Confluence versus Ample Sets in Probabilistic Branching Time

Henri Hansen
Department of Software Systems
Tampere University of Technology
PO Box 553, FI-33101 Tampere, Finland
henri.hansen@tut.fi

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

To improve the efficiency of model checking in general, and probabilistic model checking in particular, several reduction techniques have been introduced. Two of these, confluence reduction and partial-order reduction by means of ample sets, are based on similar principles, and both preserve branching-time properties for probabilistic models. Confluence reduction has been introduced for probabilistic automata, whereas ample set reduction has been introduced for Markov decision processes.

In this presentation we will explore the relationship between confluence and ample sets. To this end, we redefine confluence reduction to handle MDPs. We show that all non-trivial ample sets consist of confluent transitions, but that the converse is not true. We also show that the two notions coincide if the definition of confluence is restricted, and point out the relevant parts where the two theories differ. The results we present also hold for non-probabilistic models, as our theorems can just as well be applied in a context where all transitions are non-probabilistic.

To show a practical application of our results, we adapt a state space generation technique based on representative states, already known in combination with confluence reduction, so that it can also be applied with partial-order reduction.

1 Introduction

Probabilistic model checking has proved to be an effective way for improving the quality of communication protocols and encryption techniques, but also for studying biological systems or measuring the performance of networks. The omnipresent state-space explosion poses a serious threat to the efficiency of model checking and similar methods, which is why several reduction techniques have been introduced to deal with large systems.

Recently, two reduction techniques from non-probabilistic model checking were generalised to the probabilistic setting: *partial-order reduction* [7, 8, 12] and *confluence reduction* [4, 3]. Both make use of some kind of notion of independence between transitions of a system, and try to reduce the state space by eliminating redundant paths through the system (and therefore often also states).

Partial-order reduction, in the form of *ample sets*, was the first notion to be generalised. In [2] and [5], the concept was lifted from labelled transition systems to Markov decision processes, providing reductions that preserve quantitative $\text{LTL}_{\setminus X}$. These techniques were refined in [1] to also preserve probabilistic CTL, a branching logic. Later, a revision of partial-order reduction for distributed schedulers was introduced and implemented in PRISM [6].

Recently, also confluence reduction was lifted to the probabilistic realm. In [11, 10] a probabilistic variant was introduced that, just like the ample set reduction of [1], preserves branching properties. It was defined as a reduction technique for action-based probabilistic automata [9], but as we will show in this presentation, it can also easily be used in the context of Markov decision processes.

Submitted to:
YR-CONCUR 2011

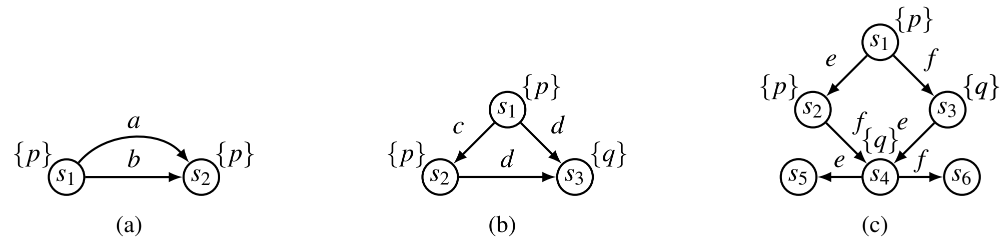
$\{p\}$ s_1 a b s_2 $\{p\}$
(a)
$\{p\}$ s_1 c d $\{p\}$ s_2 d s_3 $\{q\}$
(b)
$\{p\}$ s_1 e f $\{p\}$ s_2 s_3 $\{q\}$ f e $\{q\}$ s_5 e s_4 f s_6
(c)

Figure 1: Confluence triumphs over ample sets.

Ample sets and confluent transitions are defined and detected quite differently: ample sets are defined by first giving an independence relation for the action labels, whereas confluence is a property of a set of (invisible) transitions in the final state space. Even so, the underlying ideas are similar on the intuitive level: the set of outgoing transitions from a state is reduced in such a way that the observable behaviour remains identical, by detecting and then giving priority to transitions that are unobservable and do not influence the potential observable behaviour.

Therefore, an obvious question is: to what extent do confluence and partial-order reduction coincide? We will address this question by comparing the notion of probabilistic ample sets from [1] to the notion of strongly probabilistic confluent sets from [11][1].

2 Results

We showed that probabilistic confluence reduction is strictly more powerful than probabilistic ample set reduction, by proving that every non-trivial ample set can be mimicked by a confluent set. So, if the ample set method allows a state to explore only one of its outgoing transitions, the confluence method also allows this. Therefore, any reduction enabled by the use of ample sets is also enabled by confluence.

However, the converse turns out not to hold. Figure 1 illustrates three MDPs (with the atomic propositions per state indicated in brackets, and having only deterministic transitions), where confluent transitions cannot be chosen as ample sets. Here, we will give an intuitive idea of the definitions of and differences between confluence and ample sets, based on these figures. Note that actions in MDPs are unobservable if they are stuttering, i.e., if they do not change the truth values of the atomic propositions. Also note that we did not even need to include probabilistic transitions for these counter examples.

Example 1. In Figure 1(a), both transitions are stuttering, since the labellings of s_1 and s_2 coincide. Also, they are both deterministic and reach the same target state, making them satisfy the conditions to be confluent. After all, giving one of them priority over the other would not change anything about the observable behaviour of the system. However, the actions a and b are not independent, since they disable each other. Therefore, the only valid ample set for s_1 is $\{a, b\}$, and no reduction can be obtained using partial-order reduction.

In Figure 1(b), the action c is stuttering. It is also confluent, since the d-transition can be taken before and afterwards without influencing the state we end up in. However, $\{c\}$ is not a valid ample set at s_1, as also in this case c and d and dependent (since d disables c).

Finally, in Figure 1(c) the ample set of state s_1 could in principle be $\{e\}$, were it not the case that later on in the system e and f disable each other. Confluence is of a more local nature, and can reduce in this case. Even if the e-action from s_3 to s_4 would be named differently, confluence would not care about

[1] A full paper containing all the details can be found at `http://wwwhome.cs.utwente.nl/~timmer/research.php`.

this, as opposed to the ample set conditions. They cannot deal with differently named stuttering actions, as they rely on a dependency relation of the action level.

In all these three cases, confluence reduction would allow transitions (and therefore possibly also states) to be removed, whereas ample set reduction would not. □

With the above example in mind, we precisely pinpointed in what way confluence is more general than ample sets, by restricting the definition of confluence to make it coincide with ample sets.

The theory is presented in such a way, that the results hold for non-probabilistic automata as well, as they form a special case of the theory, when all probability distributions are deterministic.

2.1 Implications

Our findings imply that results and techniques applicable to confluence, can be used in conjunction with ample sets. As an example, a state space generation technique based on *representative states*, already known in the context of confluence reduction [4], can also be applied with partial-order reduction.

Basically, for this we perceive the system as being partitioned into sets of states that can reach a common *representative* through confluent transitions. As each state in such a set S_i can, due to confluence, simulate all other states in S_i, we can just take one of them as a representative for that set and omit the other states. To make sure that all visible transitions are enabled immediately from the representative (without the need of some confluent transitions first), we need to choose the representative from the terminal strongly connected component (TSCC) of the sub-MDP restricted by the states of S_i and the confluent transitions. The representative can easily be found using a slightly adapted variant of Tarjan's algorithm for strongly connected components, as explained in detail in [4, 3].

The technique makes the cycle condition of ample sets redundant, in addition to further reducing the number of states and transitions. The latter is important, especially if the MDP is to be subjected to further analysis.

3 Conclusions and discussion

We redefined probabilistic confluence reduction to an MDP-based setting, enabling a comparison to probabilistic partial-order reduction based on ample sets. We proved that every non-trivial ample set can be mimicked by a confluent set, and that in some cases reductions are possible using confluence but not using ample sets. Therefore, at least in theory confluence reduction is able to reduce more than the ample set method. We also showed the exact way in which confluence would have to be strengthened for the two notions to coincide. These results also hold for the non-probabilistic variants of the two reduction techniques.

Although the additional power of confluence might be difficult to put to use by syntactic heuristics, our precise comparison of confluence and partial-order reduction at the least fills a gap in our theoretical understanding of the two notions. Moreover, the results we present support the idea that confluence reduction is a well-suited alternative to the thus far more often used partial-order reduction method, for instance in contexts where the heuristics for detecting confluence might be much easier (as seems to be the case for process-algebraic modelling languages).

Our observation that probabilistic ample set reduction can be mimicked by probabilistic confluence reduction has additional implications, some of which are highly practical. One such implication is that the use of representatives for reduced state space generation, already applied earlier in combination with confluence reduction, can also be applied for partial-order reduction. Furthermore, as a nontrivial result, the cycle condition can be omitted, when using representatives.

As both ample sets and confluence are detected symbolically on the language level, the quality of the heuristics applied there will decide which notion works best in practice. The results in this paper already strengthen our theoretical understanding of the two methods, and this is independent of the heuristics that are applied. Also, no matter how such heuristics might be improved, the results in this paper will remain valid.

Even though a case study on probabilistic confluence reduction in [11] seemed to outperform similar reductions based on ample sets, future work could focus more on the relative merits of the two notions in practice and potentially on the improvement of the syntactical heuristics, if some "best of both worlds" approach is found.

3.1 Acknowledgments

This research has been partially funded by NWO under grants 612.063.817 (SYRUP) and Dn 63-257 (ROCKS), by the European Union under FP7-ICT-2007-1 grant 214755 (QUASIMODO), and by the Finnish Foundation for Technology Promotion. We thank Stefan Blom for his helpful suggestions.

References

[1] C. Baier, P. R. D'Argenio & M. Größer (2006): *Partial Order Reduction for Probabilistic Branching Time*. In: *Proc. of the Third Workshop on Quantitative Aspects of Programming Languages (QAPL)*, *ENTCS* 153(2), pp. 97–116.

[2] C. Baier, M. Größer & F. Ciesinski (2004): *Partial Order Reduction for Probabilistic Systems*. In: *Proc. of the 1st Int. Conf. on Quantitative Evaluation of Systems (QEST)*, IEEE Computer Society, pp. 230–239.

[3] S. C. C. Blom (2001): *Partial τ-confluence for efficient state space generation*. Technical Report SEN-R0123, CWI.

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

[5] P. R. D'Argenio & P. Niebert (2004): *Partial Order Reduction on Concurrent Probabilistic Programs*. In: *Proc. of the 1st Int. Conf. on Quantitative Evaluation of Systems (QEST)*, IEEE Computer Society, pp. 240–249.

[6] S. Giro, P. R. D'Argenio & L. María Ferrer Fioriti (2009): *Partial Order Reduction for Probabilistic Systems: A Revision for Distributed Schedulers*. In: *Proc. of the 20th Int. Conf. on Concurrency Theory (CONCUR)*, *LNCS* 5710, Springer, pp. 338–353.

[7] P. Godefroid (1996): *Partial-order Methods for the Verification of Concurrent Systems: an Approach to the State-explosion Problem*. *LNCS* 1032, Springer.

[8] D. Peled (1993): *All from One, One for All: on Model Checking Using Representatives*. In: *Proc. of the 5th Int. Conf. on Computer Aided Verification (CAV)*, *LNCS* 697, Springer, pp. 409–423.

[9] R. Segala (1995): *Modeling and Verification of Randomized Distributed Real-Time Systems*. Ph.D. thesis, Massachusetts Institute of Technology.

[10] M. Timmer, M. I. A. Stoelinga & J. C. van de Pol (2010): *Confluence Reduction for Probabilistic Systems (extended version)*. Technical Report 1011.2314, ArXiv e-prints.

[11] M. Timmer, M. I. A. Stoelinga & J. C. van de Pol (2011): *Confluence reduction for Probabilistic Systems*. In: *Proc. of the 17th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, *LNCS* 6605, Springer, pp. 311–325.

[12] A. Valmari (1989): *Stubborn sets for reduced state space generation*. In: *Proc. of the 10th Int. Conf. on Application and Theory of Petri Nets*, *LNCS* 483, Springer, pp. 491–515.