A linear process algebraic format for probabilistic systems with data*

Joost-Pieter Katoen
MOVES Group
RWTH Aachen University, Germany
katoen@cs.rwth-aachen.de

Jaco van de Pol, Mariëlle Stoelinga, Mark Timmer
FMT Group
University of Twente, The Netherlands
{vdpol, marielle, timmer}@cs.utwente.nl

In this presentation we introduce a novel linear process algebraic format for probabilistic automata. The key ingredient is a symbolic transformation of probabilistic process algebra terms that incorporate data into this linear format while preserving strong probabilistic bisimulation. This generalises similar techniques for traditional process algebras with data, and — more importantly — treats data and data-dependent probabilistic choice in a fully symbolic manner, paving the way to the symbolic analysis of parameterised probabilistic systems.

1 Introduction

Efficient model-checking algorithms exist, supported by powerful software tools, for verifying qualitative and quantitative properties for a wide range of probabilistic models. These techniques are applied in areas like security, randomised distributed algorithms, systems biology, and dependability and performance analysis. Major deficiencies of probabilistic model checking are the *state explosion problem* and the restricted treatment of *data*.

As opposed to process calculi like μCRL [14] and E-LOTOS that support rich data types, the treatment of data in modeling formalisms for probabilistic systems is mostly neglected. Instead, the focus has been on understanding random phenomena and modeling the interplay between randomness and nondeterminism. Data is treated in a restricted manner: probabilistic process algebras typically allow a random choice over a fixed distribution, and input languages for model checkers such as the reactive module language of PRISM [25] or the probabilistic variant of Promela [2] only support basic data types, but neither support more advanced data structures or *parameterised*, i.e., state-dependent, random choice. To model realistic systems, however, convenient means for data modeling are indispensable.

Although parameterised probabilistic choice is semantically well-defined [7], the incorporation of data yields a significant increase of, or even an infinite, state space. Applying aggressive abstraction techniques for probabilistic models (e.g., [9, 1, 17, 20, 22]) obtain smaller models at the model level, but the successful analysis of data requires *symbolic* reduction techniques. These minimise stochastic models by syntactic transformations at the *language level* in order to minimise state spaces *prior to* their generation, while preserving functional and quantitative properties. Other approaches that partially deal with data are probabilistic CEGAR [18, 21] and the probabilistic GCL [19].

*This research has been partially funded by NWO under grant 612.063.817 (SYRUP) and grant Dn 63-257 (ROCKS), and by the European Union under FP7-ICT-2007-1 grant 214755 (QUASIMODO).

Submitted to:
QAPL 2010

2 Approach

Our aim is to develop symbolic minimisation techniques — operating at the syntax level — for data-dependent probabilistic systems. The starting point for our work is covered by this presentation. We define a probabilistic variant of the process algebraic μCRL language [14], named prCRL, which treats data as first-class citizens. The language prCRL contains a carefully chosen minimal set of basic operators, on top of which syntactic sugar can be defined easily, and allows data-dependent probabilistic branching. It enables the specification of systems incorporating probabilistic choice, nondeterministic choice, conditional behaviour, parallel composition, hiding and encapsulation.

As a basic example, consider the following prCRL specification. It models a system that continuously and randomly writes data elements of some finite type D. After each write, it beeps with probability 0.1:

$$B = \tau \sum_{d:D} \tfrac{1}{|D|} : \mathrm{send}(d)(0.1 : \mathrm{beep} \cdot B \oplus 0.9 : B)$$

To enable symbolic reductions, we define a two-phase algorithm to transform prCRL terms into LPPEs: a probabilistic variant of *linear process equations* (LPEs) [3], which is a restricted form of process equations akin to the Greibach normal form for string grammars. Basically, an LPPE is a flat process description, consisting of a collection of summands that describe symbolic transitions. Each summand can perform an action and probabilistically move on to some next state, given that a certain condition based on the system state is true.

As an example, we provide an LPPE that is strongly probabilistic bisimilar to the system B defined above, given that it is initialised with $\mathrm{pc} = 1$:

$$\begin{aligned}
&X(\mathrm{pc} : \{1,2,3\}, x : D) = \\
&\qquad \mathrm{pc} = 1 \Rightarrow \tau \sum_{d:D} \tfrac{1}{|D|} : X(2,d) \\
&\quad + \mathrm{pc} = 2 \Rightarrow \mathrm{send}(x)(0.1 : X(3,x) \oplus 0.9 : X(1,x)) \\
&\quad + \mathrm{pc} = 3 \Rightarrow \mathrm{beep} \cdot X(1,x)
\end{aligned}$$

Note that two additional process parameters had to be introduced. The first is used as a sort of program counter, whereas the second is used for remembering the value that was chosen to send.

The algorithm we provide is able to transform every well-formed prCRL specification to an LPPE. We prove that this transformation is correct, in the sense that it preserves strong probabilistic bisimulation [23]. Similar linearisations have been provided for plain μCRL [8] and a real-time variant thereof [26].

To motivate the expected advantage of a probabilistic linear format, we draw an analogy with the purely functional case. There, LPEs provided a uniform and simple format for a process algebra with data. As a consequence of this simplicity, the LPE format was essential for theory development and tool construction. It lead to elegant proof methods, like the use of invariants for process algebra [3], and the cones and foci method for proof checking process equivalence [15, 11]. It also enabled the application of model checking techniques to process algebra, such as optimisations from static analysis [12] (including dead variable reduction [24]), data abstraction [10], distributed model checking [6], symbolic model checking (either with BDDs [5] or by constructing the product of an LPE and a parameterised μ-calculus formula [13, 16]), and confluence reduction [4], a form of partial-order reduction. In all these cases, the LPE format enabled a smooth theoretical development with rigorous correctness proofs (often checked in PVS), and a unifying tool implementation, enabling the cross-fertilisation of the various techniques by composing them as LPE-LPE transformations. The LPPE format will allow similar methods to be developed for probabilistic systems.

We developed a Haskell implementation of the linearisation algorithm for prCRL specifications. Based on results obtained using this implementation, we will demonstrate the whole process of going from prCRL to LPPE and applying reductions to this LPPE by discussing a case study of a leader election protocol.

3 Conclusions and future work

We developed a linear process algebraic format for systems incorporating both nondeterministic and probabilistic choice. The key ingredients are: (1) the combined treatment of data and data-dependent probabilistic choice in a fully symbolic manner; (2) a symbolic transformation of probabilistic process algebra terms with data into this linear format, while preserving strong probabilistic bisimulation.

This work is the first essential step towards the symbolic minimisation of probabilistic state spaces, as well as the analysis of parameterised probabilistic protocols. Our results show that the treatment of probabilities is simple and elegant, and rather orthogonal to the traditional setting [26].

Future work will concentrate on branching bisimulation preserving symbolic minimisation techniques such as confluence reduction [4] — techniques already proven useful for LPEs.

References

[1] L. de Alfaro & P. Roy (2007): *Magnifying-lens abstraction for Markov decision processes*. In: *Proc. of the 19th Int. Conf. on Computer Aided Verification (CAV)*, *LNCS* 4590. pp. 325–338.

[2] C. Baier, F. Ciesinski & M. Größer (2004): *PROBMELA: a modeling language for communicating probabilistic processes*. In: *Proc. of the 2nd ACM/IEEE Int. Conf. on Formal Methods and Models for Co-Design (MEMOCODE)*. pp. 57–66.

[3] M. Bezem & J.F. Groote (1994): *Invariants in Process Algebra with Data*. In: *Proc. of the 5th Int. Conf. on Concurrency Theory (CONCUR)*, *LNCS* 836. pp. 401–416.

[4] S. Blom & J. van de Pol (2002): *State Space Reduction by Proving Confluence*. In: *Proc. of the 14th Int. Conf. on Computer Aided Verification (CAV)*, *LNCS* 2404. pp. 596–609.

[5] S. Blom & J. van de Pol (2008): *Symbolic Reachability for Process Algebras with Recursive Data Types*. In: *Proc. of the 5th Int. Colloquium on Theoretical Aspects of Computing (ICTAC)*, *LNCS* 5160. Springer, pp. 81–95.

[6] Stefan Blom, Bert Lisser, Jaco van de Pol & Michael Weber (2009): *A Database Approach to Distributed State-Space Generation*. *Journal of Logic and Computation* Advance Access, published March 5.

[7] H.C. Bohnenkamp, P.R. D'Argenio, H. Hermanns & J.-P. Katoen (2006): *MODEST: A Compositional Modeling Formalism for Hard and Softly Timed Systems*. *IEEE Transactions of Software Engineering* 32(10), pp. 812–830.

[8] D. Bosscher & A. Ponse (1995): *Translating a process algebra with symbolic data values to linear format*. In: *Proc. of the 1st Workshop on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, *BRICS Notes Series* NS-95-2. pp. 119–130.

[9] P.R. D'Argenio, B. Jeannet, H.E. Jensen & K.G Larsen (2001): *Reachability analysis of probabilistic systems by successive refinements*. In: *Proc. of the Joint Int. Workshop on Process Algebra and Probabilistic Methods, Performance Modeling and Verification (PAPM-PROBMIV)*, *LNCS* 2165. pp. 39–56.

[10] M.V. Espada & J. van de Pol (2007): *An abstract interpretation toolkit for μCRL*. *Formal Methods in System Design* 30(3), pp. 249–273.

[11] W. Fokkink, J. Pang & J. van de Pol (2006): *Cones and foci: A mechanical framework for protocol verification*. Formal Methods in System Design 29(1), pp. 1–31.

[12] J.F. Groote & B. Lisser (2001): *Computer Assisted Manipulation of Algebraic Process Specifications*. Technical Report SEN-R0117, CWI.

[13] J.F. Groote & R. Mateescu (1998): *Verification of Temporal Properties of Processes in a Setting with Data*. In: Proc. of the 7th Int. Conf. on Algebraic Methodology and Software Technology (AMAST), LNCS 1548. Springer, pp. 74–90.

[14] J.F. Groote & A. Ponse (1995): *The syntax and semantics of μCRL*. In: Proc. of Algebra of Communicating Processes, Workshops in Computing. pp. 26–62.

[15] J.F. Groote & J. Springintveld (2001): *Focus points and convergent process operators: a proof strategy for protocol verification*. Journal of Logic and Algebraic Programming 49(1-2), pp. 31–60.

[16] J.F. Groote & T.A.C. Willemse (2005): *Model-checking processes with data*. Science of Computer Programming 56(3), pp. 251–273.

[17] T.A. Henzinger, M. Mateescu & V. Wolf (2009): *Sliding Window Abstraction for Infinite Markov Chains*. In: Proc. of the 21st Int. Conf. on Computer Aided Verification (CAV), LNCS 5643. pp. 337–352.

[18] H. Hermanns, B. Wachter & L. Zhang (2008): *Probabilistic CEGAR*. In: Proc. of the 20th Int. Conf. on Computer Aided Verification (CAV), LNCS 5123. pp. 162–175.

[19] J. Hurd, A. McIver & C. Morgan (2005): *Probabilistic guarded commands mechanized in HOL*. Theoretical Computer Science 346(1), pp. 96–112.

[20] J.-P. Katoen, D. Klink, M. Leucker & V. Wolf (2007): *Three-valued abstraction for continuous-time Markov chains*. In: Proc. of the 19th Int. Conf. on Computer Aided Verification (CAV), LNCS 4590. pp. 311–324.

[21] M. Kattenbelt, M.Z. Kwiatkowska, G. Norman & D. Parker (2009): *Abstraction Refinement for Probabilistic Software*. In: Proc. of the 19th Int. Conf. on Verification, Model Checking, and Abstract Interpretation (VMCAI), LNCS 5403. pp. 182–197.

[22] M.Z. Kwiatkowska, G. Norman & D. Parker (2006): *Game-based abstraction for Markov decision processes*. In: Proc. of the 3rd Int. Conf. on Quantitative Evaluation of Systems (QEST). pp. 157–166.

[23] K.G. Larsen & A. Skou (1991): *Bisimulation through Probabilistic Testing*. Information and Computation 94(1), pp. 1–28.

[24] J. van de Pol & M. Timmer (2009): *State Space Reduction of Linear Processes using Control Flow Reconstruction*. In: Proc. of the 7th Int. Symp. on Automated Technology for Verification and Analysis (ATVA), LNCS 5799. pp. 54–68.

[25] `http://www.prismmodelchecker.org/`.

[26] Y.S. Usenko (2002): *Linearization in μCRL*. Ph.D. thesis, Eindhoven University of Technology.