Confluence Reduction for Probabilistic Systems (extended version)

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

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

Abstract. This paper presents a novel technique for state space reduction of probabilistic specifications, based on a newly developed notion of confluence for probabilistic automata. We prove that this reduction preserves branching probabilistic bisimulation and can be applied on-the-fly. To support the technique, we introduce a method for detecting confluent transitions in the context of a probabilistic process algebra with data, facilitated by an earlier defined linear format. A case study demonstrates that significant reductions can be obtained.

1 Introduction

Model checking of probabilistic systems is getting more and more attention, but there still is a large gap between the number of techniques supporting traditional model checking and those supporting probabilistic model checking. Especially methods aimed at reducing state spaces are greatly needed to battle the omnipresent state space explosion.

In this paper, we generalise the notion of confluence [10] from labelled transition systems (LTSs) to probabilistic automata (PAs) [14]. Basically, we define under which conditions unobservable transitions (often called τ-transitions) do not influence a PA's behaviour (i.e., they commute with all other transitions). Using this new notion of probabilistic confluence, we introduce a symbolic technique that reduces PAs while preserving branching probabilistic bisimulation.

The non-probabilistic case. Our methodology follows the approach for LTSs from [4]. It consists of the following steps: (i) a system is specified as the parallel composition of several processes with data; (ii) the specification is linearised to a canonical form that facilitates symbolic manipulations; (iii) first-order logic formulas are generated to check symbolically which τ-transitions are confluent; (iv) an LTS is generated in such a way that confluent τ-transitions are given priority, leading to an on-the-fly (potentially exponential) state space reduction. Refinements by [12] make it even possible to perform confluence detection on-the-fly by means of boolean equation systems.

The probabilistic case. After recalling some basic concepts from probability theory and probabilistic automata, we introduce three novel notions of probabilistic

* 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).

confluence. Inspired by [3], these are *weak probabilistic confluence*, *probabilistic confluence* and *strong probabilistic confluence* (in decreasing order of reduction power, but in increasing order of detection efficiency).

We prove that the stronger notions imply the weaker ones, and that τ-transitions that are confluent according to any of these notions always connect branching probabilistically bisimilar states. Basically, this means that they can be given priority without losing any behaviour. Based on this idea, we propose a reduction technique using weak probabilistic confluence, which merges all states that can reach each other by traversing only confluent transitions. Additionally, we propose a reduction technique that can be applied using the two stronger notions of confluence. As opposed to the first technique it does not need to merge states; rather, it chooses a representative state that has all relevant behaviour. We prove that both reduction techniques yield a branching probabilistically bisimilar PA. Therefore, they preserve virtually all interesting temporal properties.

As we want to analyse systems that would normally be too large, we need to detect confluence symbolically and use it to reduce on-the-fly during state space generation. That way, the unreduced PA never needs to be generated. Since we have not found an efficient method for detecting (weak) probabilistic confluence, we only provide a detection method for strong probabilistic confluence. Here, we exploit a previously defined probabilistic process-algebraic linear format, which is capable of modelling any system consisting of parallel components with data [9]. In this paper, we show how symbolic τ-transitions can be proven confluent by solving formulas in first-order logic over this format. As a result, confluence can be detected symbolically, and the reduced PA can be generated on-the-fly. We present a case study of leader election protocols, showing significant reductions.

Related work. As mentioned before, we basically generalise the techniques presented in [4] to probabilistic automata.

In the probabilistic setting, several reduction techniques similar to ours exist. Most of these are generalisations of the well-known concept of partial-order reduction (POR) [13]. In [2] and [5], the concept of POR was lifted to Markov decision processes, providing reductions that preserve quantitative LTL\X. This was refined in [1] to probabilistic CTL, a branching logic. Recently, a revision of POR for distributed schedulers was introduced and implemented in PRISM [7].

Our confluence reduction differs from these techniques on several accounts. First, POR is applicable on state-based systems, whereas our confluence reduction is the first technique that can be used for action-based systems. As the transformation between action- and state-based blows up the state space [11], having confluence reduction really provides new possibilities. Second, the definition of confluence is quite elegant, and (strong) confluence seems to be of a more local nature (which makes the correctness proofs easier). Third, the detection of POR requires language-specific heuristics, whereas confluence reduction acts at a more semantic level and can be implemented by a generic theorem prover. (Alternatively, decision procedures for a fixed set of data types could be devised.)

Our case study shows that the reductions obtained using probabilistic confluence are comparable to the reductions obtained by POR [8].

2 Preliminaries

Given a set S, an element $s \in S$ and an equivalence relation $R \subseteq S \times S$, we write $[s]_R$ for the *equivalence class* of s under R, i.e., $[s]_R = \{s' \in S \mid (s, s') \in R\}$. We write $S/R = \{[s]_R \mid s \in S\}$ for the set of all equivalence classes in S.

2.1 Probability theory and probabilistic automata

Definition 1 (Probability distributions). *A* probability distribution *over a countable set S is a function $\mu\colon S \to [0,1]$ such that $\sum_{s \in S} \mu(s) = 1$. Given $S' \subseteq S$, we write $\mu(S')$ to denote $\sum_{s' \in S'} \mu(s')$. We use* $\mathrm{Distr}(S)$ *to denote the set of all probability distributions over S, and* $\mathrm{Distr}^*(S)$ *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 probability distribution μ with $\mu(s_1) = p_1$, $\mu(s_2) = p_2, \dots$ ($p_i \neq 0$), we write $\mu = \{s_1 \mapsto p_1, s_2 \mapsto p_2, \dots\}$ and let $\mathrm{spt}(\mu) = \{s_1, s_2, \dots\}$ denote its support. For the deterministic distribution *μ determined by $\mu(t) = 1$ we write $\mathbb{1}_t$.*

Given an equivalence relation R over S and two probability distributions μ, μ' over S, we say that $\mu \equiv_R \mu'$ if and only if $\mu(C) = \mu'(C)$ for all $C \in S/R$.

Probabilistic automata (PAs) are similar to labelled transition systems, except that transitions do not have a fixed successor state anymore. Instead, the state reached after taking a certain transition is determined by a probability distribution [14]. The transitions themselves can be chosen nondeterministically.

Definition 2 (Probabilistic automata). *A* probabilistic automaton (PA) *is a tuple $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$, where S is a countable set of states of which $s^0 \in S$ is initial, L is a countable set of actions, and $\Delta \subseteq S \times L \times \mathrm{Distr}(S)$ is a countable transition relation. We assume that every PA contains an unobservable action $\tau \in L$. If $(s, a, \mu) \in \Delta$, we write $s \xrightarrow{a} \mu$, meaning that state s enables action a, after which the probability to go to $s' \in S$ is $\mu(s')$. If $\mu = \mathbb{1}_t$, we write $s \xrightarrow{a} t$.*

Definition 3 (Paths and traces). *Given a PA $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$, we define a* path *of $\mathcal{A}$ to be either a finite sequence $\pi = s_0 \overset{a_1,\mu_1}{\rightsquigarrow} s_1 \overset{a_2,\mu_2}{\rightsquigarrow} s_2 \overset{a_3,\mu_3}{\rightsquigarrow} \dots \overset{a_n,\mu_n}{\rightsquigarrow} s_n$, or an infinite sequence $\pi' = s_0 \overset{a_1,\mu_1}{\rightsquigarrow} s_1 \overset{a_2,\mu_2}{\rightsquigarrow} s_2 \overset{a_3,\mu_3}{\rightsquigarrow} \dots$.*

For finite paths we require $s_i \in S$ for all $0 \leq i \leq n$, and $s_i \xrightarrow{a_{i+1}} \mu_{i+1}$ as well as $\mu_{i+1}(s_{i+1}) > 0$ for all $0 \leq i < n$. For infinite paths these properties should hold for all $i \geq 0$. A fragment $s \overset{a,\mu}{\rightsquigarrow} s'$ denotes that the transition $s \xrightarrow{a} \mu$ was chosen from state s, after which the successor s' was selected by chance (so $\mu(s') > 0$).

- *If $\pi = s_0 \overset{a,\mathbb{1}_{s_1}}{\rightsquigarrow} s_1 \overset{a,\mathbb{1}_{s_2}}{\rightsquigarrow} \dots \overset{a,\mathbb{1}_{s_n}}{\rightsquigarrow} s_n$ is a path of $\mathcal{A}$ ($n \geq 0$), we write $s_0 \overset{a}{\twoheadrightarrow} s_n$. If each transition is also allowed to be faced backwards, we write $s_0 \overset{a}{\twoheadleftarrow\!\!\twoheadrightarrow} s_n$. If there exists a state t such that $s \overset{a}{\twoheadrightarrow} t$ and $s' \overset{a}{\twoheadrightarrow} t$, we write $s \overset{a}{\twoheadrightarrow}\overset{a}{\twoheadleftarrow} s'$.*
- *We use prefix(π, i) to denote $s_0 \overset{a_1,\mu_1}{\rightsquigarrow} \dots \overset{a_i,\mu_i}{\rightsquigarrow} s_i$, and step$(\pi, i)$ to denote the transition (s_{i-1}, a_i, μ_i). When π is finite we define $|\pi| = n$ and last$(\pi) = s_n$.*
- *We use finpaths$_{\mathcal{A}}$ to denote the set of all finite paths of $\mathcal{A}$, and finpaths$_{\mathcal{A}}(s)$ for all finite paths where $s_0 = s$.*
- *A path's* trace *is the sequence of actions obtained by omitting all its states, distributions and τ-steps; given $\pi = s_0 \overset{a_1,\mu_1}{\rightsquigarrow} s_1 \overset{\tau,\mu_2}{\rightsquigarrow} s_2 \overset{a_3,\mu_3}{\rightsquigarrow} \dots \overset{a_n,\mu_n}{\rightsquigarrow} s_n$, we denote the sequence $a_1 a_3 \dots a_n$ by trace(π).*

2.2 Schedulers

To resolve the nondeterminism in probabilistic automata schedulers are used [16]. Basically, a scheduler is just a function defining for each finite path which transition to take next. The decisions of schedulers are allowed to be *randomised*, i.e., instead of choosing a single transition a scheduler might resolve a nondeterministic choice by a probabilistic choice. Schedulers can be *partial*, i.e., they might assign some probability to the decision of not choosing any next transition.

Definition 4 (Schedulers). *A* scheduler *for a PA $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ is a function*

$$\mathcal{S} \colon \mathit{finpaths}_{\mathcal{A}} \to \mathrm{Distr}(\{\bot\} \cup \Delta),$$

such that for every $\pi \in \mathit{finpaths}_{\mathcal{A}}$ the transitions (s, a, μ) that are scheduled by $\mathcal{S}$ after π (i.e., $\mathcal{S}(\pi)(s, a, \mu) > 0$) are indeed possible after π, i.e., $s = \mathit{last}(\pi)$. The decision of not choosing any transition is represented by $\bot$.

We now define the notions of finite and maximal paths of a PA given a scheduler.

Definition 5 (Paths and maximal paths). *Let $\mathcal{A}$ be a PA and $\mathcal{S}$ a scheduler for $\mathcal{A}$. Then, the set of* finite paths of $\mathcal{A}$ under $\mathcal{S}$ *is given by*

$$\mathit{finpaths}^{\mathcal{S}}_{\mathcal{A}} = \{\pi \in \mathit{finpaths}_{\mathcal{A}} \mid \forall 0 \leq i < |\pi| \, . \, \mathcal{S}(\mathit{prefix}(\pi, i))(\mathit{step}(\pi, i+1)) > 0\}.$$

We define $\mathit{finpaths}^{\mathcal{S}}_{\mathcal{A}}(s) \subseteq \mathit{finpaths}^{\mathcal{S}}_{\mathcal{A}}$ as the set of all such paths starting in s. The set of maximal paths of $\mathcal{A}$ under $\mathcal{S}$ *is given by*

$$\mathit{maxpaths}^{\mathcal{S}}_{\mathcal{A}} = \{\pi \in \mathit{finpaths}^{\mathcal{S}}_{\mathcal{A}} \mid \mathcal{S}(\pi)(\bot) > 0\}.$$

Similarly, $\mathit{maxpaths}^{\mathcal{S}}_{\mathcal{A}}(s)$ is the set of maximal paths of $\mathcal{A}$ under $\mathcal{S}$ starting in s.

We now define the behaviour of a PA $\mathcal{A}$ under a scheduler $\mathcal{S}$. As schedulers resolve all nondeterministic choices, this behaviour is fully probabilistic. We can therefore compute the probability that, starting from a given state s, the path generated by $\mathcal{S}$ has some finite prefix π. This probability is denoted by $P^{\mathcal{S}}_{\mathcal{A},s}(\pi)$.

Definition 6 (Path probabilities). *Let $\mathcal{A}$ be a PA, $\mathcal{S}$ a scheduler for $\mathcal{A}$, and s a state of $\mathcal{A}$. Then, we define the function $P^{\mathcal{S}}_{\mathcal{A},s} \colon \mathit{finpaths}_{\mathcal{A}}(s) \to [0, 1]$ by*

$$P^{\mathcal{S}}_{\mathcal{A},s}(s) = 1; \qquad P^{\mathcal{S}}_{\mathcal{A},s}(\pi \overset{a,\mu}{\rightsquigarrow} t) = P^{\mathcal{S}}_{\mathcal{A},s}(\pi) \cdot \mathcal{S}(\pi)(\mathit{last}(\pi), a, \mu) \cdot \mu(t).$$

Based on these probabilities we can compute the probability distribution $F^{\mathcal{S}}_{\mathcal{A}}(s)$ over the states where a PA $\mathcal{A}$ under a scheduler $\mathcal{S}$ terminates when starting in state s. Note that $F^{\mathcal{S}}_{\mathcal{A}}(s)$ is potentially substochastic (i.e., the probabilities do not add up to 1) when $\mathcal{S}$ allows infinite behaviour.

Definition 7 (Final state probabilities). *Let $\mathcal{A}$ be a PA and $\mathcal{S}$ a scheduler for $\mathcal{A}$. Then, we define the function $F^{\mathcal{S}}_{\mathcal{A}} \colon S \to \mathrm{Distr}^*(S)$ by*

$$F^{\mathcal{S}}_{\mathcal{A}}(s) = \Big\{ s' \mapsto \sum_{\substack{\pi \in \mathit{maxpaths}^{\mathcal{S}}_{\mathcal{A}}(s) \\ \mathit{last}(\pi) = s'}} P^{\mathcal{S}}_{\mathcal{A},s}(\pi) \cdot \mathcal{S}(\pi)(\bot) \mid s' \in S \Big\} \qquad \forall s \in S.$$

3 Branching probabilistic bisimulation

The notion of branching bisimulation for non-probabilistic systems was first introduced in [17]. Basically, it relates states that have an identical branching structure in the presence of τ-actions. Segala defined a generalisation of branching bisimulation for PAs [15], which we present here using the simplified definitions of [16]. First, we intuitively explain weak steps for PAs. Based on these ideas, we then formally introduce branching probabilistic bisimulation.

3.1 Weak steps for probabilistic automata

As τ-steps cannot be observed, we want to abstract from them. Non-probabilistically, this is done via the weak step. A state s can do a weak step to s' under an action a, denoted by $s \xRightarrow{a} s'$, if there exists a path $s \xrightarrow{\tau} s_1 \xrightarrow{\tau} \dots \xrightarrow{\tau} s_n \xrightarrow{a} s'$ with $n \geq 0$ (often, also τ-steps after the a-action are allowed, but this will not concern us). Traditionally, $s \xRightarrow{a} s'$ is thus satisfied by an *appropriate path*.

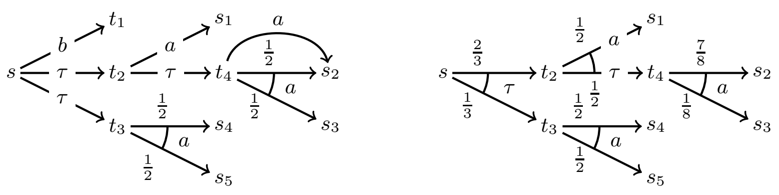
In the probabilistic setting, $s \xRightarrow{a} \mu$ is satisfied by an *appropriate scheduler*. A scheduler $\mathcal{S}$ is appropriate if for every maximal path π that is scheduled from s with non-zero probability, *trace*$(\pi) = a$ and the a-transition is the last transition of the path. Also, the final state distribution $F_{\mathcal{A}}^{\mathcal{S}}(s)$ must be equal to μ.

Example 8. Consider the PA shown in Figure 1(a). We demonstrate that $s \xRightarrow{a} \mu$, with $\mu = \{s_1 \mapsto \frac{8}{24}, s_2 \mapsto \frac{7}{24}, s_3 \mapsto \frac{1}{24}, s_4 \mapsto \frac{4}{24}, s_5 \mapsto \frac{4}{24}\}$. Take the scheduler $\mathcal{S}$:

$$\begin{aligned}
\mathcal{S}(s) &= \{(s, \tau, \mathbb{1}_{t_2}) \mapsto 2/3, (s, \tau, \mathbb{1}_{t_3}) \mapsto 1/3\} \\
\mathcal{S}(t_2) &= \{(t_2, a, \mathbb{1}_{s_1}) \mapsto 1/2, (t_2, \tau, \mathbb{1}_{t_4}) \mapsto 1/2\} \\
\mathcal{S}(t_3) &= \{(t_3, a, \{s_4 \mapsto 1/2, s_5 \mapsto 1/2\}) \mapsto 1\} \\
\mathcal{S}(t_4) &= \{(t_4, a, \mathbb{1}_{s_2}) \mapsto 3/4, (t_4, a, \{s_2 \mapsto 1/2, s_3 \mapsto 1/2\}) \mapsto 1/4\} \\
\mathcal{S}(t_1) &= \mathcal{S}(s_1) = \mathcal{S}(s_2) = \mathcal{S}(s_3) = \mathcal{S}(s_4) = \mathcal{S}(s_5) = \mathbb{1}_{\perp}
\end{aligned}$$

Here we used $\mathcal{S}(s)$ to denote the choice made for every possible path ending in s.

The scheduler is depicted in Figure 1(b). Where it chooses probabilistically between two transitions with the same label, this is represented as a *combined transition*. For instance, from t_4 the transition $(t_4, a, \{s_2 \mapsto 1\})$ is selected with



(a) A PA $\mathcal{A}$. (b) Tree of $s \xRightarrow{a} \mu$.

Fig. 1. Weak steps.

probability 3/4, and $(t_4, a, \{s_2 \mapsto 1/2, s_3 \mapsto 1/2\})$ with probability 1/4. This corresponds to the combined transition $(t_4, a, \{s_2 \mapsto 7/8, s_3 \mapsto 1/8\})$.

Clearly, all maximal paths enabled from s have trace a and end directly after their a-transition. The path probabilities can also be calculated. For instance,

$$P^{\mathcal{S}}_{\mathcal{A},s}(s \overset{\tau,\{t_2 \mapsto 1\}}{\rightsquigarrow} t_2 \overset{\tau,\{t_4 \mapsto 1\}}{\rightsquigarrow} t_4 \overset{a,\{s_2 \mapsto 1\}}{\rightsquigarrow} s_2) = \left(\tfrac{2}{3} \cdot 1\right) \cdot \left(\tfrac{1}{2} \cdot 1\right) \cdot \left(\tfrac{3}{4} \cdot 1\right) = \tfrac{6}{24}$$

$$P^{\mathcal{S}}_{\mathcal{A},s}(s \overset{\tau,\{t_2 \mapsto 1\}}{\rightsquigarrow} t_2 \overset{\tau,\{t_4 \mapsto 1\}}{\rightsquigarrow} t_4 \overset{a,\{s_2 \mapsto 1/2, s_3 \mapsto 1/2\}}{\rightsquigarrow} s_2) = \left(\tfrac{2}{3} \cdot 1\right) \cdot \left(\tfrac{1}{2} \cdot 1\right) \cdot \left(\tfrac{1}{4} \cdot \tfrac{1}{2}\right) = \tfrac{1}{24}$$

As no other maximal paths from s go to s_2, $F^{\mathcal{S}}_{\mathcal{A}}(s)(s_2) = \frac{6}{24} + \frac{1}{24} = \frac{7}{24} = \mu(s_2)$. Similarly, it can be shown that $F^{\mathcal{S}}_{\mathcal{A}}(s)(s_i) = \mu(s_i)$ for $i \in \{1, 3, 4, 5\}$, so indeed it holds that $F^{\mathcal{S}}_{\mathcal{A}}(s) = \mu$. □

3.2 Branching probabilistic bisimulation

Before introducing branching probabilistic bisimulation, we need a restriction on weak steps. Given an equivalence relation R, we let $s \overset{a}{\Longrightarrow}_R \mu$ denote that $(s, t) \in R$ for every state t before the a-step in the tree corresponding to $s \overset{a}{\Longrightarrow} \mu$.

Definition 9 (Branching steps). *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA, $s \in S$, and R an equivalence relation over S. Then, $s \overset{a}{\Longrightarrow}_R \mu$ if either (1) $a = \tau$ and $\mu = \mathbb{1}_s$, or (2) there exists a scheduler $\mathcal{S}$ such that $F^{\mathcal{S}}_{\mathcal{A}}(s) = \mu$ and for every maximal path $s \overset{a_1,\mu_1}{\rightsquigarrow} s_1 \overset{a_2,\mu_2}{\rightsquigarrow} s_2 \overset{a_3,\mu_3}{\rightsquigarrow} \dots \overset{a_n,\mu_n}{\rightsquigarrow} s_n \in maxpaths^{\mathcal{S}}_{\mathcal{A}}(s)$ it holds that $a_n = a$, as well as $a_i = \tau$ and $(s, s_i) \in R$ for all $1 \leq i < n$.*

Definition 10 (Branching probabilistic bisimulation). *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA, then an equivalence relation $R \subseteq S \times S$ is a* branching probabilistic bisimulation *for $\mathcal{A}$ if for all $(s, t) \in R$*

$$s \xrightarrow{a} \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \,.\, t \overset{a}{\Longrightarrow}_R \mu' \wedge \mu \equiv_R \mu'.$$

We say that $p, q \in S$ are branching probabilistically bisimilar*, denoted $p \leftrightarrows_{\mathrm{bp}} q$, if there exists a branching probabilistic bisimulation R for $\mathcal{A}$ such that $(p, q) \in R$.*

Two PAs are branching probabilistically bisimilar if their initial states are (in the disjoint union of the two systems; see Remark 5.3.4 of [16] for the details).

This notion has some appealing properties. First, the definition is robust in the sense that it can be adapted to using $s \overset{a}{\Longrightarrow}_R \mu$ instead of $s \xrightarrow{a} \mu$ in its condition. Although this might seem to strengthen the concept, it does not. Second, the relation $\leftrightarrows_{\mathrm{bp}}$ induced by the definition is an equivalence relation.

Proposition 11. *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA. Then, an equivalence relation $R \subseteq S \times S$ is a branching probabilistic bisimulation for $\mathcal{A}$ iff for all $(s, t) \in R$*

$$s \overset{a}{\Longrightarrow}_R \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \,.\, t \overset{a}{\Longrightarrow}_R \mu' \wedge \mu \equiv_R \mu'.$$

Proposition 12. *The relation $\leftrightarrows_{\mathrm{bp}}$ is an equivalence relation.*

Moreover, Segala showed that branching bisimulation preserves all properties that can be expressed in the probabilistic temporal logic WPCTL (provided that no infinite path of τ-actions can be scheduled with non-zero probability) [15].

4 Confluence for probabilistic automata

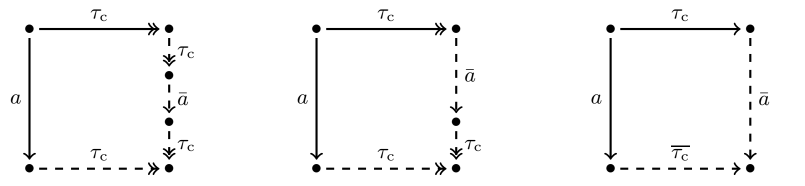
The reductions we introduce are based on sets of *confluent* τ-transitions. Basically, such transitions do not influence a system's behaviour, i.e., a confluent step $s \xrightarrow{\tau} s'$ implies that $s \leftrightarroweq_{\mathrm{bp}} s'$. Confluence therefore paves the way to state space reductions modulo branching probabilistic bisimulation (e.g., by giving confluent τ-transitions priority). Note that not all τ-transitions connect bisimilar states; even though their actions are unobservable, τ-steps might disable behaviour. The aim of our analysis is to underapproximate which τ-transitions are confluent.

For non-probabilistic systems, several notions of confluence already exist [3]. Basically, they all require that if an action a is enabled from a state that also enables a confluent τ-transition, then (1) a will still be enabled after taking that τ-transition (possibly requiring some additional confluent τ-transitions first), and (2) we can always end up in the same state traversing only confluent τ-steps, no matter whether we started by the a- or the τ-transition.

Figure 2 depicts the three notions of confluence we will generalise [3]. They should be interpreted as follows: for any state from which the solid transitions are enabled (universally quantified), there should be a matching for the dashed transitions (existentially quantified). A double-headed arrow denotes a path of zero of more transitions with the corresponding label, and an arrow with label $\overline{a}$ denotes a step that is optional in case $a = \tau$ (i.e., its source and target state may then coincide). The weaker the notion, the more reduction potentially can be achieved (although detection is harder). Note that we first need to find a subset of τ-transitions that we believe are confluence; then, the diagrams are checked.

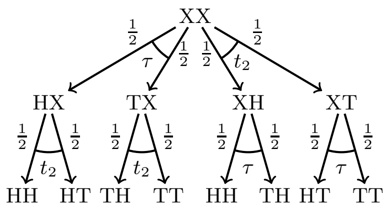
For probabilistic systems, no similar notions of confluence have been defined before. The situation is indeed more difficult, as transitions do not have a single target state anymore. To still enable reductions based on confluence, only τ-transitions with a unique target state might be considered confluent. The next example shows what goes wrong without this precaution. For brevity, from now on we use *bisimilar* as an abbreviation for *branching probabilistically bisimilar*.

Example 13. Consider two people each throwing a die. The PA in Figure 3(a) models this behaviour given that it is unknown who throws first. The first character of each state name indicates whether the first player has not thrown yet (X), or threw heads (H) or tails (T), and the second character indicates the same for the second player. For lay-out purposes, some states were drawn twice.



(a) Weak confluence. (b) Confluence. (c) Strong confluence.

Fig. 2. Three variants of confluence.


(a) The original specification.

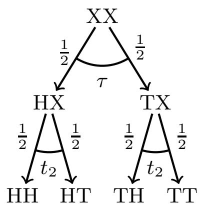
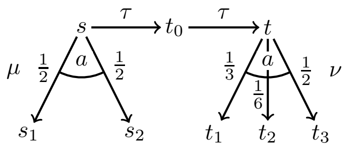

(b) A wrong reduction.

Fig. 3. Two people throwing dice.

We hid the first player's throw action, and kept the other one visible. Now, it might appear that the order in which the a- and the τ-transition occur does not influence the behaviour. However, the τ-step does not connect bisimilar states (assuming HH, HT, TH, and TT to be distinct). After all, from state XX it is possible to reach a state (XH) from where HH is reached with probability 0.5 and TH with probability 0.5. From HX and TX no such state is reachable anymore. Giving the τ-transition priority, as depicted in Figure 3(b), therefore yields a reduced system that is *not* bisimilar to the original system anymore. $\square$



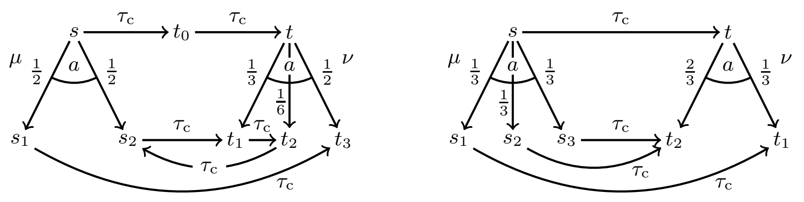
Another difficulty arises when defining probabilistic confluence. Although for LTSs it is clear that a path $a\tau$ should reach the same state as τa, for PAs this is more involved as the a-step leads us to a distribution over states. So, how should the model shown here be completed for the τ-steps to be confluent?

Since we want confluent τ-transitions to connect bisimilar states, we must assure that s, t_0, and t are bisimilar. Therefore, μ and ν must assign equal probabilities to each *class* of bisimilar states. Given the assumption that the other confluent τ-transitions already connect bisimilar states, this is the case if $\mu \equiv_R \nu$ for $R = \{(s, s') \mid s \xrightarrow{\tau}\!\!\twoheadrightarrow \twoheadleftarrow\!\!\xleftarrow{\tau} s'$ using only confluent τ-steps$\}$. The following definition formalises these observations. Here we use the notation $s \xrightarrow{\tau_c} s'$, given a set of τ-transitions c, to denote that $s \xrightarrow{\tau} s'$ and $(s, \tau, s') \in c$.

We define three notions of probabilistic confluence, all requiring the target state of a confluent step to be able to mimic the behaviour of its source state. In the weak version, mimicking may be postponed and is based on joinability (Definition 14a). In the default version, mimicking must happen immediately, but is still based on joinability (Definition 14b). Finally, the strong version requires immediate mimicking by directed steps (Definition 16).

Definition 14 ((Weak) probabilistic confluence). *Let $\mathcal{A} = \langle S, s^0, L, \Delta\rangle$ be a PA and $c \subseteq \{(s, a, \mu) \in \Delta \mid a = \tau, \mu$ is deterministic$\}$ a set of τ-transitions. (a) Then, c is* weakly probabilistically confluent *if $R = \{(s, s') \mid s \xrightarrow{\tau_c}\!\!\twoheadrightarrow \twoheadleftarrow\!\!\xleftarrow{\tau_c} s'\}$ is an equivalence relation, and for every path $s \xrightarrow{\tau_c}\!\!\twoheadrightarrow t$ and all $a \in L, \mu \in \mathrm{Distr}(S)$*

$$s \xrightarrow{a} \mu \implies \exists t' \in S \,.\, t \xrightarrow{\tau_c}\!\!\twoheadrightarrow t' \wedge \\ \Big(\big(\exists \nu \in \mathrm{Distr}(S) \,.\, t' \xrightarrow{a} \nu \wedge \mu \equiv_R \nu\big) \vee \big(a = \tau \wedge \mu \equiv_R \mathbb{1}_{t'}\big)\Big).$$



(a) Weak probabilistic confluence. (b) Strong probabilistic confluence.

Fig. 4. Weak versus strong confluence.

(b) If for every path $s \xrightarrow{\tau_c}\!\!\!\!\!\rightarrow t$ and every transition $s \xrightarrow{a} \mu$ the above implication can be satisfied by taking $t' = t$, then we say that c is probabilistically confluent*.*

For the strongest variant of confluence, moreover, we require the target states of μ to be connected by direct τ_c-transitions to the target states of ν.

Definition 15 (Equivalence up to τ_c-steps). *Let μ, ν be two probability distributions, and let $\nu = \{t_1 \mapsto p_1, t_2 \mapsto p_2, \dots\}$. Then, μ is* equivalent to ν up to τ_c-steps*, denoted by $\mu \overset{\tau_c}{\rightsquigarrow} \nu$, if there exists a partition $\mathrm{spt}(\mu) = \biguplus_{i=1}^{n} S_i$ such that $n = |\mathrm{spt}(\nu)|$ and $\forall 1 \leq i \leq n \colon \mu(S_i) = \nu(t_i) \wedge \forall s \in S_i \colon s \xrightarrow{\tau_c} t_i$.*

Definition 16 (Strong probabilistic confluence). *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA and $c \subseteq \{(s, a, \mu) \in \Delta \mid a = \tau, \mu \text{ is deterministic}\}$ a set of τ-transitions, then c is* strongly probabilistically confluent *if for all $s \xrightarrow{\tau_c} t, a \in L, \mu \in \mathrm{Distr}(S)$*

$$s \xrightarrow{a} \mu \implies \left((\exists \nu \in \mathrm{Distr}(S) \;.\; t \xrightarrow{a} \nu \wedge \mu \overset{\tau_c}{\rightsquigarrow} \nu) \vee (a = \tau \wedge \mu = \mathbb{1}_t) \right).$$

Proposition 17. *Strong probabilistic confluence implies probabilistic confluence, and probabilistic confluence implies weak probabilistic confluence.*

A transition $s \xrightarrow{\tau} t$ is called (weakly, strongly) probabilistically confluent if there exists a (weakly, strongly) probabilistically confluent set c such that $(s, \tau, t) \in c$.

Example 18. Observe the PAs in Figure 4. Assume that all transitions of s, t_0 and t are shown, and that all s_i, t_i, are potentially distinct. We marked all τ-transitions as being confluent, and will verify this for some of them.

In Figure 4(a), both the upper τ_c-steps are weakly probabilistically confluent, most interestingly $s \xrightarrow{\tau_c} t_0$. To verify this, first note that $t_0 \xrightarrow{\tau_c} t$ is (as t_0 has no other outgoing transitions), from where the a-transition of s can be mimicked. To see that indeed $\mu \equiv_R \nu$ (using R from Definition 14), observe that R yields two equivalence classes: $C_1 = \{s_2, t_1, t_2\}$ and $C_2 = \{s_1, t_3\}$. As required, $\mu(C_1) = \frac{1}{2} = \nu(C_1)$ and $\mu(C_2) = \frac{1}{2} = \nu(C_2)$. Clearly $s \xrightarrow{\tau_c} t_0$ is not probabilistically confluent, as t_0 cannot immediately mimic the a-transition of s.

In Figure 4(b) the upper τ_c-transition is strongly probabilistically confluent (and therefore also (weakly) probabilistically confluent). For this, t must be able to directly mimic the a-transition from s. Indeed, it can do so by the transition $t \xrightarrow{a} \nu$. Moreover, $\mu \overset{\tau_c}{\rightsquigarrow} \nu$ also holds, which is easily seen by taking the partition $S_1 = \{s_1\}, S_2 = \{s_2, s_3\}$. □

The following theorem shows that weakly probabilistically confluent τ-transitions indeed connect bisimilar states. With Proposition 17 in mind, this also holds for (strong) probabilistic confluence. Additionally, we show that confluent sets can be joined (so there is a unique maximal confluent set of τ-transitions).

Theorem 19. *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA, $s, s' \in S$ two of its states, and c a weakly probabilistically confluent subset of its τ-transitions. Then,*

$$s \xleftrightarrow{\tau_c} s' \text{ implies } s \leftrightarrow_{\mathrm{bp}} s'.$$

Proposition 20. *Let c, c' be (weakly, strongly) probabilistically confluent sets of τ-transitions. Then, $c \cup c'$ is also (weakly, strongly) probabilistically confluent.*

5 State space reduction using probabilistic confluence

As confluent τ-transitions lead from a state s to a state s' such that s' is equivalent to s (with respect to branching probabilistic bisimulation), all states that can reach each other via such transitions can be merged. That is, we can take the original PA modulo the equivalence relation $\xleftrightarrow{\tau_c}$ and obtain a reduced and bisimilar system. The next definition and theorem formally state this.

Definition 21 **($\mathcal{A}/R$).** *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA and R an equivalence relation over S, then we write $\mathcal{A}/R$ to denote the PA $\mathcal{A}$ modulo R. That is,*

$$\mathcal{A}/R = \langle S/R, [s^0]_R, L, \Delta_R \rangle,$$

with $\Delta_R \subseteq S/R \times L \times \mathrm{Distr}(S/R)$ such that $[s]_R \xrightarrow{a}_R \mu$ if and only there exists a state $s' \in [s]_R$ such that $s' \xrightarrow{a} \mu'$ and $\forall [t]_R \in S/R \,.\, \mu([t]_R) = \sum_{t' \in [t]_R} \mu'(t')$.

Theorem 22. *Let $\mathcal{A}$ be a PA and c a weakly probabilistically confluent subset of its τ-transitions, then $\left(\mathcal{A}/\xleftrightarrow{\tau_c}\right) \leftrightarrow_{\mathrm{bp}} \mathcal{A}$.*

The downside of this method is that, in general, it is hard to compute the equivalence classes according to $\xleftrightarrow{\tau_c}$. Therefore, a slightly adapted reduction technique was proposed in [3], and later used in [4]. There, for each equivalence class a single representative state s was chosen in such a way that all transitions leaving the equivalence class are directly enabled from s. This method relies on (strong) probabilistic confluence, and does not work for the weak variant.

To find a valid representative, we first look at the directed (unlabeled) graph $G = (S, \xrightarrow{\tau_c})$. It contains all states of the original system, and denotes precisely which states can reach each other by taking only τ_c-transitions. Because of the restrictions on τ_c-transitions, the subgraph of G corresponding to each equivalence class $[s]_{\xleftrightarrow{\tau_c}}$ has exactly one terminal strongly connected component (TSCC), from which the representative state for that equivalence class should be chosen. Intuitively, this follows from the fact that τ_c-transitions always lead to a state with at least the same observable transitions as the previous state, and maybe more. (This is not the case for weak probabilistic confluence, therefore the reduction using representatives does not work for that variant of confluence.) The next definition formalises these observations.

Definition 23 (Representation maps). *Let $\mathcal{A}$ be a PA and c a subset of its τ-transitions. Then, a function $\phi_c \colon S \to S$ is a* representation map *for $\mathcal{A}$ under c if*

- $\forall s, s' \in S \,.\, s \xrightarrow{\tau_c} s' \implies \phi_c(s) = \phi_c(s')$;
- $\forall s \in S \,.\, s \overset{\tau_c}{\twoheadrightarrow} \phi_c(s)$.

The first condition ensures that equivalent states are mapped to the same representative, and the second makes sure that every representative is in a TSCC. If c is a probabilistically confluent set of τ-transitions, the second condition and Theorem 19 immediately imply that $s \leftrightarroweq_{\text{bp}} \phi_c(s)$ for every state s.

The next proposition states that for finite-state PAs and probabilistically confluent sets c, there always exists a representation map. As τ_c-transitions are required to always have a deterministic distribution, probabilities are not involved and the proof is identical to the proof for the non-probabilistic case [3].

Proposition 24. *Let $\mathcal{A}$ be a PA and c a probabilistically confluent subset of its τ-transitions. Moreover, let $S_{\mathcal{A}}$ be finite. Then, there exists a function $\phi_c \colon S \to S$ such that ϕ_c is a representation map for $\mathcal{A}$ under c.*

We can now define a PA modulo a representation map ϕ_c. The set of states of such a PA consists of all representatives. When originally $s \xrightarrow{a} \mu$ for some state s, in the reduced system $\phi_c(s) \xrightarrow{a} \mu'$ where μ' assigns a probability to each representative equal to the probability of reaching any state that maps to this representative in the original system. The system will not have any τ_c-transitions.

Definition 25 ($\mathcal{A}/\phi_c$). *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA and c a set of τ-transitions. Moreover, let ϕ_c be a representation map for $\mathcal{A}$ under c. Then, we write $\mathcal{A}/\phi_c$ to denote the PA $\mathcal{A}$ modulo ϕ_c. That is,*

$$\mathcal{A}/\phi_c = \langle \phi_c(S), \phi_c(s^0), L, \Delta_{\phi_c} \rangle,$$

where $\phi_c(S) = \{\phi_c(s) \mid s \in S\}$, and $\Delta_{\phi_c} \subseteq \phi_c(S) \times L \times \text{Distr}(\phi_c(S))$ such that $s \xrightarrow{a}_{\phi_c} \mu$ if and only if $a \neq \tau_c$ and there exists a transition $t \xrightarrow{a} \mu'$ in $\mathcal{A}$ such that $\phi_c(t) = s$ and $\forall s' \in \phi_c(S) \,.\, \mu(s') = \mu'(\{s'' \in S \mid \phi_c(s'') = s'\})$.

From the construction of the representation map it follows that $\mathcal{A}/\phi_c \leftrightarroweq_{\text{bp}} \mathcal{A}$ if c is (strongly) probabilistically confluent.

Theorem 26. *Let $\mathcal{A}$ be a PA and c a probabilistically confluent set of τ-transitions. Also, let ϕ_c be a representation map for $\mathcal{A}$ under c. Then, $(\mathcal{A}/\phi_c) \leftrightarroweq_{\text{bp}} \mathcal{A}$.*

Using this result, state space generation of PAs can be optimised in exactly the same way as has been done for the non-probabilistic setting [4]. Basically, every state visited during the generation is replaced on-the-fly by its representative. In the absence of τ-loops this is easy; just repeatedly follow confluent τ-transitions until none are enabled anymore. When τ-loops are present, a variant of Tarjan's algorithm for finding SCCs can be applied (see [3] for the details).

6 Symbolic detection of probabilistic confluence

Before any reductions can be obtained in practice, probabilistically confluent τ-transitions need to be detected. As our goal is to prevent the generation of large state spaces, this has to be done symbolically.

We propose to do so in the framework of prCRL and LPPEs [9], where systems are modelled by a process algebra and every specification is *linearised* to an intermediate format: the LPPE (linear probabilistic process equation). Basically, an LPPE is a process X with a vector of global variables $\boldsymbol{g}$ of type $\boldsymbol{G}$ and a set of *summands*. A summand is a symbolic transition that is chosen nondeterministically, provided that its guard is enabled (similar to a guarded command). Each summand i is of the form

$$\sum_{\boldsymbol{d_i}:\boldsymbol{D_i}} c_i(\boldsymbol{g},\boldsymbol{d_i}) \Rightarrow a_i(\boldsymbol{g},\boldsymbol{d_i}) \sum_{\boldsymbol{e_i}:\boldsymbol{E_i}}\!\!\bullet\; f_i(\boldsymbol{g},\boldsymbol{d_i},\boldsymbol{e_i}) \colon X(\boldsymbol{n_i}(\boldsymbol{g},\boldsymbol{d_i},\boldsymbol{e_i})).$$

Here, $\boldsymbol{d_i}$ is a (possibly empty) vector of local variables of type $\boldsymbol{D_i}$, which is chosen nondeterministically such that the condition c_i holds. Then, the action $a_i(\boldsymbol{g},\boldsymbol{d_i})$ is taken and a vector $\boldsymbol{e_i}$ of type $\boldsymbol{E_i}$ is chosen probabilistically (each $\boldsymbol{e_i}$ with probability $f_i(\boldsymbol{g},\boldsymbol{d_i},\boldsymbol{e_i})$). Then, the next state is set to $\boldsymbol{n_i}(\boldsymbol{g},\boldsymbol{d_i},\boldsymbol{e_i})$.

The semantics of an LPPE is given as a PA, whose states are precisely all vectors $\boldsymbol{g} \in \boldsymbol{G}$. For all $\boldsymbol{g} \in \boldsymbol{G}$, there is a transition $\boldsymbol{g} \xrightarrow{a} \mu$ if and only if for at least one summand i there is a choice of local variables $\boldsymbol{d_i} \in \boldsymbol{D_i}$ such that

$$c_i(\boldsymbol{g},\boldsymbol{d_i}) \wedge a_i(\boldsymbol{g},\boldsymbol{d_i}) = a \wedge \forall \boldsymbol{e_i} \in \boldsymbol{E_i} \;.\; \mu(\boldsymbol{n_i}(\boldsymbol{g},\boldsymbol{d_i},\boldsymbol{e_i})) = \sum_{\substack{\boldsymbol{e'_i}\in \boldsymbol{E_i} \\ \boldsymbol{n_i}(\boldsymbol{g},\boldsymbol{d_i},\boldsymbol{e_i}) = \boldsymbol{n_i}(\boldsymbol{g},\boldsymbol{d_i},\boldsymbol{e'_i})}} f_i(\boldsymbol{g},\boldsymbol{d_i},\boldsymbol{e'_i}).$$

Example 27. As an example of an LPPE, observe the following specification:

$$\begin{aligned} X(pc : \{1,2\}) = &\sum_{n:\{1,2,3\}} pc = 1 \Rightarrow \text{output}(n) \sum_{i:\{1,2\}}\!\!\bullet\; \tfrac{i}{3} \colon X(i) && (1)\\ + &\qquad\quad pc = 2 \Rightarrow \text{beep} \sum_{j:\{1\}}\!\!\bullet\; 1 \colon X(j) && (2) \end{aligned}$$

The system has one global variable pc (which can be either 1 or 2), and consists of two summands. When $pc = 1$, the first summand is enabled and the system nondeterministically chooses n to be 1, 2 or 3, and outputs the chosen number. Then, the next state is chosen probabilistically; with probability $\frac{1}{3}$ it will be $X(1)$, and with probability $\frac{2}{3}$ it will be $X(2)$. When $pc = 2$, the second summand is enabled, making the system beep and deterministically returning to $X(1)$.

In general, the conditions and actions may depend on both the global variables (in this case pc) and the local variables (in this case n for the first summand), and the probabilities and expressions for determining the next state may additionally depend on the probabilistic variables (in this case i and j). □

Instead of designating *individual* τ-transitions to be probabilistically confluent, we designate *summands* to be so in case we are sure that *all* transitions they might generate are probabilistically confluent. For a summand i to be confluent, clearly $a_i(\boldsymbol{g}, \boldsymbol{d_i}) = \tau$ should hold for all possible values of $\boldsymbol{g}$ and $\boldsymbol{d_i}$. Also, the next state of each of the transitions it generates should be unique: for every possible valuation of $\boldsymbol{g}$ and $\boldsymbol{d_i}$, there should be a single $\boldsymbol{e_i}$ such that $f_i(\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{e_i}) = 1$.

Moreover, a confluence property should hold. For efficiency, we detect a strong variant of strong probabilistic confluence. Basically, a confluent τ-summand i has to commute properly with every summand j (including itself). More precisely, when both are enabled, executing one should not disable the other and the order of their execution should not influence the observable behaviour or the final state. Additionally, i commutes with itself if it generates only one transition. Formally:

$$\begin{aligned}\big(c_i(\boldsymbol{g}, \boldsymbol{d_i}) \wedge c_j(\boldsymbol{g}, \boldsymbol{d_j})\big) \rightarrow \big(i = j \wedge \boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}) = \boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j})\big) \vee \\ \begin{pmatrix} \quad c_j(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}) \wedge c_i(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j}, \boldsymbol{e_j}), \boldsymbol{d_i}) \\ \wedge\, a_j(\boldsymbol{g}, \boldsymbol{d_j}) = a_j(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}) \\ \wedge\, f_j(\boldsymbol{g}, \boldsymbol{d_j}, \boldsymbol{e_j}) = f_j(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}, \boldsymbol{e_j}) \\ \wedge\, \boldsymbol{n_j}(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}, \boldsymbol{e_j}) = \boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j}, \boldsymbol{e_j}), \boldsymbol{d_i}) \end{pmatrix}\end{aligned} \qquad (1)$$

where $\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{d_j}$ and $\boldsymbol{e_j}$ universally quantify over $\boldsymbol{G}$, $\boldsymbol{D_i}$, $\boldsymbol{D_j}$, and $\boldsymbol{E_j}$, respectively. We used $\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i})$ to denote the unique target state of summand i given global state $\boldsymbol{g}$ and local state $\boldsymbol{d_i}$ (so $\boldsymbol{e_i}$ does not need to appear).

As these formulas are quantifier-free and in practice often either trivially false or true, they can easily be solved using an SMT solver for the data types involved. Note that n^2 formulas need to be solved (n being the number of summands); the complexity of this depends on the data types. In our experiments, all formulas could be checked with fairly simple heuristics (such as validating them vacuously by finding contradictory conditions, or by detecting that two summands never use or change the same variable).

Theorem 28. *Let X be an LPPE and $\mathcal{A}$ its PA. Then, if for a summand i we have $\forall \boldsymbol{g} \in \boldsymbol{G}, \boldsymbol{d_i} \in \boldsymbol{D_i} \,.\, a_i(\boldsymbol{g}, \boldsymbol{d_i}) = \tau \wedge \exists \boldsymbol{e_i} \in \boldsymbol{E_i} \,.\, f_i(\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{e_i}) = 1$ and formula (1) holds, the set of transitions generated by i is probabilistically confluent.*

7 Case study

To illustrate the power of probabilistic confluence reduction, we applied it on the leader election protocol introduced in [9]. This protocol, between two nodes, decides on a leader by having both parties throw a die and compare the results. In case of a tie the nodes throw again, otherwise the one that threw highest will be the leader. We hid all actions needed for rolling the dice and communication, keeping only the declarations of leader and follower. The complete model in LPPE format, consisting of twelve summands, can be found in Appendix B.

In [9] we showed the effect of dead-variable reduction on this system. Now, we apply probabilistic confluence reduction both to the LPPE that was already reduced in this way (`leaderReduced`) and to the original one (`leader`). To do

Table 1. Applying confluence reduction to two leader election protocols.

	Original		Reduced		Visited		Running time	
Specification	States	Trans.	States	Trans.	States	Trans.	Before	After
`leader`	3763	6158	1399	1922	1471	4022	0.49 sec	0.35 sec
`leaderReduced`	1693	2438	589	722	661	1382	0.22 sec	0.13 sec
`leader-2-6`	535	710	199	212	271	512	0.15 sec	0.18 sec
`leader-2-36`	18325	23690	6589	6662	9181	17102	13.23 sec	7.38 sec
`leader-3-12`	161803	268515	56839	68919	84059	158403	70.31 sec	39.50 sec
`leader-3-18`	533170	880023	188287	226011	276692	518991	471.42 sec	343.92 sec
`leader-3-19`	out of memory		220996	264996	324544	608433	–	379.19 sec
`leader-4-5`	443840	939264	128553	200312	206569	418632	467.69 sec	93.36 sec

this automatically, we implemented a prototype tool in Haskell for confluence detection and reduction using heuristics[1], relying on Theorem 28.

We used confluence information when generating the state space, applying Theorem 26. As the specification does not contain loops of confluent τ-summands, we could from each state repeatedly execute confluent τ-summands until reaching a state that does not enable any confluent τ-summand anymore, adding only this state to the state space (so no detection of TSCCs was needed).

The results, obtained on a 2.4 GHz, 2 GB Intel Core 2 Duo MacBook, are shown in Table 1; we list the size of the original and reduced state space, as well as the number of states and transitions that were visited during its generation using confluence. Probabilistic confluence reduction clearly has quite an effect on the size of the state space, as well as the number of visited states and therefore the running time. Notice that it nicely works hand-in-hand with dead-variable reduction. Applying both, we reduced by almost an order of magnitude.

We also modeled another leader election protocol that uses asynchronous channels and allows for more parties (Algorithm $\mathcal{B}$ from [6]). We looked at either 2, 3 or 4 parties, who throw either a normal die or one with more or less sides (5, 12, 18, 19, 36). Confluence reduction reduces the state space by about 65%, and the number of visited states (and therefore the running time) by about 50%. With probabilistic POR, comparable results were obtained for similar protocols [8]. As was to be expected, detecting confluence mostly pays off for the larger state spaces. Still, confluence detection only took a fraction of a second for each system; practically all the effort is in the state space generation. From about 180000 states swapping occurs, explaining the excessive growth in running time. Confluence reduction clearly allows us to do more before reaching this limit.

8 Conclusions

This paper introduced three new notions of confluence for probabilistic automata. We first established several facts about these notions, most importantly that they identify branching probabilistically bisimilar states. Then, we showed how probabilistic confluence can be used for state space reduction. As we used

[1] The implementation, case studies and a test script can be downloaded from `http://fmt.cs.utwente.nl/~timmer/papers/tacas2011.html`.

representatives in terminal strongly connected components, these reductions can even be applied to systems containing τ-loops. We discussed how confluence can be detected in the context of a probabilistic process algebra with data by proving formulas in first-order logic. This way, we enabled on-the-fly reductions when generating the state space corresponding to a process-algebraic specification. A case study illustrated the power of our methods.

References

[1] C. Baier, P.R. D'Argenio, and M. Größer. Partial order reduction for probabilistic branching time. In *Proceedings of the 3rd Workshop on Quantitative Aspects of Programming Languages (QAPL)*, volume 153(2) of *ENTCS*, pages 97–116, 2006.

[2] C. Baier, M. Größer, and F. Ciesinski. Partial order reduction for probabilistic systems. In *Proc. of the 1st International Conference on Quantitative Evaluation of Systems (QEST)*, pages 230–239. IEEE Computer Society, 2004.

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

[4] S.C.C. Blom and J.C. van de Pol. State space reduction by proving confluence. In *Proc. of the 14th International Conference on Computer Aided Verification (CAV)*, volume 2404 of *LNCS*, pages 596–609. Springer, 2002.

[5] P.R. D'Argenio and P. Niebert. Partial order reduction on concurrent probabilistic programs. In *Proc. of the 1st International Conference on Quantitative Evaluation of Systems (QEST)*, pages 240–249. IEEE Computer Society, 2004.

[6] W. Fokkink and J. Pang. Simplifying Itai-Rodeh leader election for anonymous rings. In *Proc. of the 4th International Workshop on Automated Verification of Critical Systems (AVoCS)*, volume 128(6) of *ENTCS*, pages 53–68, 2005.

[7] S. Giro, P.R. D'Argenio, and L. María Ferrer Fioriti. Partial order reduction for probabilistic systems: A revision for distributed schedulers. In *Proc. of the 20th International Conference on Concurrency Theory (CONCUR)*, volume 5710 of *LNCS*, pages 338–353. Springer, 2009.

[8] M. Größer. *Reduction Methods for Probabilistic Model Checking.* PhD thesis, Technische Universität Dresden, 2008.

[9] J.-P. Katoen, J.C. van de Pol, M.I.A. Stoelinga, and M. Timmer. A linear process-algebraic format for probabilistic systems with data. In *Proc. of the 10th International Conference on Application of Concurrency to System Design (ACSD)*, pages 213–222. IEEE Computer Society, 2010.

[10] R. Milner. *Communication and Concurrency.* Prentice Hall, 1989.

[11] R. De Nicola and F.W. Vaandrager. Action versus state based logics for transition systems. In *Semantics of Systems of Concurrent Processes*, volume 469 of *LNCS*, pages 407–419. Springer, 1990.

[12] G.J. Pace, F. Lang, and R. Mateescu. Calculating τ-confluence compositionally. In *Proc. of the 15th International Conference on Computer Aided Verification (CAV)*, volume 2725 of *LNCS*, pages 446–459. Springer, 2003.

[13] D. Peled. All from one, one for all: on model checking using representatives. In *Proc. of the 5th International Conference on Computer Aided Verification (CAV)*, volume 697 of *LNCS*, pages 409–423. Springer, 1993.

[14] R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, Massachusetts Institute of Technology, 1995.

[15] R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computation*, 2(2):250–273, 1995.

[16] M.I.A. Stoelinga. *Alea jacta est: verification of probabilistic, real-time and parametric systems*. PhD thesis, University of Nijmegen, 2002.

[17] R.J. van Glabbeek and W.P. Weijland. Branching time and abstraction in bisimulation semantics. *Journal of the ACM*, 43(3):555–600, 1996.

A Proofs

A.1 Proof of Proposition 11

Proposition 11. *Let $\mathcal{A} = \langle S, s^0, L, \Delta\rangle$ be a PA. Then, an equivalence relation $R \subseteq S \times S$ is a branching probabilistic bisimulation for $\mathcal{A}$ iff for all $(s,t) \in R$*

$$s \overset{a}{\Longrightarrow}_R \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \;.\; t \overset{a}{\Longrightarrow}_R \mu' \wedge \mu \equiv_R \mu'.$$

Proof. If every weak step $s \overset{a}{\Longrightarrow}_R \mu$ can be mimicked, then also every step $s \xrightarrow{a} \mu$ can be mimicked. After all, from $s \xrightarrow{a} \mu$ it follows that $s \overset{a}{\Longrightarrow}_R \mu$ for any R (by taking a scheduler that chooses the transition (s, a, μ) with probability 1 from s, and chooses $\perp$ with probability 1 for all other histories). Therefore, the definition given in this proposition is at least as restrictive as the original definition.

Conversely, we show that when every step $s \xrightarrow{a} \mu$ can be mimicked, then also every weak step $s \overset{a}{\Longrightarrow}_R \mu$ can be mimicked. When $a = \tau$ and $\mu = \mathbb{1}_s$ this weak step can be mimicked trivially by $t \overset{\tau}{\Longrightarrow}_R \mathbb{1}_t$. Therefore, from now on we assume that there exists a scheduler $\mathcal{S}$ such that $F_{\mathcal{A}}^{\mathcal{S}}(s) = \mu$, and for every maximal path $s \overset{a_1,\mu_1}{\leadsto} s_1 \overset{a_2,\mu_2}{\leadsto} s_2 \overset{a_3,\mu_3}{\leadsto} \dots \overset{a_n,\mu_n}{\leadsto} s_n \in \mathit{maxpaths}_{\mathcal{A}}(s)$

- $a_i = \tau$ and $(s, s_i) \in R$ for all $1 \leq i < n$;
- $a_n = a$.

As every single transition can be mimicked by t, we can define a scheduler $\mathcal{S}'$ that mimics every choice of $\mathcal{S}$. So, when $\mathcal{S}$ chooses the transition (s, a_1, μ_1) with probability p, we let $\mathcal{S}'$ schedule the transitions necessary for $t \overset{a_1}{\Longrightarrow}_R \mu_1'$ (with $\mu_1 \equiv_R \mu_1'$) with probability p. That is, when for instance $t \xrightarrow{a_1} t_1$ and $t \xrightarrow{a_1} t_2$ should both be assigned probability 0.5 to yield $t \overset{a_1}{\Longrightarrow}_R \mu_1'$, we let $\mathcal{S}'$ choose them with probability $0.5p$. This way, with probability p the tree starting from t reaches a distribution over states that is R-equivalent to μ. As we can then again mimic the transitions of $\mathcal{S}$ from there, and this can continue until the end of each maximal path of $\mathcal{S}$, we obtain a scheduler $\mathcal{S}'$ for which $F_{\mathcal{A}}^{\mathcal{S}'}(t) = \mu'$ with $\mu \equiv_R \mu'$. Moreover, all the states visited before the a-actions in the tree starting from t also remain in the same R equivalence class because of the restrictions of the $\Longrightarrow_R$ relation and the fact that the mimicked steps should yield an R-equivalent distribution. Therefore, indeed $t \overset{a}{\Longrightarrow}_R \mu' \wedge \mu \equiv_R \mu'$. □

A.2 Proof of Proposition 12

Before proving Proposition 12, we first provide a definition and two lemmas.

Definition 29 (Relation composition). *Given two relations R_1 and R_2 over a set S, we use $R_2 \circ R_1$ to denote their* composition*: $R_2 \circ R_1 = \{(x, z) \in S \times S \mid \exists y \in S \,.\, (x, y) \in R_1, (y, z) \in R_2\}$.*

Lemma 30. *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA, $s \in S$, and R an equivalence relation over S. Let $R' \subseteq S \times S$ such that $R' \supseteq R$. Then*

$$s \xRightarrow{a}_R \mu \text{ implies } s \xRightarrow{a}_{R'} \mu$$

Proof. Let $s \xRightarrow{a}_R$. If $a = \tau$ and $\mu = \mathbb{1}_s$ then by definition $s \xRightarrow{a}_{R'} \mu$ for any R', so from now on we assume the other case: there exists a scheduler $\mathcal{S}$ such that $F^{\mathcal{S}}_{\mathcal{A}}(s) = \mu$, and for every path $s \overset{a_1,\mu_1}{\rightsquigarrow} s_1 \overset{a_2,\mu_2}{\rightsquigarrow} s_2 \overset{a_3,\mu_3}{\rightsquigarrow} \dots \overset{a_n,\mu_n}{\rightsquigarrow} s_n \in \mathit{maxpaths}_{\mathcal{A}}(s)$

- $a_i = \tau$ and $(s, s_i) \in R$ for all $1 \leq i < n$;
- $a_n = a$.

Now it is easy to see that the same scheduler proofs the validity of $s \xRightarrow{a}_{R'} \mu$. After all, the only thing that has to be checked when changing R is that $(s, s_i) \in R'$ still holds for all $1 \leq i < n$. However, as $(s, s_i) \in R$ is assumed and $R' \supseteq R'$, this is immediate. □

Lemma 31. *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA. Let $R \subseteq S \times S$ be an equivalence relation such that for all $(s, t) \in R$ it holds that*

$$s \xRightarrow{a}_R \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \,.\, t \xRightarrow{a}_R \mu' \wedge \mu \equiv_R \mu'.$$

Then, for every equivalence relation $R' \subseteq S \times S$ such that $R' \supseteq R$, it holds that

$$s \xRightarrow{a}_{R'} \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \,.\, t \xRightarrow{a}_{R'} \mu' \wedge \mu \equiv_{R'} \mu'.$$

Proof. Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA, and let $R \subseteq S \times S$ be an equivalence relation such that for all $(s, t) \in R$ it holds that

$$s \xRightarrow{a}_R \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \,.\, t \xRightarrow{a}_R \mu' \wedge \mu \equiv_R \mu'$$

By Proposition 11 it follows that for all $(s, t) \in R$ it holds that

$$s \xrightarrow{a} \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \,.\, t \xRightarrow{a}_R \mu' \wedge \mu \equiv_R \mu'$$

Let $R' \subseteq S \times S$ be an equivalence relation such that $R' \supseteq R$. Then, by Lemma 30 it also holds that

$$s \xrightarrow{a} \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \,.\, t \xRightarrow{a}_{R'} \mu' \wedge \mu \equiv_R \mu'$$

Using Proposition 5.2.1.1 and 5.2.1.5 from [16] we obtain that

$$s \xrightarrow{a} \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \,.\, t \xRightarrow{a}_{R'} \mu' \wedge \mu \equiv_{R'} \mu'$$

Now, applying Proposition 11 again, this lemma follows. □

Proposition 12. *The relation $\leftrightarroweq_{\mathrm{bp}}$ is an equivalence relation.*

Proof. Reflexivity of $\leftrightarroweq_{\mathrm{bp}}$ trivially holds; the identity relation $\{(s,s) \mid s \in S\}$ can be used as the branching probabilistic bisimulation.

For symmetry, assume that $p \leftrightarroweq_{\mathrm{bp}} q$. Then, there must exist a branching bisimulation $R \subseteq S \times S$ such that $(p,q) \in S$. As every branching bisimulation is an equivalence relation, also $(q,p) \in S$, so also $q \leftrightarroweq_{\mathrm{bp}} p$.

For transitivity, let $p \leftrightarroweq_{\mathrm{bp}} q$ and $q \leftrightarroweq_{\mathrm{bp}} r$. Then, using Proposition 11, there exists an equivalence relation $R_1 \subseteq S \times S$ such that $(p,q) \in R_1$, and for all $(s,t) \in R_1$ it holds that

$$s \overset{a}{\Longrightarrow}_{R_1} \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \;.\; t \overset{a}{\Longrightarrow}_{R_1} \mu' \wedge \mu \equiv_{R_1} \mu'.$$

Similarly, there exists an equivalence relation $R_2 \subseteq S \times S$ such that $(q,r) \in R_2$, and for all $(s,t) \in R_2$ it holds that

$$s \overset{a}{\Longrightarrow}_{R_2} \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \;.\; t \overset{a}{\Longrightarrow}_{R_2} \mu' \wedge \mu \equiv_{R_2} \mu'.$$

We define $R_3 = (R_2 \circ R_1) \cup (R_1 \circ R_2)$, and let R be the transitive closure of R_3. We first prove that R is an equivalence relation by showing (1) reflexivity, (2) symmetry, and (3) transitivity.

(1) As R_1 are R_2 are equivalence relations, they are reflexive; thus, for every state $s \in S$ it holds that $(s,s) \in R_1$ and $(s,s) \in R_2$. Therefore, $(s,s) \in R_2 \circ R_1$ and thus $(s,s) \in R$.

2) First observe that when $(x,z) \in R_2 \circ R_1$, then there must be a $y \in S$ such that $(x,y) \in R_1$ and $(y,z) \in R_2$, and therefore by symmetry of R_1 and R_2 also $(y,x) \in R_1$ and $(z,y) \in R_2$, and thus $(z,x) \in R_1 \circ R_2$.
Now let $(s,t) \in R$. Then there is an integer $n \geq 2$ such that there exists a sequence of states $s_1, s_2, \ldots, s_n$ such that $s_1 = s$ and $s_n = t$, and for all $1 \leq i < n$ it holds that $(s_i, s_{i+1}) \in (R_2 \circ R_1)$ or $(s_i, s_{i+1}) \in (R_1 \circ R_2)$. By the observation above we can reverse the order of the states, obtaining the sequence $s_n, s_{n-1}, \ldots, s_1$ such that still $s_n = t$ and $s_1 = s$, and for all $1 \leq i < n$ it holds that $(s_i, s_{i+1}) \in (R_2 \circ R_1)$ or $(s_i, s_{i+1}) \in (R_1 \circ R_2)$. To be precise, when $(s_i, s_{i+1}) \in (R_2 \circ R_1)$, then $(s_{i+1}, s_i) \in (R_1 \circ R_2)$, and when $(s_i, s_{i+1}) \in (R_1 \circ R_2)$, then $(s_{i+1}, s_i) \in (R_2 \circ R_1)$. The sequence obtained in this way proves that $(t,s) \in R$.

(3) By definition.

We now prove that $p \leftrightarroweq_{\mathrm{bp}} r$ by showing that $(p,r) \in R$, and that for all $(s,u) \in R$ it holds that

$$s \xrightarrow{a} \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \;.\; u \overset{a}{\Longrightarrow}_{R} \mu' \wedge \mu \equiv_{R} \mu'$$

As $(p,q) \in R_1$ and $(q,r) \in R_2$, it follows immediately that $(p,r) \in R_2 \circ R_1$ and therefore indeed $(p,r) \in R$.

We prove the second part with induction to the number of transitive steps needed to include (s,u) in R.

Base case. Let $(s,u) \in R$ because $(s,u) \in R_2 \circ R_1$ (the case where $(s,u) \in R$ because $(s,u) \in R_1 \circ R_2$ can be proven symmetrically). This implies that there exists a state t such that $(s,t) \in R_1$ and $(t,u) \in R_2$. Let $s \xrightarrow{a} \mu$. Then we know that there exists a $\mu' \in \mathrm{Distr}(S)$ such that $t \overset{a}{\Longrightarrow}_{R_1} \mu'$ and $\mu \equiv_{R_1} \mu'$. By Lemma 30 it follows that $t \overset{a}{\Longrightarrow}_R \mu'$, and using Proposition 5.2.1.1 and 5.2.1.5 from [16] we see that $\mu \equiv_R \mu'$.

As $R \supseteq R_2$, we know by Lemma 31 that for all $(t,u) \in R_2$ it holds that

$$t \overset{a}{\Longrightarrow}_R \mu' \text{ implies } \exists \mu'' \in \mathrm{Distr}(S) \;.\; u \overset{a}{\Longrightarrow}_R \mu'' \wedge \mu' \equiv_R \mu''.$$

We thus showed that $s \xrightarrow{a} \mu$ implies $t \overset{a}{\Longrightarrow}_R \mu'$ (with $\mu \equiv_R \mu'$), and that $t \overset{a}{\Longrightarrow}_R \mu'$ implies $u \overset{a}{\Longrightarrow}_R \mu''$ (with $\mu' \equiv_R \mu''$). Therefore, it follows that if $s \xrightarrow{a} \mu$, indeed there exists a $\mu'' \in \mathrm{Distr}(S)$ such that $u \overset{a}{\Longrightarrow}_R \mu''$. As $\equiv_R$ is an equivalence relation, $\mu \equiv_R \mu''$ follows by transitivity.

Induction hypothesis. Let $(s,t) \in R$ by k transitive steps. Then,

$$s \xrightarrow{a} \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \;.\; t \overset{a}{\Longrightarrow}_R \mu' \wedge \mu \equiv_R \mu'.$$

Inductive step. Let $(s,u) \in R$ by $k+1$ transitive steps. That is, there exists some t such that $(s,t) \in R$ by means of k transitive steps, and either $(t,u) \in R_2 \circ R_1$ or $(t,u) \in R_1 \circ R_2$. We then need to show that

$$s \xrightarrow{a} \mu \text{ implies } \exists \mu'' \in \mathrm{Distr}(S) \;.\; u \overset{a}{\Longrightarrow}_R \mu'' \wedge \mu \equiv_R \mu''.$$

By the induction hypothesis we already know that $s \xrightarrow{a} \mu$ implies $t \overset{a}{\Longrightarrow}_R \mu'$ for some $\mu' \equiv_R \mu$. Moreover, using Proposition 11 and the same reasoning as for the base case, we know that $t \overset{a}{\Longrightarrow}_R \mu'$ implies that $u \overset{a}{\Longrightarrow}_R \mu''$ for some $\mu'' \equiv_R \mu$. Therefore, by transitivity of $\equiv_R$ the statement holds. □

A.3 Proof of Proposition 17

Lemma 32. *Let $\mathcal{A}$ be a PA, $c \subseteq \{(s,a,\mu) \in \Delta \mid a = \tau, \mu \text{ is deterministic}\}$ a set of weakly probabilistically confluent τ-transitions, and $R = \{(s,s') \mid s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} s'\}$. Then, $\mu \overset{\tau_c}{\rightsquigarrow} \nu$ implies $\mu \equiv_R \mu$.*

Proof. Let $\mathcal{A}$ be a PA, $c \subseteq \{(s,a,\mu) \in \Delta \mid a = \tau, \mu \text{ is deterministic}\}$ a set of τ-transitions. Moreover, assume that $\mu \overset{\tau_c}{\rightsquigarrow} \nu$.

Thus, denoting ν by $\nu = \{t_1 \mapsto p_1, t_2 \mapsto p_2, \dots\}$, there exists a partition $\mathrm{spt}(\mu) = \biguplus_{i=1}^{n} S_i$ such that $n = |\mathrm{spt}(\nu)|$ and $\forall 1 \leq i \leq n \colon \mu(S_i) = \nu(t_i) \wedge \forall s \in S_i \colon s \xrightarrow{\tau_c} t_i$.

Now let R' be the smallest equivalence relation that relates the states of every set S_i to each other and to their corresponding t_i. That is, for every S_i and for all $s, s' \in S_i$ it holds that $(s,s') \in R'$ and $(s,t_i) \in R'$.

By definition R' is an equivalence relation, and clearly $\mu \equiv_{R'} \nu$. After all, for every t_i it holds that

$$\nu([t_i]_{R'}) = \nu(\{t_j \mid (t_i, t_j) \in R'\}) = \sum_{\substack{1 \leq j \leq n \\ (t_i, t_j) \in R'}} \nu(t_j) = \sum_{\substack{1 \leq j \leq n \\ (t_i, t_j) \in R'}} \mu(S_j)$$

$$= \sum_{\substack{1 \leq j \leq n \\ \forall s \in S_j \,.\, (t_i, s) \in R'}} \mu(S_j) = \mu([t_i]_{R'})$$

Now let $R = \{(s, s') \mid s \xrightarrow{\tau_c}\!\!\twoheadrightarrow\!\!\twoheadleftarrow\!\!\xleftarrow{\tau_c} s'\}$. A simple tiling argument shows that R is transitive, and reflexivity and symmetry are trivial. Therefore, R is an equivalence relation. Moreover, as $s \xrightarrow{\tau_c} t_i$ implies $s \xrightarrow{\tau_c}\!\!\twoheadrightarrow\!\!\twoheadleftarrow\!\!\xleftarrow{\tau_c} t_i$, and for every $s, s' \in S_i$ we have $s \xrightarrow{\tau_c}\!\!\twoheadrightarrow\!\!\twoheadleftarrow\!\!\xleftarrow{\tau_c} s'$ since they can join at t_i, clearly R also relates the states of every set S_i to each other and to their corresponding t_i. Since R' is the smallest equivalence relation having this property, it follows that $R \supseteq R'$. Because of this, $\mu \equiv_{R'} \nu$ implies $\mu \equiv_R \nu$ (using Proposition 5.2.1.1 and 5.2.1.5 from [16]), which is what we wanted to show. □

Proposition 17. *Strong probabilistic confluence implies probabilistic confluence, and probabilistic confluence implies weak probabilistic confluence.*

Proof. Let $\mathcal{A}$ be a PA and $c \subseteq \{(s, a, \mu) \in \Delta \mid a = \tau, \mu \text{ is deterministic}\}$ a strongly probabilistically confluent set of τ-transitions. Then, for every transition $s \xrightarrow{\tau_c} t$ and all $a \in L, \mu \in \text{Distr}(S)$ it holds that

$$s \xrightarrow{a} \mu \implies \big((\exists \nu \in \text{Distr}(S) \,.\, t \xrightarrow{a} \nu \wedge \mu \overset{\tau_c}{\rightsquigarrow} \nu) \vee (a = \tau \wedge \mu = \mathbb{1}_t) \big)\,.$$

Now let $R = \{(s, s') \mid s \xrightarrow{\tau_c}\!\!\twoheadrightarrow\!\!\twoheadleftarrow\!\!\xleftarrow{\tau_c} s'\}$. As stated in the proof of Lemma 32, R is an equivalence relation.

We need to proof that for every path $s \xrightarrow{\tau_c}\!\!\twoheadrightarrow t$ it holds that

$$s \xrightarrow{a} \mu \implies \big((\exists \nu \in \text{Distr}(S) \,.\, t \xrightarrow{a} \nu \wedge \mu \equiv_R \nu) \vee (a = \tau \wedge \mu \equiv_R \mathbb{1}_t) \big)\,.$$

So, let $s \xrightarrow{\tau_c} t_1 \xrightarrow{\tau_c} \dots \xrightarrow{\tau_c} t_n$ be such a path, and assume strong probabilistic confluence. Let $s \xrightarrow{a} \mu$, and first assume that $a \neq \tau$. Then, by definition of strong probabilistic confluence it must hold that $t_1 \xrightarrow{a} \mu_1$ such that $\mu \overset{\tau_c}{\rightsquigarrow} \mu_1$, and therefore $t_2 \xrightarrow{a} \mu_2$ such that $\mu_1 \overset{\tau_c}{\rightsquigarrow} \mu_2$, and so on, until $t_n \xrightarrow{a} \mu_n$ such that $\mu_{n-1} \overset{\tau_c}{\rightsquigarrow} \mu_n$. By Lemma 32 and transitivity of $\equiv_R$, it follows that $\mu \equiv_R \mu_n$. So, the first disjunct of the formula we needed to prove holds (note that for the empty path s it also holds by reflexivity of $\equiv_R$).

Now assume that $a = \tau$. If $\mu \neq \mathbb{1}_{t_1}$, then the situation is the same as above. If $\mu = \mathbb{1}_{t_1}$, then it follows that $\mu \equiv_R \mathbb{1}_{t_n}$ since $t_1 \xrightarrow{\tau_c}\!\!\twoheadrightarrow t_n$ and thus $(t_1, t_n) \in R$. So, the second disjunct of the formula we needed to prove holds.

So, in both cases c is probabilistically confluent.

The fact the probabilistic confluence implies weak probabilistic confluence is immediate from the definition, as the former is a restriction of the latter. □

A.4 Proof of Theorem 19

Before proving Theorem 19, we first provide an important lemma.

Lemma 33. *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA and $s, t \in S$. Moreover, let c be a weakly probabilistically confluent set of τ-transitions of $\mathcal{A}$. Then,*

$$s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} t \text{ if and only if } s \overset{\tau_c}{\twoheadleftarrow\!\!\twoheadrightarrow} t.$$

Proof. Let $s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} t$. Then, by definition there must exist a state $s' \in S$ such that $s \overset{\tau_c}{\twoheadrightarrow} s'$ and $t \overset{\tau_c}{\twoheadrightarrow} s'$. Therefore, it immediately follows that $s \overset{\tau_c}{\twoheadleftarrow\!\!\twoheadrightarrow} t$.

Now let $s \overset{\tau_c}{\twoheadleftarrow\!\!\twoheadrightarrow} t$. Then, there must be path such as $s \xrightarrow{\tau_c} s_0 \xrightarrow{\tau_c} s_1 \xleftarrow{\tau_c} s_2 \xleftarrow{\tau_c} s_3 \xrightarrow{\tau_c} s_4 \xleftarrow{\tau_c} t$. Potentially (as is the case here) the path contains a fragment of the form $s_i \xleftarrow{\tau_c} s_{i+1} \xrightarrow{\tau_c} s_{i+2}$. Clearly this violates the conditions for the path to show that $s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} t$, as the arrows point in the wrong direction. Also, note that a path without such a fragment does prove that $s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} t$. Therefore, we will show that in any path that can be used to show $s \overset{\tau_c}{\twoheadleftarrow\!\!\twoheadrightarrow} t$ we can eliminate these kind of fragments, obtaining a path that proves $s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} t$. When $s_i \xleftarrow{\tau_c} s_{i+1} \xrightarrow{\tau_c} s_{i+2}$, then by definition of weak probabilistic confluence either (1) $s_i = s_{i+2}$, or (2) there exists a state t such that $s_{i+2} \overset{\tau_c}{\twoheadrightarrow} t$ and $s_i \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} t$.

In case (1), the whole fragment can just be reduced to the state s_i, indeed eliminating the bad fragment. In case (2), assume that $s_i \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} t$ is satisfied by the path $s_i \xrightarrow{\tau_c} t_0 \xrightarrow{\tau_c} \dots \xrightarrow{\tau_c} t_n \xrightarrow{\tau_c} t' \xleftarrow{\tau_c} t'_1 \xleftarrow{\tau_c} \dots \xleftarrow{\tau_c} t'_n \xleftarrow{\tau_c} t$, and that $s_{i+2} \overset{\tau_c}{\twoheadrightarrow} t$ is satisfied by the path $s_{i+2} \xrightarrow{\tau_c} t'_{n+1} \xrightarrow{\tau_c} \dots \xrightarrow{\tau_c} t'_m \xrightarrow{\tau_c} t$. Then, the whole fragment can be reduced to $s_i \xrightarrow{\tau_c} t_0 \xrightarrow{\tau_c} \dots \xrightarrow{\tau_c} t_n \xrightarrow{\tau_c} t' \xleftarrow{\tau_c} t'_1 \xleftarrow{\tau_c} \dots \xleftarrow{\tau_c} t'_n \xleftarrow{\tau_c} t \xleftarrow{\tau_c} t'_m \xleftarrow{\tau_c} \dots \xleftarrow{\tau_c} t'_{n+1} \xleftarrow{\tau_c} s_{i+2}$, which is of the correct form.

Repeating this for all bad fragments, a path proving $s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} t$ appears. □

Theorem 19. *Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA, $s, s' \in S$ two of its states, and c a weakly probabilistically confluent subset of its τ-transitions. Then,*

$$s \overset{\tau_c}{\twoheadleftarrow\!\!\twoheadrightarrow} s' \text{ implies } s \underline{\leftrightarrow}_{\mathrm{bp}} s'.$$

Proof. Let $\mathcal{A} = \langle S, s^0, L, \Delta \rangle$ be a PA and c a weakly probabilistically confluent set of τ-transitions. We prove that $s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} s'$ implies $s \underline{\leftrightarrow}_{\mathrm{bp}} s'$. Clearly, when this hold also $s \xrightarrow{\tau_c} s'$ implies that $s \underline{\leftrightarrow}_{\mathrm{bp}} s'$. Then, as $\underline{\leftrightarrow}_{\mathrm{bp}}$ is an equivalence relation, the theorem follows.

Let $s, s' \in S$ such that $s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} s'$. To prove that indeed $s \underline{\leftrightarrow}_{\mathrm{bp}} s'$, we show that $R = \{(s, t) \mid s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} t\}$ is a branching probabilistic bisimulation. Obviously $(s, s') \in R$. Lemma 33 and the fact that $\overset{\tau_c}{\twoheadleftarrow\!\!\twoheadrightarrow}$ is an equivalence relation imply that R is an equivalence relation.

To show that R is a branching probabilistic bisimulation, let $(s, t) \in R$ be an arbitrary pair of states in R. We prove that

$$s \xrightarrow{a} \mu \text{ implies } \exists \mu' \in \mathrm{Distr}(S) \,.\, t \xRightarrow{a}_R \mu' \wedge \mu \equiv_R \mu'.$$

Let u be the joining state of s and t, i.e., $s \overset{\tau_c}{\twoheadrightarrow} u$ and $t \overset{\tau_c}{\twoheadrightarrow} u$. Let $s \xrightarrow{a} \mu$. We make a case distinction based on whether $a \neq \tau$ or $a = \tau$.

– Assume that $a \neq \tau$. From the definition of weak probabilistic bisimulation it immediately follows that

$$\exists u' \in S \;.\; u \xrightarrow{\tau_c}\!\!\!\!\rightarrow u' \wedge \exists \nu \in \mathrm{Distr}(S) \;.\; u' \xrightarrow{a} \nu \wedge \mu \equiv_R \nu$$

Now let $\mathcal{S}$ be a scheduler choosing with probability 1 the transitions from t corresponding to the path $t \xrightarrow{\tau_c}\!\!\!\!\rightarrow u'$. Then, let $\mathcal{S}$ choose $u' \xrightarrow{a} \nu$ with probability 1, followed by $\perp$ with probability 1. Clearly the final state distribution of $\mathcal{S}$ is ν, and indeed $\mu \equiv_R \nu$ by definition of weak probabilistic confluence. Moreover, as the scheduler only follows τ_c-transitions before it selects $u' \xrightarrow{a} \nu$, the branching condition is satisfied.

– Assume that $a = \tau$. From the definition of weak probabilistic bisimulation it then follows that

$$\exists u' \in S \;.\; u \xrightarrow{\tau_c}\!\!\!\!\rightarrow u' \wedge \big((\exists \nu \in \mathrm{Distr}(S) \;.\; u' \xrightarrow{a} \nu \wedge \mu \equiv_R \nu) \;\vee\; (\mu \equiv_R \mathbb{1}_{u'})\big)$$

When the first disjunct is satisfied the above reasoning applies, so from now on assume that $\exists u' \in S \;.\; u \xrightarrow{\tau_c}\!\!\!\!\rightarrow u' \wedge \mu \equiv_R \mathbb{1}_{u'}$. If $t = u'$, then $t \overset{a}{\Longrightarrow}_R \mu$ is satisfied by the first clause of the definition of branching probabilistic bisimulation as $a = \tau$ and $\mu \equiv_R \mathbb{1}_{u'} = \mathbb{1}_t$. If $t \neq u'$, then the transition can be mimicked by the scheduler choosing with probability 1 the transitions from t corresponding to the path $t \xrightarrow{\tau_c}\!\!\!\!\rightarrow u'$ and then choosing $\perp$ with probability 1. Clearly the final state distribution of $\mathcal{S}$ is $\mathbb{1}_{u'}$, and indeed $\mu \equiv_R \mathbb{1}_{u'}$ by the assumption we made. Moreover, as the scheduler only follows τ_c-transitions, the branching condition is satisfied. $\square$

A.5 Proof of Proposition 20

Proposition 20. *Let c, c' be (weakly, strongly) probabilistically confluent sets of τ-transitions. Then, $c \cup c'$ is also (weakly, strongly) probabilistically confluent.*

Proof. Let $\mathcal{A}$ be a PA and $c \subseteq \{(s, a, \mu) \in \Delta \mid a = \tau, \mu \text{ is deterministic}\}$ a weakly probabilistically confluent set of τ-transitions. Then, for every path $s \xrightarrow{\tau_c}\!\!\!\!\rightarrow t$ and all $a \in L, \mu \in \mathrm{Distr}(S)$ it holds that

$$\begin{aligned} s \xrightarrow{a} \mu \implies & \exists t' \in S \;.\; t \xrightarrow{\tau_c}\!\!\!\!\rightarrow t' \wedge \\ & \big((\exists \nu \in \mathrm{Distr}(S) \;.\; t' \xrightarrow{a} \nu \wedge \mu \equiv_R \nu) \;\vee\; (a = \tau \wedge \mu \equiv_R \mathbb{1}_{t'})\big) \end{aligned}$$

where $R = \{(s, s') \mid s \xrightarrow{\tau_c}\!\!\!\!\rightarrow\!\!\leftarrow\!\!\!\!\xleftarrow{\tau_c} s'\}$ (and R is an equivalence relation).

Let c' be a different weakly probabilistically confluent set of τ-transitions. Then, for every path $s \xrightarrow{\tau_{c'}}\!\!\!\!\rightarrow t$ and all $a \in L, \mu \in \mathrm{Distr}(S)$ it holds that

$$\begin{aligned} s \xrightarrow{a} \mu \implies & \exists t' \in S \;.\; t \xrightarrow{\tau_{c'}}\!\!\!\!\rightarrow t' \wedge \\ & \big((\exists \nu \in \mathrm{Distr}(S) \;.\; t' \xrightarrow{a} \nu \wedge \mu \equiv_{R'} \nu) \;\vee\; (a = \tau \wedge \mu \equiv_{R'} \mathbb{1}_{t'})\big) \end{aligned}$$

where $R' = \{(s, s') \mid s \xrightarrow{\tau_{c'}}\!\!\!\!\rightarrow\!\!\leftarrow\!\!\!\!\xleftarrow{\tau_{c'}} s'\}$ (and R is an equivalence relation).

Now, taking $c'' = c \cup c'$, for every path $s \overset{\tau_{c''}}{\twoheadrightarrow} t$ and all $a \in L, \mu \in \mathrm{Distr}(S)$ it should hold that

$$s \xrightarrow{a} \mu \implies \exists t' \in S \; . \; t \overset{\tau_{c''}}{\twoheadrightarrow} t' \wedge \\ \big((\exists \nu \in \mathrm{Distr}(S) \; . \; t' \xrightarrow{a} \nu \wedge \mu \equiv_{R''} \nu \big) \; \vee \; (a = \tau \wedge \mu \equiv_{R''} \mathbb{1}_{t'}) \big)$$

where $R'' = \{(s, s') \mid s \overset{\tau_{c''}}{\twoheadrightarrow} \overset{\tau_{c''}}{\twoheadleftarrow} s'\}$. The fact that R'' is again an equivalence relation follows easily from the restrictions on the τ_c- and $\tau_{c'}$-steps. Moreover, the required implication follows from a common tiling argument.

A similar argument can be given for probabilistically confluent sets, and for strongly probabilistically confluent sets. $\square$

A.6 Proof of Theorem 22

Theorem 22. *Let $\mathcal{A}$ be a PA and c a weakly probabilistically confluent subset of its τ-transitions, then $\big(\mathcal{A}/\overset{\tau_c}{\twoheadleftarrow\!\twoheadrightarrow}\big) \leftrightarrows_{\mathrm{bp}} \mathcal{A}$.*

Proof. Theorem 19 already showed that all states that can reach each other via τ_c-transitions are branching probabilistically bisimilar. It is well known that such states can therefore be merged, preserving branching probabilistic bisimulation.

The equivalence relation R needed to show this relates the states of the disjoint union of $\mathcal{A}$ and $\big(\mathcal{A}/\overset{\tau_c}{\twoheadleftarrow\!\twoheadrightarrow}\big)$ in such a way that every equivalence class contains precisely one of the states $[s]_{\overset{\tau_c}{\twoheadleftarrow\!\twoheadrightarrow}}$ of $\big(\mathcal{A}/\overset{\tau_c}{\twoheadleftarrow\!\twoheadrightarrow}\big)$ and all the states s' in $\mathcal{A}$ such that $s' \in [s]_{\overset{\tau_c}{\twoheadleftarrow\!\twoheadrightarrow}}$. Clearly, $(s^0, [s^0]_{\overset{\tau_c}{\twoheadleftarrow\!\twoheadrightarrow}}) \in R$. In the remainder of the proof we omit the subscript of equivalence classes $[s]_{\overset{\tau_c}{\twoheadleftarrow\!\twoheadrightarrow}}$, as they are always the same.

Now, let $\{s_1, s_2, \ldots, s_n, [s_1]\}$ be one of the equivalence classes of R. Because of Theorem 19 all the states $s_1, s_2, \ldots, s_n$ are branching probabilistically bisimilar, so we only still need to show that $[s_1]$ can mimic the behaviour of $s_1, s_2, \ldots, s_n$ and vice versa.

So, assume that for instance $s_2 \xrightarrow{a} \mu$. Then, by definition $[s_1] \xrightarrow{a} \nu$ such that $\nu([t]) = \sum_{t' \in [t]} \mu(t')$ for every $[t] \in S/\overset{\tau_c}{\twoheadleftarrow\!\twoheadrightarrow}$. This implies that $\mu \equiv_R \nu$ by the construction of R.

Conversely, let $[s_1] \xrightarrow{a} \mu$. Then, by definition there must exist a state $s_i \in [s]$ such that $s_i \xrightarrow{a} \nu$ and $\forall [t] \in S/\overset{\tau_c}{\twoheadleftarrow\!\twoheadrightarrow} \; . \; \mu([t]) = \sum_{t' \in [t]} \nu(t')$. So, $\mu \equiv_R \nu$. Therefore, s_i can mimic the behaviour of $[s_1]$. As all the states $s_1, s_2, \ldots, s_n$ are branching probabilistically bisimilar, they can all mimic each other, so therefore all states can mimic the behaviour of $[s_1]$. $\square$

A.7 Proof of Theorem 26

Theorem 26. *Let $\mathcal{A}$ be a PA and c a probabilistically confluent set of τ-transitions. Also, let ϕ_c be a representation map for $\mathcal{A}$ under c. Then, $(\mathcal{A}/\phi_c) \leftrightarrows_{\mathrm{bp}} \mathcal{A}$.*

Proof. Theorem 22 already showed that $\big(\mathcal{A}/\overset{\tau_c}{\twoheadleftarrow\!\twoheadrightarrow}\big) \leftrightarrows_{\mathrm{bp}} \mathcal{A}$ if c is weakly probabilistically confluent, and by Proposition 17 this also holds for probabilistically confluent sets c. As $\underline{\leftrightarrow}_{\mathrm{bp}}$ is an equivalence relation by Proposition 12, it therefore suffices to prove that $\left(\mathcal{A}/\overset{\tau_c}{\twoheadleftarrow\twoheadrightarrow}\right) \underline{\leftrightarrow}_{\mathrm{bp}} (\mathcal{A}/\phi_c)$. In this proof we will omit the subscript of equivalence classes $[s]_{\overset{\tau_c}{\twoheadleftarrow\twoheadrightarrow}}$, as they are always the same.

By definition every state of $\mathcal{A}/\overset{\tau_c}{\twoheadleftarrow\twoheadrightarrow}$ is an equivalence class $[s]$, and every state of $\mathcal{A}/\phi_c$ is a representative $\phi_c(s)$. We define the relation R to be the reflexive and symmetric closure of

$$\{([s], \phi_c(s)) \mid s \in S\},$$

and show that it is a branching probabilistic bisimulation. Clearly R is an equivalence relation, and by definition $([s^0], \phi_c(s^0)) \in R$. To show that R is a branching probabilistic bisimulation we prove that $[s] \xrightarrow{a} \mu$ implies that there exists a μ' such that $\phi_c(s) \xRightarrow{a} \mu'$ and $\mu \equiv_R \mu'$, and that $\phi_c(s) \xrightarrow{a} \mu$ implies that there exists a μ' such that $[s] \xRightarrow{a} \mu'$ and $\mu \equiv_R \mu'$.

– Let $[s] \xrightarrow{a} \mu$. We prove that the exists a μ' such that $\phi_c(s) \xRightarrow{a} \mu'$ and $\mu \equiv_R \mu'$ by showing that, assuming $\mu([t]) = p$ for an arbitrary state t, there exists a μ' such that $\phi_c(s) \xRightarrow{a} \mu'$ and $\mu'(\phi_c(t)) = p$.
By Definition 21, there must exist a state $s' \in [s]$ in $\mathcal{A}$ and a μ'' such that $s' \xrightarrow{a} \mu''$ and $\forall [t] \in S/\overset{\tau_c}{\twoheadleftarrow\twoheadrightarrow} \,.\, \mu([t]) = \sum_{t' \in [t]} \mu''(t')$. That is, μ'' also assigns probability p to the event of going to a state in the equivalence class $[t]$.
Now, by definition of representatives $s' \overset{\tau_c}{\twoheadrightarrow} \phi_c(s)$, and therefore, by definition of probabilistic confluence it must be the case that

$$\left(\exists \nu \in \mathrm{Distr}(S) \,.\, \phi_c(s) \xrightarrow{a} \nu \wedge \mu'' \equiv_{R'} \nu\right) \vee \left(a = \tau \wedge \mu'' \equiv_{R'} \mathbb{1}_{\phi_c(s)}\right),$$

where $R' = \{(s, s') \mid s \overset{\tau_c}{\twoheadrightarrow}\overset{\tau_c}{\twoheadleftarrow} s'\}$. First assume that $a \neq \tau$. Then, in $\mathcal{A}$ we have $\phi_c(s) \xrightarrow{a} \nu$ with $\nu \equiv_{R'} \mu''$. Given the definition of R' and Lemma 33, this implies that $\nu([t]) = \mu''([t]) = p$. As $[t]$ is exactly the set of all states that have $\phi_c(t)$ as their representative, by Definition 25 we also have $\phi_c(s) \xrightarrow{a} \nu'$ with $\nu'(\phi_c(t)) = p$ in $\mathcal{A}/\phi_c$. As the existence of a transition implies the existence of a weak step, this finishes this part of the proof.
When $a = \tau$, either the above holds, or $\mu'' \equiv_{R'} \mathbb{1}_{\phi_c(s)}$. In the latter case, as we also already knew that μ'' assigns probability p to the event of going to a state in the equivalence class $[t]$, apparently $\phi_c(s) \in [t]$ and $p = 1$. From $\phi_c(s) \in [t]$ it follows by definition that $\phi_c(s) = \phi_c(t)$. By definition of branching steps $\phi_c(s) \xRightarrow{\tau} \mathbb{1}_{\phi_c(s)}$, and given the above indeed $\mathbb{1}_{\phi_c(s)}(\phi_c(t)) = p$.

– Let $\phi_c(s) \xrightarrow{a} \mu$, and let $\mu(\phi_c(t)) = p$ for some state t. We prove that there exists a μ' such that $[s] \xRightarrow{a} \mu'$ and $\mu'([t]) = p$. As $\phi_c(s) \xrightarrow{a} \mu$, there must exist a transition $t' \xrightarrow{a} \mu'$ in the original PA such that $\phi_c(t') = \phi_c(s)$ and

$$\forall s' \in \phi_c(S) \,.\, \mu(s') = \mu'(\{s'' \in S \mid \phi_c(s'') = s'\}).$$

So, because we assumed $\mu(\phi_c(t)) = p$, it should hold that $\mu'(\{s'' \in S \mid \phi_c(s'') = \phi_c(t)\} = p$. Stated otherwise, and recognising that the set of states with the same representative as t is precisely the set $[t]$, we get $\mu'([t]) = p$. As $t' \xrightarrow{a} \mu'$ such that $\mu'([t]) = p$, and because s and t' have the same representative, also $[s] \xrightarrow{a} \mu''$ such that $\mu''([t]) = p$. Observing that a normal step implies a weak step, we're done. $\square$

A.8 Proof of Theorem 28

Theorem 28. *Let X be an LPPE and $\mathcal{A}$ its PA. Then, if for a summand i we have $\forall \boldsymbol{g} \in \boldsymbol{G}, \boldsymbol{d_i} \in \boldsymbol{D_i} \,.\, a_i(\boldsymbol{g}, \boldsymbol{d_i}) = \tau \wedge \exists \boldsymbol{e_i} \in \boldsymbol{E_i} \,.\, f_i(\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{e_i}) = 1$ and formula (1) holds, the set of transitions generated by i is probabilistically confluent.*

Proof. Let X be an LPPE and $\mathcal{A}$ its underlying PA. So, by the operational semantics the state set S of this PA contains precisely all vectors $\boldsymbol{g} \in \boldsymbol{G}$.

Let i be a summand such that $\forall \boldsymbol{g} \in \boldsymbol{G}, \boldsymbol{d_i} \in \boldsymbol{D_i} \,.\, a_i(\boldsymbol{g}, \boldsymbol{d_i}) = \tau \wedge \exists \boldsymbol{e_i} \in \boldsymbol{E_i} \,.\, f_i(\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{e_i}) = 1$ and for every summand j it holds that

$$\big(c_i(\boldsymbol{g}, \boldsymbol{d_i}) \wedge c_j(\boldsymbol{g}, \boldsymbol{d_j})\big) \rightarrow \big(i = j \wedge \boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}) = \boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j})\big) \vee \begin{pmatrix} \quad c_j(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}) \wedge c_i(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j}, \boldsymbol{e_j}), \boldsymbol{d_i}) \\ \wedge\, a_j(\boldsymbol{g}, \boldsymbol{d_j}) = a_j(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}) \\ \wedge\, f_j(\boldsymbol{g}, \boldsymbol{d_j}, \boldsymbol{e_j}) = f_j(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}, \boldsymbol{e_j}) \\ \wedge\, \boldsymbol{n_j}(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}, \boldsymbol{e_j}) = \boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j}, \boldsymbol{e_j}), \boldsymbol{d_i}) \end{pmatrix} \quad (2)$$

By the operational semantics, the transitions generated by i are those transitions $\boldsymbol{g} \xrightarrow{a} \mu$ such that there is a choice of local variables $\boldsymbol{d_i} \in \boldsymbol{D_i}$ such that

$$c_i(\boldsymbol{g}, \boldsymbol{d_i}) \wedge a_i(\boldsymbol{g}, \boldsymbol{d_i}) = a \wedge \forall \boldsymbol{e_i} \in \boldsymbol{E_i} \,.\, \mu(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{e_i})) = \sum_{\substack{\boldsymbol{e'_i} \in \boldsymbol{E_i} \\ \boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{e_i}) = \boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{e'_i})}} f_i(\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{e'_i}).$$

Let c be the set containing these transitions. We prove that c is probabilistically confluent by showing that it is strongly probabilistically confluent, relying on Proposition 17. Note that the sets c_i from all confluent summands i can be combined into a single confluent set by Proposition 20.

Let $\boldsymbol{g} \xrightarrow{a} \mu$ be an arbitrary transition in c, and let $\boldsymbol{d'_i} \in \boldsymbol{D_i}$ be the local variables that had to be chosen for i to generate it. So,

$$c_i(\boldsymbol{g}, \boldsymbol{d'_i}) \wedge a_i(\boldsymbol{g}, \boldsymbol{d'_i}) = a \wedge \forall \boldsymbol{e_i} \in \boldsymbol{E_i} \,.\, \mu(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d'_i}, \boldsymbol{e_i})) = \sum_{\substack{\boldsymbol{e'_i} \in \boldsymbol{E_i} \\ \boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d'_i}, \boldsymbol{e_i}) = \boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d'_i}, \boldsymbol{e'_i})}} f_i(\boldsymbol{g}, \boldsymbol{d'_i}, \boldsymbol{e'_i}).$$

Because $\forall \boldsymbol{g} \in \boldsymbol{G}, \boldsymbol{d_i} \in \boldsymbol{D_i} \,.\, a_i(\boldsymbol{g}, \boldsymbol{d_i}) = \tau \wedge \exists \boldsymbol{e_i} \in \boldsymbol{E_i} \,.\, f_i(\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{e_i}) = 1$, it follows that $a = \tau$. Moreover, μ is deterministic (as it assigns probability 1 to the next state determined by $\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d'_i}, \boldsymbol{e_i})$, where $\boldsymbol{e_i}$ is the unique element of $\boldsymbol{E_i}$ such that $f_i(\boldsymbol{g}, \boldsymbol{d'_i}, \boldsymbol{e_i}) = 1$). We use $\boldsymbol{g'}$ to denote this unique target state. Thus, using the notation $\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d'_i})$ for the unique target state given the global state $\boldsymbol{g}$ and local variables $\boldsymbol{d'_i}$, we have $\boldsymbol{g'} = \boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d'_i})$.

So, we indeed can write $\boldsymbol{g} \xrightarrow{a} \mu$ as $\boldsymbol{g} \xrightarrow{\tau_c} \boldsymbol{g'}$. To show that c is strongly probabilistically confluent it remains to show that for every transition $\boldsymbol{g} \xrightarrow{a} \mu$

$$\big(\exists \nu \in \mathrm{Distr}(S) \,.\, \boldsymbol{g'} \xrightarrow{a} \nu \wedge \mu \overset{\tau_c}{\rightsquigarrow} \nu\big) \vee (a = \tau \wedge \mu = \mathbb{1}_{\boldsymbol{g'}})\,. \quad (3)$$

Let $\boldsymbol{g} \xrightarrow{a} \mu$ be such a transition for which this needs to be shown. Let j be the summand from which it originates, and let $\boldsymbol{d'_j}$ be the local variables that had to be chosen for j to generate it. So,

$$c_j(\boldsymbol{g}, \boldsymbol{d_j'}) \wedge a_j(\boldsymbol{g}, \boldsymbol{d_j'}) = a \wedge \forall \boldsymbol{e_j} \in \boldsymbol{E_j} \ . \ \mu(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j})) = \sum_{\substack{\boldsymbol{e_j'} \in \boldsymbol{E_j} \\ \boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j}) = \boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'})}} f_j(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'})$$

Now, as we showed above that $c_i(\boldsymbol{g}, \boldsymbol{d_i'})$ and $c_j(\boldsymbol{g}, \boldsymbol{d_j'})$ hold, we can apply Equation (2). Therefore, we know that either $\big(i = j \wedge \boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i'}) = \boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'})\big)$, or for every $\boldsymbol{e_j} \in \boldsymbol{E_j}$ it holds that

$$\begin{pmatrix} \quad c_j(\boldsymbol{g'}, \boldsymbol{d_j'}) \wedge c_i(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j}), \boldsymbol{d_i'}) \\ \wedge\, a_j(\boldsymbol{g}, \boldsymbol{d_j'}) = a_j(\boldsymbol{g'}, \boldsymbol{d_j'}) \\ \wedge\, f_j(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j}) = f_j(\boldsymbol{g'}, \boldsymbol{d_j'}, \boldsymbol{e_j}) \\ \wedge\, \boldsymbol{n_j}(\boldsymbol{g'}, \boldsymbol{d_j'}, \boldsymbol{e_j}) = \boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j}), \boldsymbol{d_i'}) \end{pmatrix}$$

(where we already substituted $\boldsymbol{g'}$ for $\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i'})$.

In the first case, both $\boldsymbol{g} \xrightarrow{\tau_c} \boldsymbol{g'}$ and $\boldsymbol{g} \xrightarrow{a} \mu$ result from the same summand. Therefore, $a = \tau$ is immediate, and indeed $\mu = \mathbb{1}_{\boldsymbol{g'}}$ because $\boldsymbol{g'} = \boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i'})$, $\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i'}) = \boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'})$, and $\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'})$ denotes the unique target state of j given the local variables $\boldsymbol{d_j'}$. Therefore, the second disjunct of Equation 3 holds.

In the second case, we know several things. First of all, $c_j(\boldsymbol{g'}, \boldsymbol{d_j'})$, so j is still enabled using the local variables $\boldsymbol{d_j'}$ in state $\boldsymbol{g'}$. Moreover, $a_j(\boldsymbol{g}, \boldsymbol{d_j'}) = a_j(\boldsymbol{g'}, \boldsymbol{d_j'})$, so since above we already showed that $a_j(\boldsymbol{g}, \boldsymbol{d_j'}) = a$, it follows that taking j from $\boldsymbol{g'}$ using the local variables $\boldsymbol{d_j'}$ also results in the action a. So, this shows that there indeed exists a distribution $\nu \in \mathrm{Distr}(S)$ such that $\boldsymbol{g'} \xrightarrow{a} \nu$.

The final part of the proofs explains that $\mu \overset{\tau_c}{\rightsquigarrow} \nu$. For this, we need to show that there exists a partition $\mathrm{spt}(\mu) = \biguplus_{k=1}^{n} S_k$ such that $n = |\mathrm{spt}(\nu)|$ and $\forall 1 \leq k \leq n \colon \mu(S_k) = \nu(t_k) \wedge \forall s \in S_k \colon s \xrightarrow{\tau_c} t_k$.

To see this, assume that $\nu = \{\boldsymbol{g_1'} \mapsto p_1, \boldsymbol{g_2'} \mapsto p_2, \dots\}$. Now, let the partition be given by $S_k = \{\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'}) \mid \boldsymbol{e_j'} \in \boldsymbol{E_j}, \boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'}), \boldsymbol{d_i'}) = \boldsymbol{g_k'}\}$. Clearly, by construction the partition contains precisely as many elements as $\mathrm{spt}(\nu)$. It is also indeed a valid partition: (1) there is no state in $\mathrm{spt}(\mu)$ that is not present in at least one of the S_k's, since every such state can be written as $\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'})$ and because of the requirement that $c_i(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j}), \boldsymbol{d_i'})$ each of them goes to at least one of the $\boldsymbol{g_k'}$'s (which is also guaranteed to be in the support of ν if it is in the support of μ, as follows from the computation below); (2) there is no state in $\mathrm{spt}(\mu)$ that occurs in more than one S_k, since any such state can be written as $\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'})$ and therefore precisely occurs only in the S_k such that $\boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'}), \boldsymbol{d_i'}) = \boldsymbol{g_k'}$.

Furthermore, for any S_k:

$$\begin{aligned} \mu(S_k) &= \mu(\{\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'}) \mid \boldsymbol{e_j'} \in \boldsymbol{E_j}, \boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'}), \boldsymbol{d_i'}) = \boldsymbol{g_k'}\}) \\ &= \sum_{\substack{\boldsymbol{e_j'} \in \boldsymbol{E_j} \\ \boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'}), \boldsymbol{d_i'}) = \boldsymbol{g_k'}}} f_j(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'}) \\ &= \sum_{\substack{\boldsymbol{e_j'} \in \boldsymbol{E_j} \\ \boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j'}, \boldsymbol{e_j'}), \boldsymbol{d_i'}) = \boldsymbol{g_k'}}} f_j(\boldsymbol{g'}, \boldsymbol{d_j'}, \boldsymbol{e_j'}) \end{aligned}$$

$$= \sum_{\substack{\boldsymbol{e'_j} \in \boldsymbol{E_j} \\ \boldsymbol{n_j}(\boldsymbol{g'}, \boldsymbol{d'_j}, \boldsymbol{e_j}) = \boldsymbol{g'_k}}} f_j(\boldsymbol{g'}, \boldsymbol{d'_j}, \boldsymbol{e'_j})$$

$$= \nu(\boldsymbol{g'_k})$$

The first equality just unfolds the construction of S_k, the second applies the operational semantics of summands, the third applied the requirement that $f_j(\boldsymbol{g}, \boldsymbol{d'_j}, \boldsymbol{e_j}) = f_j(\boldsymbol{g'}, \boldsymbol{d'_j}, \boldsymbol{e_j})$ for every $\boldsymbol{e_j} \in \boldsymbol{E_j}$, the fourth applies the requirement that $\boldsymbol{n_j}(\boldsymbol{g'}, \boldsymbol{d'_j}, \boldsymbol{e_j}) = \boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d'_j}, \boldsymbol{e_j}), \boldsymbol{d'_i})$ for every $\boldsymbol{e_j} \in \boldsymbol{E_j}$, and the last equality again uses the operational semantics.

The fact that every state $s \in S_k$ has a τ_c-transition to $\boldsymbol{g'_k}$ follows directly from the requirement that $c_i(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d'_j}, \boldsymbol{e_j}), \boldsymbol{d'_i})$ for every $\boldsymbol{e_j} \in \boldsymbol{E_j}$ and the construction that $\boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d'_j}, \boldsymbol{e'_j}), \boldsymbol{d'_i}) = \boldsymbol{g'_k}$. □

B Case study: a leader election protocol

The LPPE for the leader election protocol discussed in Section 7 is shown in Figure 5. For readability, in every summand we only show the parameters that are used or updated. As summations can use an existing parameter name for a local variable, statements such as *d_2* := *d_2* occur. This means that, in the next state, the global variable *d_2* will have the value of the local variable *d_2*. We use the notation reset(x) to denote that a variable x is reset to its initial value.

We will now give an idea of how confluence detection works for this LPPE. In fact, for the case studies we implemented a tool that does this kind of reasoning automatically.

As the summands 9, 10, 11, and 12 do not have a τ-action, and the target states of the summands 1, 2, 7, and 8 are not determined by a deterministic distribution, the only candidates for confluent summands are 3, 4, 5, and 6. It turns out that all of these are indeed confluent. To show for instance the confluence of summand 3, we need to prove that, for every summand j, and all $\boldsymbol{g}, \boldsymbol{d_i}, \boldsymbol{d_j}$, and $\boldsymbol{e_j}$:

$$\big(c_i(\boldsymbol{g}, \boldsymbol{d_i}) \wedge c_j(\boldsymbol{g}, \boldsymbol{d_j})\big) \rightarrow \big(i = j \wedge \boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}) = \boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j})\big) \vee \begin{pmatrix} \quad c_j(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}) \wedge c_i(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j}, \boldsymbol{e_j}), \boldsymbol{d_i}) \\ \wedge\, a_j(\boldsymbol{g}, \boldsymbol{d_j}) = a_j(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}) \\ \wedge\, f_j(\boldsymbol{g}, \boldsymbol{d_j}, \boldsymbol{e_j}) = f_j(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}, \boldsymbol{e_j}) \\ \wedge\, \boldsymbol{n_j}(\boldsymbol{n_i}(\boldsymbol{g}, \boldsymbol{d_i}), \boldsymbol{d_j}, \boldsymbol{e_j}) = \boldsymbol{n_i}(\boldsymbol{n_j}(\boldsymbol{g}, \boldsymbol{d_j}, \boldsymbol{e_j}), \boldsymbol{d_i}) \end{pmatrix}$$

Note that summands 1, 5, 7, 9, and 11 can never be enabled at the same time as summand 3, as summand 3 requires *pc_2* = 2 and these summands all require *pc_2* to have another value. Also, 6 can never be enabled at the same time as 3 because of their contradictory requirements on *set_3*. Therefore, the formula we need to prove holds vacuously for all these summands (as the left-hand side of the implication can never be true).

So, we only still need to prove the formula for the summands 2, 4, 8, 10, and 12, and for 3 itself. Simple observation shows that summand 3 only deals with the variables *pc_2*, *set_3*, *e_2*, *val_3*, and *d_2*, and that none of the summands 2, 4, 8, 10, and 12 either uses or updates any of these variables. This immediately implies that summand 3 cannot disable any of these summands or the other way around, that summand 3 cannot influence the actions or probabilities of these summands, and that the state that is reached after first executing summand 3 and then one of these summands must be identical to the state that is reached when doing this the other way around (given the same probabilistic choice). Therefore, the formula holds for all possible valuations of its parameters (by means of the second disjunct).

To see that summand 3 is confluent with itself, notice that it does not have any local variables to choose from. Therefore, the first disjunct of the formula holds trivially.

The same kind of reasoning can be applied to show that summands 4, 5, and 6 are confluent.

$$\begin{aligned}
&Z(\mathit{val\_1} : \{\mathit{1..6}\}, \mathit{set\_1} : \mathit{Bool}, \mathit{pc\_2} : \{\mathit{1..4}\}, \mathit{d\_2} : \{\mathit{1..6}\}, \mathit{e\_2} : \{\mathit{1..6}\},\\
&\quad \mathit{val\_3} : \{\mathit{1..6}\}, \mathit{set\_3} : \mathit{Bool}, \mathit{pc\_4} : \{\mathit{1..4}\}, \mathit{d\_4} : \{\mathit{1..6}\}, \mathit{e\_4} : \{\mathit{1..6}\}) =
\end{aligned}$$

$$\mathit{pc\_2} = 1 \Rightarrow \tau \sum_{\mathit{d\_2}:\{\mathit{1..6}\}} \tfrac{1}{6} : Z(\mathit{pc\_2} := 2, \mathit{d\_2} := \mathit{d\_2}, \mathrm{reset}(\mathit{e\_2})) \tag{1}$$

$$+\ \mathit{pc\_4} = 1 \Rightarrow \tau \sum_{\mathit{d\_4}:\{\mathit{1..6}\}} \tfrac{1}{6} : Z(\mathit{pc\_4} := 2, \mathit{d\_4} := \mathit{d\_4}, \mathrm{reset}(\mathit{e\_4})) \tag{2}$$

$$+\ \mathit{pc\_2} = 2 \wedge \neg \mathit{set\_3} \Rightarrow \tau \sum_{k:\{1\}} 1.0 : Z(\mathit{pc\_2} := 3, \mathrm{reset}(\mathit{e\_2}), \mathit{val\_3} := \mathit{d\_2}, \mathit{set\_3} := \mathit{true}) \tag{3}$$

$$+\ \mathit{pc\_4} = 2 \wedge \neg \mathit{set\_1} \Rightarrow \tau \sum_{k:\{1\}} 1.0 : Z(\mathit{val\_1} := \mathit{d\_4}, \mathit{set\_1} := \mathit{true}, \mathit{pc\_4} := 3, \mathrm{reset}(\mathit{e\_4})) \tag{4}$$

$$+\ \mathit{pc\_2} = 3 \wedge \mathit{set\_1} \Rightarrow \tau \sum_{k:\{1\}} 1.0 : Z(\mathit{set\_1} := \mathit{false}, \mathit{pc\_2} := 4, \mathit{e\_2} := \mathit{val\_1}) \tag{5}$$

$$+\ \mathit{pc\_4} = 3 \wedge \mathit{set\_3} \Rightarrow \tau \sum_{k:\{1\}} 1.0 : Z(\mathit{set\_3} := \mathit{false}, \mathit{pc\_4} := 4, \mathit{e\_4} := \mathit{val\_3}) \tag{6}$$

$$+\ \mathit{pc\_2} = 4 \wedge \mathit{d\_2} = \mathit{e\_2} \Rightarrow \tau \sum_{\mathit{d\_2}:\{\mathit{1..6}\}} \tfrac{1}{6} : Z(\mathit{pc\_2} := 2, \mathit{d\_2} := \mathit{d\_2}, \mathrm{reset}(\mathit{e\_2})) \tag{7}$$

$$+\ \mathit{pc\_4} = 4 \wedge \mathit{d\_4} = \mathit{e\_4} \Rightarrow \tau \sum_{\mathit{d\_4}:\{\mathit{1..6}\}} \tfrac{1}{6} : Z(\mathit{pc\_4} := 2, \mathit{d\_4} := \mathit{d\_4}, \mathrm{reset}(\mathit{e\_4})) \tag{8}$$

$$+\ \mathit{pc\_2} = 4 \wedge \mathit{d\_2} > \mathit{e\_2} \Rightarrow \mathit{leader}(\mathit{one}) \sum_{k:\{1\}} 1.0 : Z(\mathit{pc\_2} := 1, \mathrm{reset}(\mathit{d\_2}), \mathrm{reset}(\mathit{e\_2})) \tag{9}$$

$$+\ \mathit{pc\_4} = 4 \wedge \mathit{d\_4} > \mathit{e\_4} \Rightarrow \mathit{leader}(\mathit{two}) \sum_{k:\{1\}} 1.0 : Z(\mathit{pc\_4} := 1, \mathrm{reset}(\mathit{d\_4}), \mathrm{reset}(\mathit{e\_4})) \tag{10}$$

$$+\ \mathit{pc\_2} = 4 \wedge \mathit{d\_2} < \mathit{e\_2} \Rightarrow \mathit{follower}(\mathit{one}) \sum_{k:\{1\}} 1.0 : Z(\mathit{pc\_2} := 1, \mathrm{reset}(\mathit{d\_2}), \mathrm{reset}(\mathit{e\_2})) \tag{11}$$

$$+\ \mathit{pc\_4} = 4 \wedge \mathit{d\_4} < \mathit{e\_4} \Rightarrow \mathit{follower}(\mathit{two}) \sum_{k:\{1\}} 1.0 : Z(\mathit{pc\_4} := 1, \mathrm{reset}(\mathit{d\_4}), \mathrm{reset}(\mathit{e\_4})) \tag{12}$$

Fig. 5. The LPPE of a leader election protocol.