, volume 1. Prentice Hall, 1993.

[10] M. Kwiatkowska, G. Norman, and D. Parker. PRISM: Probabilistic symbolic model checker. In T. Field, P. Harrison, J. Bradley, and U. Harder, editors, *Proc. 12th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation (TOOLS'02)*, volume 2324 of *LNCS*, pages 200–204. Springer, 2002.

[11] O. Maler and A. Pnueli, editors. *Hybrid Systems: Computation and Control: 6th International Workshop, HSCC 2003*. LNCS 2623. Springer, 2003.

[12] A. Rensink. The GROOVE simulator: A tool for state space generation. In J.L. Pfaltz, M. Nagl, and B. Bhlen, editors, *Applications of Graph Transformation with Industrial Relevance Proc. 2nd Intl. Workshop AGTIVE'03, Charlottesville, USA, 2003*, volume 3062 of *LNCS*. Springer, 2004.

[13] G. Rozenberg, editor. *Handbook on Graph Grammars: Foundations*, volume 1. World Scientific, Singapore, 1997.

[14] Mary Shaw and David Garlan. *Software architecture: perspectives on an emerging discipline*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.