Confluence Reduction for Markov Automata

Mark Timmer, Jaco van de Pol, and Mariëlle Stoelinga*

Formal Methods and Tools, Faculty of EEMCS
University of Twente, The Netherlands
{timmer, vdpol, marielle}@cs.utwente.nl

Abstract. Markov automata are a novel formalism for specifying systems exhibiting nondeterminism, probabilistic choices and Markovian rates. Recently, the process algebra MAPA was introduced to efficiently model such systems. As always, the state space explosion threatens the analysability of the models generated by such specifications. We therefore introduce confluence reduction for Markov automata, a powerful reduction technique to keep these models small. We define the notion of confluence directly on Markov automata, and discuss how to syntactically detect confluence on the MAPA language as well. That way, Markov automata generated by MAPA specifications can be reduced on-the-fly while preserving divergence-sensitive branching bisimulation. Three case studies demonstrate the significance of our approach, with reductions in analysis time up to an order of magnitude.

1 Introduction

Over the past two decades, model checking algorithms were generalised to handle more and more expressive models. This now allows us to verify probabilistic as well as hard and soft real-time systems, modelled by timed automata, Markov decision processes, probabilistic automata, continuous-time Markov chains, interactive Markov chains, and Markov automata. Except for timed automata—which incorporate real-time deadlines—all other models are subsumed by the Markov automaton (MA) [14, 13, 12]. MAs can therefore be used as a semantic model for a wide range of formalisms, such as generalised stochastic Petri nets (GSPNs) [2], dynamic fault trees [9], Arcade [8] and the domain-specific language AADL [10].

Before the introduction of MAs, the above models could not be analysed to their full extent. For instance, the semantics of a (potentially nondeterministic) GSPN were given as a fully probabilistic CTMC. To this end, weights had to be assigned to resolve the nondeterminism between immediate transitions. As argued in [20], it is often much more natural to omit most of these weights, retaining rates and probability as well as nondeterminism, and thus obtaining an MA. For example, consider the GSPN in Figure 1(a), taken from [13]. Immediate

* This research has been partially funded by NWO under grants 612.063.817 (SYRUP), 12238 (ArRangeer) and Dn 63-257 (ROCKS), and the EU under grants 318490 (SENSATION) and 318003 (TREsPASS).

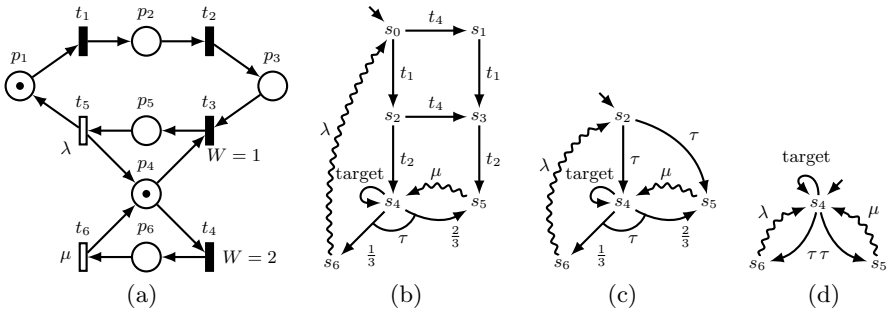


Fig. 1. A GSPN and the corresponding unreduced and reduced state spaces. For the reduced model in (d) the weights of transitions t_3 and t_4 are assumed to be absent.

transitions are indicated in black, Markovian transitions in white, and we assume a partial weight assignment. The underlying MA is given in Figure 1(b), where s_0 corresponds to the initial situation with one token in p_1 and p_4. We added a selfloop labelled *target* to indicate a possible state of interest s_4 (having one token in p_3 and p_4), and for convenience labelled the interactive transitions of the MA by the immediate transition of the GSPN they resulted from (except for the probabilistic transition, which is the result of t_3 and t_4 together).

Recently, the data-rich process-algebraic language MAPA was introduced to efficiently specify MAs in a compositional manner [23]. As always, though, the state space explosion threatens the feasibility of model checking, especially in the presence of data and interleaving. Therefore, reduction techniques for MAs are vital to keep the state spaces of these models manageable. In this paper we introduce such a technique, generalising *confluence reduction* to MAs. It is a powerful state space reduction technique based on commutativity of transitions, removing spurious nondeterminism often arising from the parallel composition of largely independent components. Basically, *confluent transitions* never disable behaviour, since all transitions enabled from their source states can be *mimicked* from their target states. To the best of our knowledge, it is the first technique of this kind for MAs. We give heuristics to apply confluence reduction directly on specifications in the MAPA language, reducing them on-the-fly while preserving divergence-sensitive branching bisimulation.

To illustrate confluence reduction, reconsider the MA in Figure 1(b) and assume that $t_1 = t_2 = t_4 = \tau$, i.e., all action-labelled transitions, except for the *target*-transition, are invisible. We are able to detect automatically that the t_1-transitions are confluent; they can thus safely be given priority over t_4, without losing any behaviour. Figure 1(c) shows the reduced state space, generated on-the-fly using confluence reduction. If all weights are omitted from the specification, an even smaller reduced state space is obtained (Figure 1(d)), while the only change in the unreduced state space is the substitution of the probabilistic choice by a nondeterministic choice.

Outline of the approach. First, we introduce the technical background of our work (Section 2). Then, we define our novel notion of confluence for MAs (Section 3). It specifies sufficient conditions for invisible transitions to not alter the behaviour of an MA; i.e., if a transition is confluent, it could be given priority over all other transitions with the same source state.

We formally show that confluent transitions connect divergence-sensitive branching bisimilar states, and present a mapping of states to representatives to efficiently generate a reduced MA based on confluence (Section 4). We discuss how confluence can be detected symbolically on specifications in the MAPA language (Section 5) and illustrate the significance of our technique using three case studies (Section 6). We show state spaces shrinking by more than 80%, making the entire process from MAPA specification to results more than ten times as fast for some models.[1]

Related work. Confluence reduction for process algebras was first introduced for non-probabilistic systems [7], and later for probabilistic automata [24]. Also, several types of *partial order reduction* (POR) have been defined, both for non-probabilistic [26, 21, 16] and probabilistic systems [11, 4, 3]. These techniques are based on ideas similar to confluence, and have been compared to confluence recently, both in a theoretical [18] and in a practical manner [19]. The results showed that branching-time POR is strictly subsumed by confluence, and that the additional advantages of confluence can be employed nicely in the context of statistical model checking.

Compared to the earlier approaches to confluence reduction for process algebras [7, 24], our novel notion of confluence is different in three important ways:

- It can handle MAs, and hence is applicable to a larger class of systems.
- It fixes a subtle flaw in the earlier papers, which did not guarantee closure under unions. We solve this by introducing an underlying classification of the interactive transitions. This way we do guarantee closure under unions, a key requirement for the way we detect confluence on MAPA specifications.
- It preserves divergences and hence minimal reachability probabilities, incorporating a technique used earlier in [18].

Since none of the existing techniques is able to deal with MAs, we believe that our generalisation—the first reduction technique for MAs abstracting from internal transitions—is a major step forward in efficient quantitative verification.

2 Preliminaries

Definition 1 (Basics). *A* probability distribution *over a countable set S is a function $\mu\colon S \to [0,1]$ such that $\sum_{s\in S} \mu(s) = 1$. For $S' \subseteq S$, let $\mu(S') = \sum_{s\in S'} \mu(s)$. We define* $\mathrm{spt}(\mu) = \{s \in S \mid \mu(s) > 0\}$ *to be the* support *of μ, and write $\mathbb{1}_s$ for the* Dirac distribution *for s, determined by $\mathbb{1}_s(s) = 1$.*

[1] Due to space limitations, we discuss the notion of divergence-sensitive branching bisimulation only on an intuitive level, deferring the formal definitions and proofs of all our results to a technical report [25].

We use $\mathrm{Distr}(S)$ *to denote the set of all probability distributions over* S*, and* $\mathrm{SDistr}(S)$ *for the set of all* substochastic *probability distributions over* S*, i.e., where* $0 \leq \sum_{s \in S} \mu(s) \leq 1$*. Given a function* f*, we denote by* $f(\mu)$ *the* lifting *of* μ *over* f*, i.e.,* $f(\mu)(s) = \mu(f^{-1}(s))$*, with* $f^{-1}(s)$ *the inverse image of* s *under* f*.*

Given an equivalence relation $R \subseteq S \times S$*, we write* $[s]_R$ *for the equivalence class of* s *induced by* R*, i.e.,* $[s]_R = \{s' \in S \mid (s, s') \in R\}$*. Given two probability distributions* $\mu, \mu' \in \mathrm{Distr}(S)$ *and an equivalence relation* R*, we write* $\mu \equiv_R \mu'$ *to denote that* $\mu([s]_R) = \mu'([s]_R)$ *for every* $s \in S$*.*

An MA is a transition system in which the set of transitions is partitioned into probabilistic action-labelled interactive transitions (equivalent to the transitions of a PA), and Markovian transitions labelled by the rate of an exponential distribution (equivalent to the transitions of a CTMC). We assume a countable universe of actions *Act*, with $\tau \in \mathit{Act}$ the invisible internal action.

Definition 2 (Markov automata). *A* Markov automaton (MA) *is a tuple* $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow \rangle$*, where*

- S *is a countable set of* states*, of which* $s^0 \in S$ *is the* initial state*;*
- $A \subseteq \mathit{Act}$ *is a countable set of* actions*;*
- $\hookrightarrow \subseteq S \times A \times \mathrm{Distr}(S)$ *is the* interactive transition relation*;*
- $\rightsquigarrow \subseteq S \times \mathbb{R}_{>0} \times S$ *is the* Markovian transition relation*.*

If $(s, a, \mu) \in \hookrightarrow$*, we write* $s \overset{a}{\hookrightarrow} \mu$ *and say that the action* a *can be* executed *from state* s*, after which the probability to go to each* $s' \in S$ *is* $\mu(s')$*. If* $(s, \lambda, s') \in \rightsquigarrow$*, we write* $s \overset{\lambda}{\rightsquigarrow} s'$ *and say that* s *moves to* s' *with rate* λ*.*

The *rate* between two states $s, s' \in S$ is $\mathit{rate}(s, s') = \sum_{(s,\lambda,s') \in \rightsquigarrow} \lambda$, and the *outgoing rate of* s is $\mathit{rate}(s) = \sum_{s' \in S} \mathit{rate}(s, s')$. We require $\mathit{rate}(s) < \infty$ for every state $s \in S$. If $\mathit{rate}(s) > 0$, the *branching probability distribution* after this delay is denoted by $\mathbb{P}_s$ and defined by $\mathbb{P}_s(s') = \frac{\mathit{rate}(s,s')}{\mathit{rate}(s)}$ for every $s' \in S$.

By definition of the exponential distribution, the probability of leaving a state s within t time units is given by $1 - e^{-\mathit{rate}(s) \cdot t}$ (given $\mathit{rate}(s) > 0$), after which the next state is chosen according to $\mathbb{P}_s$.

MAs adhere to the *maximal progress assumption*, prescribing τ-transitions to never be delayed. Hence, a state that has at least one outgoing τ-transition can never take a Markovian transition. This fact is captured below in the definition of extended transitions, which is used to provide a uniform manner for dealing with both interactive and Markovian transitions.

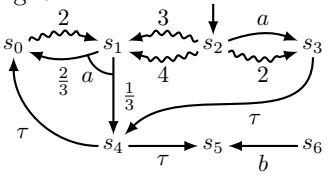
Definition 3 (Extended action set). *Let* $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow \rangle$ *be an MA, then the* extended action set of $\mathcal{M}$ *is given by* $A^\chi = A \cup \{\chi(r) \mid r \in \mathbb{R}_{>0}\}$*. Given a state* $s \in S$ *and an action* $\alpha \in A^\chi$*, we write* $s \overset{\alpha}{\to} \mu$ *if either*

- $\alpha \in A$ *and* $s \overset{\alpha}{\hookrightarrow} \mu$*, or*
- $\alpha = \chi(\mathit{rate}(s))$*,* $\mathit{rate}(s) > 0$*,* $\mu = \mathbb{P}_s$ *and there is no* μ' *such that* $s \overset{\tau}{\hookrightarrow} \mu'$*.*

A transition $s \overset{\alpha}{\to} \mu$ *is called an* extended transition*. We use* $s \overset{\alpha}{\to} t$ *to denote* $s \overset{\alpha}{\to} \mathbb{1}_t$*, and write* $s \to t$ *if there is at least one action* α *such that* $s \overset{\alpha}{\to} t$*. We write* $s \overset{\alpha,\mu}{\longrightarrow} s'$ *if there is an extended transition* $s \overset{\alpha}{\to} \mu$ *such that* $\mu(s') > 0$*.*

Note that each state has an extended transition per interactive transition, while it has only one extended transition for all its Markovian transitions together (if there are any).

Example 4. Consider the MA $\mathcal{M}$ shown on the right.



For this system, $\mathit{rate}(s_2, s_1) = 3+4 = 7$, $\mathit{rate}(s_2) = 7 + 2 = 9$, and $\mathbb{P}_{s_2} = \mu$ such that $\mu(s_1) = \frac{7}{9}$ and $\mu(s_3) = \frac{2}{9}$. There are two extended transitions from s_2: $s_2 \xrightarrow{a} \mathbb{1}_{s_3}$ (also written as $s_2 \xrightarrow{a} s_3$) and $s_2 \xrightarrow{\chi(9)} \mathbb{P}_{s_2}$. □

We define several notions for paths and connectivity. These are based on extended transitions, and thus may contain interactive as well as Markovian steps.

Definition 5 (Paths). *Given an MA $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow \rangle$,*

- *A* path *in $\mathcal{M}$ is a finite sequence $\pi^{\mathrm{fin}} = s_0 \xrightarrow{a_1,\mu_1} s_1 \xrightarrow{a_2,\mu_2} \ldots \xrightarrow{a_n,\mu_n} s_n$ from some state s_0 to a state s_n ($n \geq 0$), or an infinite sequence $\pi^{\mathrm{inf}} = s_0 \xrightarrow{a_1,\mu_1} s_1 \xrightarrow{a_2,\mu_2} s_2 \xrightarrow{a_3,\mu_3} \ldots$, with $s_i \in S$ for all $0 \leq i \leq n$ and all $0 \leq i$, respectively. We use $\mathit{prefix}(\pi, i)$ to denote $s_0 \xrightarrow{a_1,\mu_1} \ldots \xrightarrow{a_i,\mu_i} s_i$, and $\mathit{step}(\pi, i)$ for the transition $s_{i-1} \xrightarrow{a_i} \mu_i$. When π is finite we define $|\pi| = n$ and $\mathit{last}(\pi) = s_n$. We use $\mathit{finpaths}_{\mathcal{M}}$ for the set of all* finite paths *in $\mathcal{M}$ (not necessarily starting in the initial state s^0), and $\mathit{finpaths}_{\mathcal{M}}(s)$ for all such paths with $s_0 = s$.*
- *We denote by $\mathit{trace}(\pi)$ the sequence of actions of π while omitting all τ-actions, and use ϵ to denote the empty sequence.*

Definition 6 (Connectivity). *Let $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow \rangle$ be an MA, $s, t \in S$, and consider again the binary relation $\rightarrow \subseteq S \times S$ from Definition 3 that relates states $s, t \in S$ if there is a transition $s \xrightarrow{\alpha} \mathbb{1}_t$ for some α.*

We let $\twoheadrightarrow$ (reachability) *be the reflexive and transitive closure of $\rightarrow$, and we let $\twoheadleftarrow\!\!\twoheadrightarrow$* (convertibility) *be its reflexive, transitive and symmetric closure. We write $s \twoheadrightarrow \twoheadleftarrow t$* (joinability) *if there is a state u such that $s \twoheadrightarrow u$ and $t \twoheadrightarrow u$.*

Note that the relation $\twoheadrightarrow\!\!\twoheadleftarrow$ is symmetric, but not necessarily transitive. Also note that, intuitively, $s \twoheadleftarrow\!\!\twoheadrightarrow t$ means that s is connected by extended transitions to t—disregarding the orientation of these transitions, but requiring them all to have a Dirac distribution.

Clearly, $s \twoheadrightarrow t$ implies $s \twoheadrightarrow\!\!\twoheadleftarrow t$, and $s \twoheadrightarrow\!\!\twoheadleftarrow t$ implies $s \twoheadleftarrow\!\!\twoheadrightarrow t$. These implications do not hold the other way.

Example 7. The system in Example 4 has infinitely many paths, for example

$$\pi = s_2 \xrightarrow{\chi(9),\mu_1} s_1 \xrightarrow{a,\mu_2} s_0 \xrightarrow{\chi(2),\mathbb{1}_{s_1}} s_1 \xrightarrow{a,\mu_2} s_4 \xrightarrow{\tau,\mathbb{1}_{s_5}} s_5$$

with $\mu_1(s_1) = \frac{7}{9}$ and $\mu_1(s_3) = \frac{2}{9}$, and $\mu_2(s_0) = \frac{2}{3}$ and $\mu_2(s_4) = \frac{1}{3}$. We have $\mathit{prefix}(\pi, 2) = s_2 \xrightarrow{\chi(9),\mu_1} s_1 \xrightarrow{a,\mu_2} s_0$, and $\mathit{step}(\pi, 2) = s_1 \xrightarrow{a} \mu_2$. Also, $\mathit{trace}(\pi) = \chi(9)\, a\, \chi(2)\, a$. It is easy to see that $s_2 \twoheadrightarrow s_5$ (via s_3), as well as $s_3 \twoheadrightarrow\!\!\twoheadleftarrow s_6$ (at s_5) and $s_0 \twoheadleftarrow\!\!\twoheadrightarrow s_5$. However, $s_0 \twoheadrightarrow s_5$ and $s_0 \twoheadrightarrow\!\!\twoheadleftarrow s_5$ do not hold. □

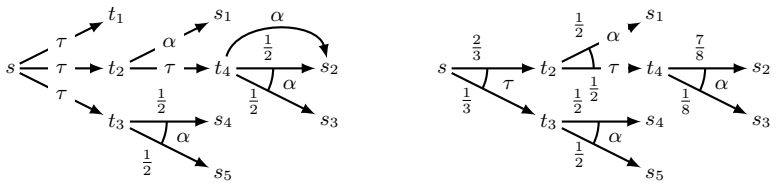


Fig. 2. An MA (left), and a tree demonstrating branching transition $s \xRightarrow{\alpha}_R \mu$ (right).

2.1 Divergence-sensitive branching bisimulation

To prove our confluence reduction technique correct, we show that it preserves divergence-sensitive branching bisimulation. Basically, this means that there is an equivalence relation R linking states in the original system to states in the reduced system, in such a way that their initial states are related and all related states can mimic each other's transitions and divergences.

More precisely, for R to be a divergence-sensitive branching bisimulation, it is required that for all $(s,t) \in R$ and every extended transition $s \xrightarrow{\alpha} \mu$, there is a *branching transition* $t \xRightarrow{\alpha}_R \mu'$ such that $\mu \equiv_R \mu'$. The existence of such a branching transition depends on the existence of a certain *scheduler*. Schedulers resolve nondeterministic choices in an MA by selecting which transitions to take given a history; they are also allowed to terminate with some probability.

Now, a state t can do a branching transition $t \xRightarrow{\alpha}_R \mu'$ if either (1) $\alpha = \tau$ and $\mu' = \mathbb{1}_t$, or (2) there exists a scheduler that terminates according to μ', always schedules precisely one α-transition (immediately before terminating), does not schedule any other visible transitions and does not leave the equivalence class $[t]_R$ before doing an a-transition.

Example 8. Observe the MA in Figure 2 (left). We find that $s \xRightarrow{\alpha}_R \mu$, with

$$\mu(s_1) = \tfrac{8}{24} \qquad \mu(s_2) = \tfrac{7}{24} \qquad \mu(s_3) = \tfrac{1}{24} \qquad \mu(s_4) = \tfrac{4}{24} \qquad \mu(s_5) = \tfrac{4}{24}$$

by the scheduling depicted in Figure 2 (right), assuming $(s, t_i) \in R$ for all t_i. □

In addition to the mimicking of transitions by branching transitions, we require R-related states to either both be able to perform an infinite invisible path with probability 1 (*diverge)*, or to both not be able to do so. We write $s \leftrightarrows_{\mathrm{b}}^{\mathrm{div}} t$ if two states s, t are divergence-sensitive branching bisimilar, and $\mathcal{M}_1 \leftrightarrows_{\mathrm{b}}^{\mathrm{div}} \mathcal{M}_2$ if two MAs are (i.e., if their initial states are so in their disjoint union).

3 Confluence for Markov automata

In [24] we defined three variants of probabilistic confluence: weak probabilistic confluence, probabilistic confluence and strong probabilistic confluence. They specify sufficient conditions for τ-transitions to not alter the behaviour of an MA. The stronger notions are easier to detect, but less powerful in their reductions.

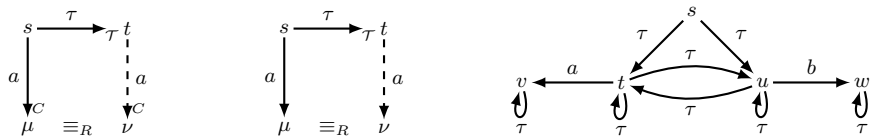
In a process-algebraic context, where confluence is detected heuristically over a syntactic description of a system, it is most practical to apply strong confluence. Therefore, in this paper we only generalise strong probabilistic confluence to the Markovian realm. Although MAs in addition to interactive transitions may also contain Markovian transitions, these are basically irrelevant for confluence. After all, states having a τ-transition can never execute a Markovian transition due to the maximal progress assumption. Hence, such transitions need not be mimicked. For the above reasons, the original definition of confluence for PAs might seem to still work for MAs. This is not true, however, for two reasons.

1. The old definition was not yet divergence sensitive. Therefore, Markovian transitions in an MA that are disabled by the maximal progress assumption, due to a divergence from the same state, may erroneously be enabled if that divergence is removed. Hence, the old notion does not even preserve Markovian divergence-*in*sensitive branching bisimulation. We now improve on the definition to resolve this issue, introducing τ-loops in the reduced system for states having confluent divergence in the original system (inspired by the way [18] deals with divergences). This not only makes the theory work for MAs, it even yields preservation of divergence-sensitive branching bisimulation, and hence of minimal reachability probabilities.
2. The old definition had a subtle flaw: earlier work relied on the assumption that confluent sets are closed under unions [7, 24]. In practical applications this was indeed a valid assumption, but for the theoretical notions of confluence this was not yet the case. We fix this flaw by classifying transitions into groups, defining confluence over sets of such groups and requiring transitions to be mimicked by a transition from their own group.

Additionally, compared to [7, 24] we improve on the way equivalence of distributions is defined, making it slightly more powerful and, in our view, easier to understand (inspired by the definitions in [19]).

Confluence classifications and confluent sets. The original lack of closure under unions was due to the requirement that confluent transitions are mimicked by confluent transitions. When taking the union of two sets of confluent transitions, this requirement was possibly invalidated. To solve this problem, we classify the interactive transitions of an MA into groups—allowing overlap and not requiring all interactive transitions to be in at least one group. Together, we call such a set of groups $P = \{C_1, C_2, \ldots, C_n\} \subseteq \mathscr{P}(\hookrightarrow)$ a *confluence classification*[2]. Now, instead of designating individual transitions to be confluent and requiring confluent transitions to be mimicked by confluent transitions, we designate groups in P to be confluent (now called *Markovian confluent*) and require transitions from a group in P to be mimicked by transitions from the same group.

[2] We use $s \xrightarrow{a}_C \mu$ to denote that $(s \xrightarrow{a} \mu) \in C$, and abuse notation by writing $(s \xrightarrow{a} \mu) \in P$ to denote that $s \xrightarrow{a}_C \mu$ for some $C \in P$. Similarly, we subscript reachability, joinability and convertibility arrows to indicate that they only traverse transitions from a certain group or set of groups of transitions.



(a) $(s \xrightarrow{a} \mu) \in P$. (b) $(s \xrightarrow{a} \mu) \notin P$. (c) A simple state space.

Fig. 3. The confluence diagrams for $s \xrightarrow{\tau}_{\mathcal{T}} t$, and a simple state space. In (a,b): If the solid transitions are present, then so should the dashed ones be.

For a set $\mathcal{T} \subseteq P$ to be Markovian confluent, first of all—like in the PA setting [24, 3]—it is only allowed to contain invisible transitions with a Dirac distribution. (Still, giving priority to such transitions may very well reduce probabilistic transitions as well, as we will see in Section 4.) Additionally, each transition $s \xrightarrow{a} \mu$ enabled before a transition $s \xrightarrow{\tau}_{\mathcal{T}} t$ should have a mimicking transition $t \xrightarrow{a} \nu$ such that μ and ν are connected by $\mathcal{T}$-transitions, and mimicking transitions should be from the same group. The definition is illustrated in Figure 3.

Definition 9 (Markovian confluence). *Let $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow \rangle$ be an MA and $P \subseteq \mathscr{P}(\hookrightarrow)$ a confluence classification. Then, a set $\mathcal{T} \subseteq P$ is* Markovian confluent for P *if it only contains sets of invisible transitions with Dirac distributions, and for all $s \xrightarrow{\tau}_{\mathcal{T}} t$ and all transitions $(s \xrightarrow{a} \mu) \neq (s \xrightarrow{\tau} t)$:*

$$\begin{cases} \forall C \in P \,.\, s \xrightarrow{a}_C \mu \implies \exists \nu \in \mathrm{Distr}(S) \,.\, t \xrightarrow{a}_C \nu \wedge \mu \equiv_R \nu & , \text{if } (s \xrightarrow{a} \mu) \in P \\ \qquad\qquad\qquad\quad\; \exists \nu \in \mathrm{Distr}(S) \,.\, t \xrightarrow{a} \nu \wedge \mu \equiv_R \nu & , \text{if } (s \xrightarrow{a} \mu) \notin P \end{cases}$$

with R the smallest equivalence relation such that

$$R \supseteq \{(s,t) \in \mathrm{spt}(\mu) \times \mathrm{spt}(\nu) \mid (s \xrightarrow{\tau} t) \in \mathcal{T}\}.$$

A transition $s \xrightarrow{\tau} t$ is Markovian confluent *if there exists a Markovian confluent set $\mathcal{T}$ such that $s \xrightarrow{\tau}_{\mathcal{T}} t$. Often, we omit the adjective 'Markovian'.*

Note that $\mu \equiv_R \nu$ requires direct transitions from the support of μ to the support of ν. Also note that, even though a (symmetric) equivalence relation R is used, transitions from the support of ν to the support of μ do not influence R.

Remark 10. Due to the confluence classification, confluent transitions are always mimicked by confluent transitions. After all, transitions from a group $C \in P$ are mimicked by transitions from C. So, if C is designated confluent by $\mathcal{T}$, then all these confluent transitions are indeed mimicked by confluent transitions.

Although the confluence classification may appear restrictive, we will see that in practice it is obtained naturally. Transitions are often instantiations of higher-level constructs, and are therefore easily grouped together. Then, it makes sense to detect the confluence of such a higher-level construct. Additionally, to show that a certain set of invisible transitions is confluent, we can just take P to consist of one group containing precisely all those transitions. Then, the requirement for P-transitions to be mimicked by the same group reduces to the old requirement that confluent transitions are mimicked by confluent transitions.

Properties of confluent sets. Since confluent transitions are always mimicked by confluent transitions, confluent paths (i.e., paths following only transitions from a confluent set) are always joinable.

Proposition 11. *Let $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow \rangle$ be an MA, $P \subseteq \mathscr{P}(\hookrightarrow)$ a confluence classification for $\mathcal{M}$ and $\mathcal{T} \subseteq P$ a Markovian confluent set for P. Then,*

$$s \twoheadrightarrow \twoheadleftarrow_{\mathcal{T}} t \qquad \text{if and only if} \qquad s \twoheadleftarrow\!\!\!\!\rightarrow_{\mathcal{T}} t$$

Due to the confluence classification, we now also do have a closure result. Closure under union tells us that it is safe to show confluence of multiple sets of transitions in isolation, and then just take their union as one confluent set. Also, it implies that there exists a unique maximal confluent set.

Theorem 12. *Let $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow \rangle$ be an MA, $P \subseteq \mathscr{P}(\hookrightarrow)$ a confluence classification for $\mathcal{M}$ and $\mathcal{T}_1, \mathcal{T}_2 \subseteq P$ two Markovian confluent sets for P. Then, $\mathcal{T}_1 \cup \mathcal{T}_2$ is also a Markovian confluent set for P.*

The next example shows why Theorem 12 would not hold without the use of a confluence classification. It applies to the old notions of confluence as well.

Example 13. Consider the system in Figure 3(c). Without the requirement that transitions are mimicked by the same group, the sets

$$\mathcal{T}_1 = \{(s, \tau, u), (t, \tau, t), (u, \tau, u), (v, \tau, v), (w, \tau, w)\}$$
$$\mathcal{T}_2 = \{(s, \tau, t), (t, \tau, t), (u, \tau, u), (v, \tau, v), (w, \tau, w)\}$$

would both be perfectly valid confluent sets. Still, $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$ is not an acceptable set. After all, whereas $t \twoheadleftarrow\!\!\!\!\rightarrow_{\mathcal{T}} u$, it fails to satisfy $t \twoheadrightarrow \twoheadleftarrow_{\mathcal{T}} u$. This property was ascertained in earlier work by requiring confluent transitions to be mimicked by confluent transitions or by explicitly requiring $\twoheadrightarrow \twoheadleftarrow_{\mathcal{T}}$ to be an equivalence relation. This is indeed not the case for $\mathcal{T}$, as the diamond starting with $s \xrightarrow{\tau} t$ and $s \xrightarrow{\tau} u$ can only be closed using the non-confluent transitions between t and u, and clearly $\twoheadrightarrow \twoheadleftarrow$ is not transitive. However, $\mathcal{T}_1$ and $\mathcal{T}_2$ do satisfy these requirements, and hence the old notions were not closed under union.

By using a confluence classification and requiring transitions to be mimicked by the same group, we ascertain that this kind of bad compositionality behaviour does not occur. After all, for $\mathcal{T}_1$ to be a valid confluent set, the confluence classification should be such that $s \xrightarrow{\tau} t$ and its mimicking transition $u \xrightarrow{\tau} t$ are in the same group. So, for $s \xrightarrow{\tau} t$ to be confluent (as prescribed by $\mathcal{T}_2$), also $u \xrightarrow{\tau} t$ would need to be confluent. The latter is impossible, since the b-transition from u cannot be mimicked from t, and hence $\mathcal{T}_2$ is disallowed. □

The final result of this section shows that confluent transitions indeed connect divergence-sensitive bisimilar states. This is a key result; it implies that confluent transitions can be given priority over other transitions without losing behaviour—when being careful not to indefinitely ignore any behaviour.

Theorem 14. *Let $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow \rangle$ be an MA, $s, s' \in S$ two of its states, $P \subseteq \mathscr{P}(\hookrightarrow)$ a confluence classification for $\mathcal{M}$ and $\mathcal{T} \subseteq P$ a Markovian confluent set for P. Then,*

$$s \twoheadleftarrow\!\!\!\!\rightarrow_{\mathcal{T}} s' \text{ implies } s \leftrightarrows^{\mathrm{div}}_{\mathrm{b}} s'.$$

4 State space reduction using confluence

We can reduce state spaces by giving priority to confluent transitions, i.e., by omitting all other transitions from a state that also enables a confluent transition (as long as no behaviour is ignored indefinitely). Better still, we aim at omitting all intermediate states on a confluent path altogether; after all, they are all bisimilar anyway by Theorem 14. Confluence even dictates that all visible transitions and divergences enabled from a state s can directly be mimicked from another state t if $s \twoheadrightarrow_{\mathcal{T}} t$. Hence, we can just keep following a confluent path and only retain the last state. To avoid getting stuck in an infinite confluent loop, we detect entering a bottom strongly connected component (BSCC) of confluent transitions and choose a unique *representative* from this BSCC for all states that can reach it. Since we showed that confluent joinability is transitive (as implied by Proposition 11), it follows immediately that all confluent paths emanating from a certain state s always end up in a unique BSCC.

Formally, we use the notion of a *representation map*, assigning a representative state $\varphi(s)$ to every state s. We make sure that $\varphi(s)$ indeed exhibits all behaviour of s due to being in a BSCC reachable from s.

Definition 15 (Representation map). *Let $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow\rangle$ be an MA and $\mathcal{T}$ a Markovian confluent set for $\mathcal{M}$. Then, a function $\varphi_{\mathcal{T}} \colon S \to S$ is a* representation map *for $\mathcal{M}$ under $\mathcal{T}$ if for all $s, s' \in S$*

- $s \twoheadrightarrow_{\mathcal{T}} \varphi_{\mathcal{T}}(s)$
- $s \to_{\mathcal{T}} s' \implies \varphi_{\mathcal{T}}(s) = \varphi_{\mathcal{T}}(s')$

Note that the first requirement ensures that every representative is reachable by all states it represents, while the second takes care that all $\mathcal{T}$-related states have the same representative. Together, they imply that every representative is in a BSCC. Since all $\mathcal{T}$-related states have the same BSCC, as discussed above, it is indeed always possible to find a representation map. We refer to [6] for the algorithm we use to construct it in our implementation.

As representatives exhibit all behaviour of the states they represent, they can be used for state space reduction. More precisely, it is possible to define the quotient of an MA modulo a representation map. This system does not have any $\mathcal{T}$-transitions anymore, except for self-loops on representatives that have outgoing $\mathcal{T}$-transitions in the original system. These ensure preservation of divergences.

Definition 16 (Quotient). *Given an MA $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow\rangle$, a confluent set $\mathcal{T}$ for $\mathcal{M}$, and a representation map $\varphi \colon S \to S$ for $\mathcal{M}$ under $\mathcal{T}$, the* quotient of $\mathcal{M}$ modulo φ *is the smallest system $\mathcal{M}/\varphi = \langle \varphi(S), \varphi(s^0), A, \hookrightarrow_{\varphi}, \rightsquigarrow_{\varphi}\rangle$ such that*

- $\varphi(S) = \{\varphi(s) \mid s \in S\}$;
- $\varphi(s) \overset{a}{\hookrightarrow}_{\varphi} \varphi(\mu)$ *if* $\varphi(s) \overset{a}{\hookrightarrow} \mu$;
- $\varphi(s) \overset{\lambda}{\rightsquigarrow}_{\varphi} \varphi(s')$ *if* $\lambda = \sum_{\lambda' \in \Lambda(s,s')} \lambda'$ *and* $\lambda > 0$,

where $\Lambda(s, s')$ is the multiset $\{\!|\lambda' \in \mathbb{R} \mid \exists s^* \in S \,.\, \varphi(s) \overset{\lambda'}{\rightsquigarrow} s^* \wedge \varphi(s^*) = \varphi(s')|\!\}$.

Note that each interactive transition from $\varphi(s)$ in $\mathcal{M}$ is lifted to $\mathcal{M}/\varphi$ by changing all states in the support of its target distribution to their representatives. Additionally, each pair $\varphi(s), \varphi(s')$ of representative states in $\mathcal{M}/\varphi$ has a connecting Markovian transition with rate equal to the total outgoing rate of $\varphi(s)$ in $\mathcal{M}$ to states s^* that have $\varphi(s')$ as their representative (unless this sum is 0). It is easy to see that this implies $\varphi(s) \xrightarrow{\chi(\lambda)}_{\varphi} \varphi(\mu)$ if and only if $\varphi(s) \xrightarrow{\chi(\lambda)} \mu$.

Since $\mathcal{T}$-transitions connect bisimilar states, and representatives exhibit all behaviour of the states they represent, we can prove the following theorem. It shows that we indeed reached our goal of providing a reduction that is safe with respect to divergence-sensitive branching bisimulation.

Theorem 17. *Let $\mathcal{M} = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow \rangle$ be an MA, $\mathcal{T}$ a Markovian confluent set for $\mathcal{M}$, and $\varphi\colon S \to S$ a representation map for $\mathcal{M}$ under $\mathcal{T}$. Then,*

$$\mathcal{M}/\varphi \leftrightarrows_{\mathrm{b}}^{\mathrm{div}} \mathcal{M}.$$

5 Symbolic detection of Markovian confluence

Although the definition of confluence in Section 3 is useful to show the correctness of our approach, it is often not feasible to check in practice. After all, we want to reduce *on-the-fly* to obtain a smaller state space without first generating the unreduced one. Therefore, we use heuristics to detect Markovian confluence in the context of the process-algebraic modelling language MAPA [23]. As these heuristics only differ slightly from the ones in [24] for probabilistic confluence, we discuss the basics and explain how the old techniques can be reused.

MAPA is data-rich and expressive, and features a restricted form: the Markovian Linear Probabilistic Process Equation (MLPPE). Every MAPA specification can be translated easily to an equivalent specification in MLPPE [23]. Hence, it suffices to define our confluence-based reduction technique on this form.

The MLPPE format. An MLPPE is a process with global variables, *interactive summands* (each yielding a set of interactive transitions) and *Markovian summands* (each yielding a set of Markovian transitions). Its semantics is given as an MA, whose states are valuations of the global variables. Basically, in each state a nondeterministic choice is made between the summands that are enabled given these values.

Each interactive summand has a condition (the guard) that specifies for which valuations of the global variables it is enabled. If so, an action can be taken and the next state (a new valuation for the global variables) is determined probabilistically. The action and next state may also depend on the current state. The Markovian summands are similar, except that they contain a rate and a unique next state instead of an action and a probabilistic next state. We assume an implicit confluence classification $P = \{C_1, \ldots, C_k\}$ that, for each interactive summand, contains a group consisting of all transitions generated by that summand. We note that this classification is only given for theoretical reasons; it is not actually constructed.

For a precise formalisation of the language and its semantics, we refer to [23].

Confluent summands. We check for *confluent summands*: summands that are guaranteed to *only* yield confluent transitions, i.e., summands i such that the set $\mathcal{T} = \{C_i\}$ is confluent. Whenever during state space generation such a summand is enabled, all other summands can be ignored (continuing until reaching a representative in a BSCC, as explained in the previous section). By Theorem 12, the union of all confluent summands is also confluent.

Since only τ-transitions can be confluent, the only summands that might be confluent are interactive summands having action τ for all valuations of the global variables. Also, the next state of each of the transitions they generate should be unique. Finally, we verify whether all transitions that may result from these summands commute with all other transitions according to Definition 9.

We only need to check commutativity with all transitions possibly generated by the interactive summands, as the Markovian summands are never enabled at the same time as an invisible transition due to the maximal progress assumption. We overapproximate commutativity by checking whether, when two summands are enabled, they do not disable each other and do not influence each other's actions, probabilities and next states. After all, that implies that each transition can be mimicked by a transition from the same summand (and hence also that it is indeed mimicked by the same group of P). This can be formally expressed as a logical formula (see [24] for the details). Such a formula can be checked by an SMT solver, or approximated using heuristics. We implemented basic heuristics, checking mainly whether two summands are never enabled at the same time or whether the variables updated by one are not used by the other and vice versa. Additionally, some laws from the natural numbers have been implemented, taking for instance into account that $x := x + 1$ cannot disable $x > 2$. In the future, we hope to extend this to more advanced theorem proving.

6 Case studies

We implemented confluence reduction in our tool SCOOP [22]. It takes MAPA specifications as input, is able to perform several reduction techniques and can generate state spaces in multiple formats, among which the one for the IMCA tool for model checking MAs [17]. We already showed in [23] the benefits of dead variable reduction. Here, we apply only confluence reduction, to focus on the power of our novel technique. We present the size of the state spaces with and without confluence reduction, as well as the time to generate them with SCOOP and to subsequently analyse them with IMCA. That way, the impact of confluence reduction on both MA generation and analysis becomes clear[3].

We conjecture that the (quantitative) behavioural equivalence induced by branching bisimulation leaves invariant the time-bounded reachability probabilities, expected times to reachability and long-run averages computed by IMCA. This indeed turned out to be the case for all our models. A logic precisely characterising Markovian branching bisimulation would be interesting future work.

[3] The tool (for download and web-based usage [5]), all MAPA models and a test script can be found on `http://fmt.cs.utwente.nl/~timmer/scoop/papers/formats`.

Specification	Original state space States	Trans.	SCOOP	IMCA	Reduced state space States	Trans.	SCOOP	IMCA	Impact States	Time
`leader-3-7`	25,505	34,257	4.7	102.5	5,564	6,819	5.1	9.3	-78%	-87%
`leader-3-9`	52,465	71,034	9.7	212.0	11,058	13,661	10.4	17.8	-79%	-87%
`leader-3-11`	93,801	127,683	18.0	429.3	19,344	24,043	19.2	31.9	-79%	-89%
`leader-4-2`	8,467	11,600	2.1	74.0	2,204	2,859	2.5	6.8	-74%	-88%
`leader-4-3`	35,468	50,612	9.0	363.8	7,876	10,352	8.7	33.3	-78%	-89%
`leader-4-4`	101,261	148,024	25.8	1,309.8	20,857	28,023	24.3	94.4	-79%	-91%
`polling-2-2-4`	4,811	8,578	0.7	3.7	3,047	6,814	0.7	2.3	-37%	-32%
`polling-2-2-6`	27,651	51,098	12.7	91.0	16,557	40,004	5.4	49.0	-40%	-48%
`polling-2-4-2`	6,667	11,290	0.9	39.9	4,745	9,368	0.9	26.6	-29%	-33%
`polling-2-5-2`	27,659	47,130	4.0	1,571.7	19,721	39,192	4.0	1,054.6	-29%	-33%
`polling-3-2-2`	2,600	4,909	0.4	7.1	1,914	4,223	0.5	4.8	-26%	-29%
`polling-4-6-1`	15,439	29,506	3.1	330.4	4,802	18,869	3.0	109.4	-69%	-66%
`polling-5-4-1`	21,880	43,760	5.1	815.9	6,250	28,130	5.1	318.3	-71%	-61%
`processor-2`	2,508	4,608	0.7	2.8	1,514	3,043	0.8	1.2	-44%	-43%
`processor-3`	10,852	20,872	3.1	66.3	6,509	13,738	3.3	23.0	-45%	-62%
`processor-4`	31,832	62,356	10.8	924.5	19,025	41,018	10.3	365.6	-45%	-60%

Table 1. State space generation and analysis using confluence reduction (on a 2.4 GHz 4 GB Intel Core 2 Duo MacBook). Runtimes in SCOOP and IMCA are in seconds.

Leader election protocol. The first case study is a leader election protocol (Algorithm $\mathcal{B}$ from [15]), used in [24] as well to demonstrate confluence reduction for probabilistic automata. It uses asynchronous channels and allows for multiple nodes, throwing dice to break the symmetry. We added a rate 1 to a node throwing a die to get an MA model based on the original case study, making the example more relevant and interesting in the current situation. We computed the minimal probability (with error bound 0.01) of electing the first node as leader within 5 time units. The results are presented in Table 1, where we denote by `leader-i-j` the variant with `i` nodes and `j`-sided dice. The computed probability varies from 0.09 for `leader-4-2` to 0.32 for `leader-3-11`. Confluence saved almost 90% of the total time to generate and analyse the models. The substantial reductions are due to extensive interleaving with little communication.

Queueing system. The second case study is the queueing system from [23]. It consists of multiple stations with incoming jobs, and one server that polls the stations for work. With some probability, communication fails. There can be different sizes of buffers in the stations, and multiple types of jobs with different service rates. In Table 1, we let `polling-i-j-k` denote the variant with `i` stations, all having buffers of size `j` and `k` types of jobs. Note that, although significant reductions are obtained, the reduction in states precisely corresponds to the reduction in transitions; this implies that only trivially confluent transitions could be reduced (i.e., invisible transitions without any other transitions from the same source state). We computed the minimal and maximal expected time to the situation that all buffers are full. This turns out to be at least 1.1—for `polling-3-2-2`—and at most 124—for `polling-2-5-2`. Reductions were less substantial, due to the presence of many probabilistic and Markovian transitions.

Processor architecture. The third case study is a GSPN model of a 2×2 concurrent processor architecture, parameterised in the level k of multitasking, taken from Figure 11.7 in [1]. We constructed a corresponding MAPA model, modelling each place as a global variable and each transition as a summand. As in [1], we

computed the throughput of one of the processors, given by the long-run average of having a token in a certain place of the GSPN. Whereas [1] resolved all nondeterminism and found for instance a throughput of 0.903 for $k = 2$, we can retain the nondeterminism and obtain the more informative interval $[0.811, 0.995]$. (When resolving nondeterminism as before, we reproduce the result 0.903.)

Our results clearly show the significant effect of confluence reduction on the state space sizes and the duration of the heavy numerical computations by IMCA. The generation times by SCOOP are not reduced as much, due to the additional overhead of computing representative states. To keep memory usage in the order of the reduced state space, the representative map is deliberately not stored and therefore potentially recomputed for some states.

7 Conclusions

We introduced confluence reduction for MAs: the first reduction technique for this model that abstracts from invisible transitions. We showed that it preserves divergence-sensitive branching bisimulation, and hence yields quantitatively behavioural equivalent models. In addition to working on MAs, our novel notion of confluence reduction has two additional advantages over previous notions. First, it preserves divergences, and hence does not alter minimal reachability probabilities. Second, it is closed under unions, enabling us to separately detect confluence of different sets of transitions and combine the results. We also showed that the representation map approach can still be used safely to reduce systems on-the-fly, and discussed how to detect confluence syntactically on the process-algebraic language MAPA. Case studies with our tool SCOOP on several instances of three different models show state space reductions up to 79%. We linked SCOOP to the IMCA model checker to illustrate the significant impact of these reductions on the expected time, time-bounded reachability and long-run average computations. Due to confluence reduction, for some models the entire process from MAPA specification to results is now more than ten times as fast.

As future work we envision to search for even more powerful ways of using commutativity for state space reduction, for instance by allowing confluent transitions to be probabilistic. Preferably, this would enable even more aggressive reductions that, instead of preserving the conservative notion of bisimulation we used, preserve the more powerful weak bisimulation from [14].

Acknowledgements. We thank Stefan Blom and Joost-Pieter Katoen for their useful suggestions, and Dennis Guck for his help with the case studies.

References

[1] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets.* John Wiley & Sons, Inc., 1994.

[2] M. Ajmone Marsan, G. Conte, and G. Balbo. A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems. *ACM Transactions on Computer Systems*, 2(2):93–122, 1984.

[3] C. Baier, P. R. D'Argenio, and M. Größer. Partial order reduction for probabilistic branching time. In *QAPL*, volume 153(2) of *ENTCS*, pages 97–116, 2006.

[4] C. Baier, M. Größer, and F. Ciesinski. Partial order reduction for probabilistic systems. In *QEST*, pages 230–239, 2004.
[5] A. Belinfante and A. Rensink. Publishing your prototype tool on the web: PUPTOL, a framework. Technical Report TR-CTIT-13-15, Centre for Telematics and Information Technology, University of Twente, 2013.
[6] S. C. C. Blom. Partial τ-confluence for efficient state space generation. Technical Report SEN-R0123, CWI, Amsterdam, 2001.
[7] S. C. C. Blom and J. C. van de Pol. State space reduction by proving confluence. In *CAV*, volume 2404 of *LNCS*, pages 596–609, 2002.
[8] H. Boudali, P. Crouzen, B. R. Haverkort, M. Kuntz, and M. I. A. Stoelinga. Architectural dependability evaluation with arcade. In *DSN*, pages 512–521, 2008.
[9] H. Boudali, P. Crouzen, and M. I. A. Stoelinga. A rigorous, compositional, and extensible framework for dynamic fault tree analysis. *IEEE Transactions on Dependable and Secure Compututing*, 7(2):128–143, 2010.
[10] M. Bozzano, A. Cimatti, J.-P. Katoen, V. Y. Nguyen, T. Noll, and M. Roveri. Safety, dependability and performance analysis of extended AADL models. *The Computer Journal*, 54(5):754–775, 2011.
[11] P. R. D'Argenio and P. Niebert. Partial order reduction on concurrent probabilistic programs. In *QEST*, pages 240–249, 2004.
[12] Y. Deng and M. Hennessy. On the semantics of Markov automata. In *ICALP*, volume 6756 of *LNCS*, pages 307–318, 2011.
[13] C. Eisentraut, H. Hermanns, and L. Zhang. Concurrency and composition in a stochastic world. In *CONCUR*, volume 6269 of *LNCS*, pages 21–39, 2010.
[14] C. Eisentraut, H. Hermanns, and L. Zhang. On probabilistic automata in continuous time. In *LICS*, pages 342–351, 2010.
[15] W. Fokkink and J. Pang. Simplifying Itai-Rodeh leader election for anonymous rings. In *AVoCS*, volume 128(6) of *ENTCS*, pages 53–68, 2005.
[16] P. Godefroid. *Partial-order Methods for the Verification of Concurrent Systems: an Approach to the State-explosion Problem*, volume 1032 of *LNCS*. 1996.
[17] D. Guck, H. Hatefi, H. Hermanns, J.-P. Katoen, and M. Timmer. Modelling, reduction and analysis of Markov automata. In *QEST*, LNCS, 2013 (to appear).
[18] H. Hansen and M. Timmer. A comparison of confluence and ample sets in probabilistic and non-probabilistic branching time. *TCS*, 2013 (to appear).
[19] A. Hartmanns and M. Timmer. On-the-fly confluence detection for statistical model checking. In *NFM*, volume 7871 of *LNCS*, pages 337–351, 2013.
[20] J.-P. Katoen. GSPNs revisited: Simple semantics and new analysis algorithms. In *ACSD*, pages 6–11, 2012.
[21] D. Peled. All from one, one for all: on model checking using representatives. In *CAV*, volume 697 of *LNCS*, pages 409–423, 1993.
[22] M. Timmer. SCOOP: A tool for symbolic optimisations of probabilistic processes. In *QEST*, pages 149–150, 2011.
[23] M. Timmer, J.-P. Katoen, J. C. van de Pol, and M. I. A. Stoelinga. Efficient modelling and generation of Markov automata. In *CONCUR*, volume 7454 of *LNCS*, pages 364–379, 2012.
[24] M. Timmer, M. I. A. Stoelinga, and J. C. van de Pol. Confluence reduction for probabilistic systems. In *TACAS*, volume 6605 of *LNCS*, pages 311–325, 2011.
[25] M. Timmer, J. C. van de Pol, and M. I. A. Stoelinga. Confluence reduction for Markov automata (extended version). Technical Report TR-CTIT-13-14, Centre for Telematics and Information Technology, University of Twente, 2013.
[26] A. Valmari. Stubborn sets for reduced state space generation. In *APN*, volume 483 of *LNCS*, pages 491–515, 1989.