On-the-fly Confluence Detection for Statistical Model Checking*

Arnd Hartmanns
Computer Science
Saarland University, Saarbrücken, Germany
arnd@cs.uni-saarland.de

Mark Timmer
Formal Methods and Tools, Faculty of EEMCS
University of Twente, The Netherlands
timmer@cs.utwente.nl

Statistical model checking is an analysis method that circumvents the state space explosion problem in model-based verification by combining probabilistic simulation with statistical methods that provide clear error bounds. As a simulation-based technique, it can only provide sound results if the underlying model is a stochastic process. In verification, however, models are usually variations of nondeterministic transition systems. The notion of confluence allows the reduction of such transition systems in classical model checking by removing spurious nondeterministic choices. In this presentation, we show that confluence can be adapted to detect and discard such choices on-the-fly during simulation, thus extending the applicability of statistical model checking to a subclass of Markov decision processes. In contrast to previous approaches that use partial order reduction, the confluence-based technique can handle additional kinds of nondeterminism. In particular, it is not restricted to interleavings. We evaluate our approach, which is implemented as part of the modes simulator for the MODEST modelling language, on a set of examples that highlight its strengths and limitations and show the improvements compared to the partial order-based method.

1 Introduction

Traditional and probabilistic model checking have grown to be useful techniques for finding inconsistencies in designs and computing quantitative aspects of systems and protocols. However, model checking is subject to the state space explosion problem, with probabilistic model checking being particularly affected due to its additional numerical complexity. Several techniques have been introduced to stretch the limits of model checking while preserving its basic nature of performing state space exploration to obtain results that unconditionally, certainly hold for the entire state space. Two of them, partial order reduction (POR) and confluence reduction, work by selecting a subset of the transitions of a model—and thus a subset of the reachable states—in a way that ensures that the reduced system is equivalent to the complete system. POR was first generalised to the probabilistic domain preserving linear time properties [2, 6], with a later extension to preserve branching time properties [1]. Confluence reduction was generalised in [8, 14], preserving branching time properties.

A much different approach for probabilistic models is statistical model checking (SMC) [11, 13, 15, 16, 17]: Instead of exploring—and storing in memory—the entire state space, or even a reduced version of it, simulation is used to generate traces through the state space. This comes at constant memory usage and thus circumvents state space explosion entirely, but cannot deliver results that hold with absolute certainty. Statistical methods such as sequential hypothesis testing are then used to make sure that the

*This work has been supported by the DFG/NWO Bilateral Research Program ROCKS, by NWO under grant 612.063.817 (SYRUP), by the EU FP7 Programme under grant agreements no. 295261 (MEALS) and 318490 (SENSATION), and by the DFG as part of SFB/TR 14 AVACS.

Submitted to:
QAPL 2013

probability of returning the wrong result is below a certain threshold. As a simulation-based approach, however, SMC is limited to fully stochastic models such as Markov chains [9].

Previously, an approach based on POR was presented [3] to extend SMC and simulation to the nondeterministic model of Markov decision processes (MDPs). In that approach, simulation proceeds as usual until a nondeterministic choice is encountered; at that point, an on-the-fly check is performed to find a singleton subset of the available transitions that satisfies the *ample set* conditions of probabilistic POR [2, 6]. If such an ample set is found, simulation can continue that way with the guarantee that ignoring the other transitions does not affect the verification results, i.e., the nondeterminism was *spurious*. Yet, the ample set conditions are based on the notion of *independence* of actions, which can in practice only feasibly be checked on a symbolic/syntactic level (using conditions such as J1 and J2 in [3]). This limits the approach to resolve spurious nondeterminism only when it results from the *interleaving* of behaviours of concurrently executing (deterministic) components.

2 Approach

In this presentation, we present as an alternative to use confluence reduction, which has recently been shown theoretically to be more powerful than branching time POR [8]. It is absolutely vital for the search for a valid singleton subset to succeed in the approach discussed above: one choice that cannot be resolved means that the entire analysis fails and SMC cannot safely be applied to the given model at all. Therefore, any additional reduction power is highly welcome. Furthermore, in practice, confluence reduction is easily implemented on the level of the concrete state space alone, without any need to go back to the symbolic/syntactic level for an independence check. It thus allows even spurious nondeterminism that is internal to components to be ignored during simulation.

Our work consists of three main contributions:

1. Since simulation works with a fully composed, closed system, we can relax the definition of confluence with respect to action labels compared to [8]. We thus achieve more reduction/detection power at no computational cost; yet, we have proven that this adapted notion of confluence still preserves $\text{PCTL}^{*}_{\setminus X}$ formulae.
2. We introduce an algorithm for detecting our new notion of probabilistic confluence on a concrete state space and state its correctness. The algorithm is inspired by, but different from, the one given in [7]; in particular, it does not require initial knowledge of the entire state space and can therefore be used on-the-fly during simulation.
3. We evaluate the new confluence-based approach to SMC on a set of three representative examples using our implementation within the modes statistical model checker [4] for the MODEST modelling language [5]. We clearly identify its strengths and limitations. Since the previous POR-based approach is also implemented in modes, we compare the two in terms of reduction power and, on the one case that can actually be handled by the POR-based implementation as well, performance.

2.1 Related work

Aside from [3] and an approach that focuses on planning problems and infinite-state models [12], the only other solution to the problem of nondeterminism in SMC that we are aware of is recent work by Henriques et al. [10]. They use reinforcement learning, a technique from artificial intelligence, to actually learn the resolutions of nondeterminism (by memoryless schedulers) that *maximise* probabilities for a given bounded LTL property. While this allows SMC for models with arbitrary nondeterministic choices

Table 1: SMC approaches for nondeterministic models (with n states)

approach	nondeterminism	probabilities	memory	error bounds
POR-based [3]	spurious interleavings	max = min	$s \ll n$	unchanged
confluence-based	spurious	max = min	$s \ll n$	unchanged
learning [10]	any	max only	$s \to n$	convergence

(not only spurious ones), scheduling decisions need to be stored for every *explored* state. Memory usage can thus be as in traditional model checking, but is highly dependent on the structure of the model and the learning process. With increasing number of runs of the algorithm, the answer it returns will converge to the actual result, but definite error probabilities are not given. The approaches based on confluence and POR do not introduce any additional overapproximation and thus have no influence on the usual error bounds of SMC. Table 1 gives a condensed overview of the three approaches (where we measure memory usage in terms of the maximal number of states s stored at any time).

3 Results and Conclusions

We defined a more liberal variant of probabilistic confluence, tailored for the core simulation step of statistical model checking. It has more reduction potential than a previous variant at no extra computational cost, but still preserves $\mathrm{PCTL}^{*}_{\setminus X}$. We provided a provably correct algorithm for on-the-fly detection of confluence during simulation and implemented this algorithm in the modes SMC tool. Compared to the previous approach based on partial order reduction [3], the use of confluence allows new kinds of nondeterministic choices to be handled, in particular lifting the limitation to spurious interleavings. In fact, for two of the three examples we will present, SMC is only possible using the new confluence-based technique, showing the additional power to be relevant. In terms of performance, it is somewhat faster than the POR-based approach, but the impact relative to (unsound) simulation using an arbitrary scheduler largely depends on the amount of lookahead that needs to be performed, for both approaches. Again, on two of our examples, the impact was moderate and should in general be acceptable to obtain trustworthy results. Most importantly, the memory overhead is negligible and one of the central advantages of SMC over traditional model checking is thus retained.

As confluence preserves branching time properties, it cannot handle the interleaving of probabilistic choices. Although these can often easily be avoided, for some models POR might work while confluence does not. Hence, neither of the techniques subsumes the other, and it is best to combine them: if one cannot be used to resolve a nondeterministic choice, the SMC algorithm can still try to apply the other. Implementing this combination is trivial and yields a technique that handles the union of what confluence and POR can deal with.

An obvious direction for future work is to compare our techniques to the learning-based approach of Henriques et al. [10] on models where all three methods are applicable. Their approach is fundamentally different in so many aspects that a fair and complete comparison would exceed the scope of this presentation.

References

[1] C. Baier, P. R. D'Argenio & M. Größer (2006): *Partial Order Reduction for Probabilistic Branching Time*. *ENTCS* 153(2).

[2] C. Baier, M. Größer & F. Ciesinski (2004): *Partial Order Reduction for Probabilistic Systems*. In: *QEST*, IEEE Computer Society, pp. 230–239.

[3] J. Bogdoll, L. M. Ferrer Fioriti, A. Hartmanns & H. Hermanns (2011): *Partial Order Methods for Statistical Model Checking and Simulation*. In: *FMOODS/FORTE*, *LNCS* 6722, Springer, pp. 59–74.

[4] J. Bogdoll, A. Hartmanns & H. Hermanns (2012): *Simulation and Statistical Model Checking for Modestly Nondeterministic Models*. In: *MMB/DFT*, *LNCS* 7201, Springer, pp. 249–252.

[5] 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 on Software Engineering* 32(10), pp. 812–830.

[6] P. R. D'Argenio & P. Niebert (2004): *Partial Order Reduction on Concurrent Probabilistic Programs*. In: *QEST*, IEEE Computer Society, pp. 240–249.

[7] J. F. Groote & J. C. van de Pol (2000): *State Space Reduction Using Partial tau-Confluence*. In: *MFCS*, *LNCS* 1893, Springer, pp. 383–393.

[8] H. Hansen & M. Timmer (2012): *A Comparison of Confluence and Ample Sets in Probabilistic and Non-Probabilistic Branching Time*. *Submitted to TCS*. See `http://wwwhome.cs.utwente.nl/~timmer/papers/HT12submission.pdf`.

[9] A. Hartmanns (2010): *Model-Checking and Simulation for Stochastic Timed Systems*. In: *FMCO*, *LNCS* 6957, Springer, pp. 372–391.

[10] David Henriques, João Martins, Paolo Zuliani, André Platzer & Edmund M. Clarke (2012): *Statistical Model Checking for Markov Decision Processes*. In: *QEST*, IEEE C. Soc., pp. 84–93.

[11] Thomas Hérault, Richard Lassaigne, Frédéric Magniette & Sylvain Peyronnet (2004): *Approximate Probabilistic Model Checking*. In: *VMCAI*, *LNCS* 2937, Springer, pp. 73–84.

[12] Richard Lassaigne & Sylvain Peyronnet (2012): *Approximate planning and verification for large Markov decision processes*. In: *SAC*, ACM, pp. 1314–1319.

[13] Axel Legay, Benoît Delahaye & Saddek Bensalem (2010): *Statistical Model Checking: An Overview*. In: *RV*, *LNCS* 6418, Springer, pp. 122–135.

[14] M.Timmer, M. I. A. Stoelinga & J. C. van de Pol (2011): *Confluence Reduction for Probabilistic Systems*. In: *TACAS*, *LNCS* 6605, Springer, pp. 311–325.

[15] Koushik Sen, Mahesh Viswanathan & Gul Agha (2005): *On Statistical Model Checking of Stochastic Systems*. In: *CAV*, *LNCS* 3576, Springer, pp. 266–280.

[16] H. L. S. Younes & R. G. Simmons (2002): *Probabilistic Verification of Discrete Event Systems Using Acceptance Sampling*. In: *CAV*, *LNCS* 2404, Springer, pp. 223–235.

[17] P. Zuliani, A. Platzer & E. M. Clarke (2010): *Bayesian statistical model checking with application to Simulink/Stateflow verification*. In: *HSCC*, ACM, pp. 243–252.