*Math. Struct. in Comp. Science* (2014), vol. 24, iss. 1, e240103, 77 pages. © Cambridge University Press 2013
doi:10.1017/S0960129513000029

# Equivalence relations for modular performance evaluation in dtsPBC

IGOR V. TARASYUK[†]

*A.P. Ershov Institute of Informatics Systems, Siberian Branch of the Russian Academy of Sciences, 6, Acad. Lavrentiev ave., 630090 Novosibirsk, Russian Federation*
*Email:* itar@iis.nsk.su

*Received 15 September 2011; revised 9 January 2013*

We define a number of stochastic equivalences in the dtsPBC framework, which is a discrete time stochastic extension of finite Petri box calculus (PBC) enriched with iteration. These equivalences allow the identification of stochastic processes that have similar behaviour but are differentiated by the semantics of the calculus. We explain how the equivalences we propose can be used to reduce transition systems of expressions, and demonstrate how to apply the equivalences to compare the stationary behaviour. The equivalences guarantee a coincidence of performance indices for stochastic systems, and can be used for performance analysis simplification. We use a case study to outline a method of modelling, performance evaluation and behaviour preserving reduction of concurrent computing systems, and apply it to the dining philosophers system.

## 1. Introduction

Process algebras (PAs), such as CCS (Milner 1989), are a widely used formal model for specifying concurrent systems and analysing their behavioural properties. In such calculi, processes are specified using compositional formulas constructed by operators from symbols of actions, and the verification of properties is accomplished syntactically by means of algebraic laws and equivalences. In recent decades, a number of stochastic extensions of PAs have been proposed. Unlike standard PAs, stochastic process algebras (SPAs) do not just specify the actions that can occur (qualitative features), but also associate with the actions the distribution parameters of their random time delays (quantitative characteristics). The best-known SPAs are MTIPP (Hermanns and Rettelbach 1994), PEPA (Hillston 1996) and EMPA (Bernardo and Gorrieri 1998).

### 1.1. *Petri box calculus*

PAs specify concurrent systems compositionally using an expressive formal syntax. On the other hand, PNs provide a graphical representation of such systems and capture explicit asynchrony in their behaviour. To combine the advantages of both models, a semantics of algebraic formulas in terms of PNs is defined. Petri box calculus (PBC) (Best

[†] The author was supported in part by Deutsche Forschungsgemeinschaft (DFG), grant 436 RUS 113/1002/01, and the Russian Foundation for Basic Research (RFBR), grant 09-01-91334.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

*et al.* 1992; Best and Koutny 1995) is a flexible and expressive process algebra supporting compositional translations from high-level concurrent programming languages into Petri nets (PNs). The formulas of PBC are not composed from single actions, as in CCS, but from multisets of elementary actions and their conjugates, called multiactions (*basic formulas*). In contrast to CCS, synchronisation is separated from parallelism (*concurrent constructs*). Synchronisation is defined as a unary multi-way stepwise operation based on the communication of actions and their conjugates, thus, it extends the CCS approach with conjugate matching labels. Synchronisation in PBC is asynchronous, unlike that in Synchronous CCS (SCCS) (Milner 1989). The other PBC operations are sequence and choice (*sequential constructs*). The calculus also includes restriction and relabelling (*abstraction constructs*). In order to specify infinite processes, refinement, recursion and iteration operations were also added (*hierarchical constructs*). Thus, unlike CCS, PBC has an additional iteration construction to specify infiniteness when the semantic interpretation in finite PNs is possible. PBC has a step operational semantics in terms of labelled transition systems based on rules in the Structured Operational Semantics (SOS) style. A denotational semantics of PBC has been proposed using a subclass of PNs, called Petri boxes, which are equipped with an interface and considered up to isomorphism. Recently, extensions of PBC with deterministic, non-deterministic and stochastic times have been presented.

### 1.2. *Time extensions of Petri box calculus*

To specify systems with time constraints, such as real-time systems, deterministic (fixed) or non-deterministic (interval) time delays are used. A time extension of PBC with a non-deterministic time model, called time Petri box calculus (tPBC), was proposed in Koutny (2000). In tPBC, timing information is added by associating time intervals (the earliest and latest firing times) with instantaneous *actions*. Its denotational semantics was defined in terms of a subclass of labelled time Petri nets (LtPNs), which are based on tPNs (Merlin and Farber 1976), and called time Petri boxes (ct-boxes). tPBC has an interleaving time operational semantics in terms of labelled transition systems. Another time enrichment of PBC, called timed Petri box calculus (TPBC), was defined in Marroquín and de-Frutos (2001), and accommodates a deterministic model of time. In contrast to tPBC, the multiactions of TPBC are not instantaneous, but have time durations. Additionally, unlike the case with tPBC, there are no 'illegal' multiaction occurrences in TPBC. The complexity of the 'illegal' occurrences mechanism was one of the main motivations in constructing TPBC, though in fact this calculus appeared to be more complicated than tPBC. The denotational semantics of TPBC was defined in terms of a subclass of labelled timed Petri nets (LTPNs), which were based on TPNs (Ramchandani 1973), and called timed Petri boxes (T-boxes). TPBC has a step timed operational semantics in terms of labelled transition systems. Note that tPBC and TPBC differ in how they capture time information, so rather than being competitors, they complement each other. The third time extension of PBC, called arc time Petri box calculus (atPBC), was constructed in Niaouris (2005), and it implements a non-deterministic time. In atPBC, multiactions are associated with time delay intervals. Its

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

denotational semantics was defined on a subclass of arc time Petri nets (atPNs), where time restrictions are associated with the arcs, called arc time Petri boxes (at-boxes). atPBC has a step operational semantics defined in terms of labelled transition systems.

The set of states for the systems with deterministic or non-deterministic delays differs drastically from that for the timeless systems, so the analysis results for timeless systems may not be valid for those with time. To solve this problem, stochastic delays are considered, which are random values with a (discrete or continuous) probability distribution. A continuous time stochastic extension of a finite part of PBC, called stochastic Petri box calculus (sPBC), was proposed in Macià *et al.* (2001). However, since there was no refinement, recursion or iteration operations in the original version of sPBC, it could only specify finite processes. An interleaving operational semantics of the calculus was constructed using labelled probabilistic transition systems. A denotational semantics of sPBC was defined in terms of a subclass of labelled continuous time stochastic PNs (LCTSPNs), which were based on CTSPNs (Marsan 1990), and called stochastic Petri boxes (s-boxes). Macià *et al.* (2004) added the iteration operation to sPBC to specify infinite processes, and also specified a producer/consumer system. In sPBC with iteration, the performance of the processes is evaluated by analysing their underlying continuous time Markov chains (CTMCs). Macià *et al.* (2008) enriched the resulting calculus through the addition of immediate multiactions, and a manufacturing system, together with the AUY-protocol, were modelled. A denotational semantics of such an sPBC extension (we call it generalised sPBC or gsPBC) was defined using a subclass of labelled generalised SPNs (LGSPNs), which were based on GSPNs (Marsan 1990), and called generalised stochastic Petri boxes (gs-boxes). Performance analysis in gsPBC is based on the underlying semi-Markov chains (SMCs). The example systems considered within sPBC and its extensions had an interleaving semantics. The performance indices were only calculated for the systems in Macià *et al.* (2008).

PBC has a step operational semantics, whereas sPBC just has an interleaving one, hence, a stochastic extension of PBC with a step semantics is needed to keep the concurrency degree of behavioural analysis at the same level as in PBC. A discrete time stochastic extension dtsPBC of finite PBC was presented in Tarasyuk (2005) and Tarasyuk (2007). A step operational semantics of the algebra was constructed using labelled probabilistic transition systems. dtsPBC has a denotational semantics in terms of a subclass of labelled discrete time stochastic PNs (LDTSPNs), which are based on DTSPNs (Molloy 1985), and called discrete time stochastic Petri boxes (dts-boxes). A number of stochastic equivalences were proposed to identify stochastic processes that have similar behaviour but are differentiated by the semantic equivalence, and the interrelationships between the equivalences were studied. Tarasyuk (2006) and Tarasyuk (2008) supplemented the syntax of dtsPBC by adding the iteration operator. The performance evaluation in dtsPBC with iteration is accomplished using the underlying discrete time Markov chains (DTMCs) of the algebraic processes. Tarasyuk *et al.* (2010) defined dtsiPBC by extending dtsPBC through the addition of immediate multiactions. The performance analysis in dtsiPBC is based on the underlying semi-Markov chains (SMCs) and (reduced) DTMCs.

Since dtsPBC has a discrete time semantics and geometrically distributed delays in the process states, unlike sPBC with its continuous time semantics and exponentially

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

distributed delays, the calculi apply two different approaches to the stochastic extension of PBC despite some similarities in their syntax and semantics, which are inherited from PBC. The main advantage of dtsPBC is that concurrency is treated naturally, as in PBC, whereas in sPBC, parallelism is simulated by interleaving, obliging the collection of information on the causal independence of activities before constructing the semantics.

Comparing dtsPBC with classical SPAs MTIPP, PEPA, EMPA, the first main difference comes from PBC, since dtsPBC is based on PBC: all algebraic operations and a notion of multiaction are inherited from PBC. The second main difference is the discrete probabilities of activities induced by the discrete time approach, whereas action rates are used in the standard SPAs with continuous time. As a consequence, dtsPBC has a non-interleaving step operational semantics. This is in contrast to the classical SPAs, where concurrency is modelled by interleaving because of the continuous probability distributions of action delays and the race condition applied when several actions can be executed in a state. The salient point of dtsPBC is a combination of discrete stochastic time and step semantics in an SPA.

### 1.3. *Equivalence relations*

A notion of equivalence is important in the theory of computing systems. Equivalences are applied both to compare the behaviours of systems and to reduce their structure. There is a wide diversity of behavioural equivalences, and their interrelations have been well explored in the literature. The best known and most widely used equivalence is bisimulation. Standardly, these equivalences only take into account functional (qualitative) aspects and ignore performance (quantitative) factors. Moreover, the equivalences are usually interleaving ones, that is, they interpret concurrency as sequential non-determinism. To respect the quantitative features of behaviour, probabilistic equivalences have additional requirements on execution probabilities. Two equivalent processes must be able to execute the same sequences of actions, and for every such sequence, its execution probabilities within both processes should coincide. In the case of bisimulation equivalence, the states from which similar future behaviours start are grouped into equivalence classes that form elements of the aggregated state space. From every two bisimilar states, the same actions can be executed, and the subsequent states resulting from the execution of an action belong to the same equivalence class. Moreover, for both states, the cumulative probabilities of moving to the same equivalence class by executing the same action coincide. Markovian equivalences are a different kind of quantitative relation, which are based on the rate (the parameter of the exponential distribution that governs time delays) rather than probability.

Interleaving probabilistic strong bisimulation equivalence was proposed in Larsen and Skou (1991) for labelled probabilistic transition systems. Interleaving Markovian strong bisimulation equivalence was constructed in Hermanns and Rettelbach (1994) for MTIPP, in Hillston (1996) for PEPA and in Bernardo and Gorrieri (1998) for EMPA. Interleaving probabilistic equivalences were defined for probabilistic processes in Jou and Smolka (1990) and van Glabbeek *et al.* (1995). Interleaving probabilistic weak bisimulation equivalence was introduced in Buchholz (1995) for labelled CTSPNs and

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

in Buchholz (1998) for generalised SPNs (GSPNs). Bernardo (2007) compared a variety of interleaving Markovian trace, test and bisimulation equivalences on sequential and concurrent Markovian process calculi. However, no appropriate equivalence notion has been defined yet for concurrent SPAs.

### 1.4. *Contributions of the paper*

In this paper, we discuss a problem of performance preservation by the equivalence notions within dtsPBC enriched with iteration. We begin by presenting the syntax of the calculus, and then describe its operational semantics in terms of labelled transition systems and its denotational semantics based on a subclass of LDTSPNs. We also propose a number of stochastic equivalences and describe how they can be used to reduce transition systems of expressions and the related formalisms, while preserving their qualitative and quantitative behaviour. We also investigate which equivalences guarantee the identity of the stationary behaviours. The resulting property implies a coincidence of performance indices based on the steady-state probabilities of the modelled stochastic systems. The equivalences possessing the property can be used to reduce the state space of a system and thus simplify its performance evaluation, which is usually very complex because of the state space explosion problem. Finally, we present a case study based on the dining philosophers system to explain how to model concurrent computing systems within the calculus, how to analyse their performance and how to reduce the systems to simplify performance evaluation while preserving their performance indices.

Summarising, the main contributions of the paper are:

— a new powerful and expressive discrete time SPA called dtsPBC;
— a step operational semantics of dtsPBC using labelled probabilistic transition systems;
— a Petri net denotational semantics of dtsPBC using discrete time stochastic Petri nets;
— a performance analysis based on the underlying discrete time Markov chains;
— a stochastic equivalence used for reduction that simplifies the performance evaluation;
— an extended case study illustrating how to apply the theoretical results in practice.

### 1.5. *Structure of the paper*

The syntax of dtsPBC is presented in Section 2. Section 3 describes the operational semantics of the calculus and Section 4 presents its denotational semantics. Stochastic algebraic equivalences are then defined and investigated in Section 5. Section 6 explains how to reduce transition systems and the related formalisms modulo the equivalences. Section 7 is devoted to the application of the relations to the stationary behaviour comparison and to determining the performance preserving equivalences. Section 8 describes the specification, performance evaluation and reduction of the dining philosophers system within the calculus. The differences between dtsPBC and other well-known or similar SPAs are considered in Section 9, and the advantages of dtsPBC are explained in Section 10. The concluding Section 11 summarises the results obtained and outlines research perspectives in this area.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

## 2. Syntax

In this section, we define the syntax of the discrete time stochastic extension of finite PBC enriched with iteration, which we call *discrete time stochastic* PBC (dtsPBC).

**Definition 2.1.** Let $X$ be a set. A finite *multiset (bag)* $M$ over $X$ is a mapping $M : X \to \mathbb{N}$ such that $|\{x \in X \mid M(x) > 0\}| < \infty$, that is, it can only contain a finite number of elements.

We denote the *set of all finite multisets* over a set $X$ by $\mathbb{N}_f^X$. The *cardinality* of a multiset $M$ is defined by $|M| = \sum_{x \in X} M(x)$. We write $x \in M$ if $M(x) > 0$ and $M \subseteq M'$ if, for all $x \in X$, we have $M(x) \leqslant M'(x)$. We define

$$(M + M')(x) = M(x) + M'(x)$$
$$(M - M')(x) = \max\{0, M(x) - M'(x)\}.$$

If for all $x \in X$ we have $M(x) \leqslant 1$, then $M$ is a proper set such that $M \subseteq X$.

Let $Act = \{a, b, \ldots\}$ be the set of *elementary actions*. Then $\widehat{Act} = \{\hat{a}, \hat{b}, \ldots\}$ is the set of *conjugated actions (conjugates)* such that $\hat{a} \neq a$ and $\hat{\hat{a}} = a$. Let $\mathcal{A} = Act \cup \widehat{Act}$ be the set of *all actions* and $\mathcal{L} = \mathbb{N}_f^{\mathcal{A}}$ be the set of *all multiactions*. Note that $\emptyset \in \mathcal{L}$, and that this corresponds to an internal activity, that is, the execution of a multiaction that contains no visible action names. The *alphabet* of $\alpha \in \mathcal{L}$ is defined by

$$\mathcal{A}(\alpha) = \{x \in \mathcal{A} \mid \alpha(x) > 0\}.$$

An *activity (stochastic multiaction)* is a pair $(\alpha, \rho)$, where $\alpha \in \mathcal{L}$ and $\rho \in (0; 1)$ is the probability of the multiaction $\alpha$. This probability is interpreted as the probability of independent execution of the stochastic multiaction at the next discrete time moment. Such probabilities are used to calculate the probabilities of executing (possibly empty) multisets of stochastic multiactions after one time unit delay. The probabilities of stochastic multiactions are required not to be equal to 1 to avoid the extra model complexity arising from the need to assign weights to make a choice when several stochastic multiactions with probability 1 can be executed from a state. In this case, some problems arise with conflict resolution – see Molloy (1985) for a discussion on SPNs. This decision also allows us to avoid some technical difficulties related to conditioning events with probability 0. Another reason is that not allowing probability 1 for multiactions excludes a potential source of periodicity (hence, non-ergodicity) in the underlying DTMCs of the algebraic expressions. On the other hand, there is also no sense in allowing zero probabilities for multiactions since that would mean they would never be performed. Let $\mathcal{SL}$ be the set of *all activities*. Note that the same multiaction $\alpha \in \mathcal{L}$ may have different probabilities in the same specification. The *alphabet* of $(\alpha, \rho) \in \mathcal{SL}$ is defined by $\mathcal{A}(\alpha, \rho) = \mathcal{A}(\alpha)$. The *alphabet* of $\Gamma \in \mathbb{N}_f^{\mathcal{SL}}$ is defined by

$$\mathcal{A}(\Gamma) = \cup_{(\alpha,\rho) \in \Gamma} \mathcal{A}(\alpha).$$

For $(\alpha, \rho) \in \mathcal{SL}$, we define its *multiaction part* by $\mathcal{L}(\alpha, \rho) = \alpha$ and its *probability part* by $\Omega(\alpha, \rho) = \rho$. The *multiaction part* of $\Gamma \in \mathbb{N}_f^{\mathcal{SL}}$ is defined by $\mathcal{L}(\Gamma) = \sum_{(\alpha,\rho) \in \Gamma} \alpha$. Remember that sums are considered with the multiplicity when applied to multisets.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Activities are combined into formulas by the following operations:

— *sequential execution* ;
— *choice* []
— *parallelism* ||
— *relabelling* $[f]$ of actions
— *restriction* rs over a single action
— *synchronisation* sy on an action and its conjugate
— *iteration* $[**]$ with the three arguments initialisation, body and termination.

Sequential execution and choice have a standard interpretation, as in other process algebras, but parallelism does not include synchronisation, unlike the corresponding operation in CCS.

Relabelling functions $f : \mathcal{A} \to \mathcal{A}$ are bijections that preserve conjugates, that is, for all $x \in \mathcal{A}$, we have $f(\hat{x}) = \widehat{f(x)}$. Relabelling is extended to multiactions in the usual way: for $\alpha \in \mathcal{L}$, we define $f(\alpha) = \sum_{x \in \alpha} f(x)$. Relabelling is extended to the multisets of activities as follows: for $\Gamma \in \mathbb{N}_f^{\mathcal{SL}}$, we define

$$f(\Gamma) = \sum_{(\alpha,\rho)\in\Gamma} (f(\alpha), \rho).$$

Restriction over an elementary action $a \in Act$ means that, for a given expression, any process behaviour containing $a$ or its conjugate $\hat{a}$ is forbidden.

Let $\alpha, \beta \in \mathcal{L}$ be two multiactions such that for some elementary action $a \in Act$, we have $a \in \alpha$ and $\hat{a} \in \beta$, or $\hat{a} \in \alpha$ and $a \in \beta$. The synchronisation of $\alpha$ and $\beta$ by $a$ is then defined by $\alpha \oplus_a \beta = \gamma$, where

$$\gamma(x) = \begin{cases} \alpha(x) + \beta(x) - 1 & \text{if } x = a \text{ or } x = \hat{a} \\ \alpha(x) + \beta(x) & \text{otherwise.} \end{cases}$$

In other words, we require that $\alpha \oplus_a \beta = \alpha + \beta - \{a, \hat{a}\}$, that is, we remove one exemplar of $a$ and one exemplar of $\hat{a}$ from the multiset sum $\alpha + \beta$ since the synchronisation of $a$ and $\hat{a}$ produces $\emptyset$.

In iteration, the initialisation subprocess is executed first, then the body is performed zero or more times, and then, finally, the termination is executed.

Static expressions specify the structure of processes. The expressions correspond to unmarked LDTSPNs (note that LDTSPNs are marked by definition). Remember that a marking is the allocation of tokens in the places of a PN, and the markings are used to describe the dynamic behaviour of PNs in terms of transition firings.

**Definition 2.2.** Let $(\alpha, \rho) \in \mathcal{SL}$, $a \in Act$. A *static expression* of dtsPBC is defined by the grammar

$$E ::= (\alpha, \rho) \mid E;E \mid E[]E \mid E\|E \mid E[f] \mid E \text{ rs } a \mid E \text{ sy } a \mid [E * E * E].$$

*StatExpr* denotes the set of *all the static expressions* of dtsPBC.

To make this grammar unambiguous, we could either add parentheses in the productions with binary operations, for example, $(E;E)$, $(E[]E)$, $(E\|E)$, or associate priorities with

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

operations. We prefer the PBC approach, which is just to add parentheses to resolve ambiguities, and we assume no priorities.

To avoid some technical difficulties with the iteration operator, we will not allow any concurrency at the highest level of the second argument of iteration. However, this is not a severe restriction since we can always prefix parallel expressions by an activity with the empty multiaction. In Example 4.9, we shall demonstrate that relaxing the restriction can result in nets that are not safe. Alternatively, we could use a different, safe, version of the iteration operator, but its net translation has six arguments – see Best *et al.* (2001) for further discussion on this subject. Remember that a PN is *n-bounded* ($n \in I\!N$) if for all its reachable (that is, reachable from the initial marking by the sequences of transition firings) markings, there are at most $n$ tokens in every place, and a PN is *safe* if it is 1-bounded.

**Definition 2.3.** Let $(\alpha, \rho) \in \mathcal{SL}$, $a \in Act$. A *regular static expression* of dtsPBC is defined by the grammar

$$E ::= (\alpha,\rho) \mid E;E \mid E[]E \mid E\|E \mid E[f] \mid E \text{ rs } a \mid E \text{ sy } a \mid [E * D * E],$$
$$\text{where } D ::= (\alpha,\rho) \mid D;E \mid D[]D \mid D[f] \mid D \text{ rs } a \mid D \text{ sy } a \mid [D * D * E].$$

*RegStatExpr* denotes the set of *all the regular static expressions* of dtsPBC.

Dynamic expressions specify the states of processes and correspond to LDTSPNs (marked by default). Dynamic expressions are obtained from static expressions, which are annotated with upper or lower bars to specify the active components of the system at the current time instant. The dynamic expression with an upper bar (the overlined one) $\overline{E}$ denotes the *initial*, and the expression with a lower bar (the underlined one) $\underline{E}$ denotes the *final* state of the process specified by a static expression $E$. The *underlying static expression* of a dynamic expression is obtained by removing all the upper and lower bars from it.

**Definition 2.4.** Let $E \in StatExpr$, $a \in Act$. A *dynamic expression* of dtsPBC is defined by the grammar

$$G ::= \overline{E} \mid \underline{E} \mid G;E \mid E;G \mid G[]E \mid E[]G \mid G\|G \mid G[f] \mid G \text{ rs } a \mid G \text{ sy } a \mid$$
$$[G * E * E] \mid [E * G * E] \mid [E * E * G].$$

*DynExpr* denotes the set of *all the dynamic expressions* of dtsPBC.

If the underlying static expression of a dynamic expression is not regular, then the corresponding LDTSPN can be non-safe, as Example 4.9 will show (however, the LDTSPN is 2-bounded in the worst case (Best *et al.* 2001)).

**Definition 2.5.** A dynamic expression is *regular* if its underlying static expression is regular.

*RegDynExpr* denotes the set of *all the regular dynamic expressions* of dtsPBC.

## 3. Operational semantics

In this section, we define the step operational semantics using labelled transition systems – we present an example at the end of the section.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Table 1. *Inaction rules for overlined and underlined regular static expressions.*

| | | |
|---|---|---|
| $\overline{E;F} \Rightarrow \overline{E};F$ | $\underline{E};F \Rightarrow E;\overline{F}$ | $E;\underline{F} \Rightarrow \underline{E;F}$ |
| $\overline{E[]F} \Rightarrow \overline{E}[]F$ | $\overline{E[]F} \Rightarrow E[]\overline{F}$ | $\underline{E}[]F \Rightarrow \underline{E[]F}$ |
| $E[]\underline{F} \Rightarrow \underline{E[]F}$ | $\overline{E\|F} \Rightarrow \overline{E}\|\overline{F}$ | $\underline{E}\|\underline{F} \Rightarrow \underline{E\|F}$ |
| $\overline{E[f]} \Rightarrow \overline{E}[f]$ | $\underline{E}[f] \Rightarrow \underline{E[f]}$ | $\overline{E \text{ rs } a} \Rightarrow \overline{E} \text{ rs } a$ |
| $\underline{E} \text{ rs } a \Rightarrow \underline{E \text{ rs } a}$ | $\overline{E \text{ sy } a} \Rightarrow \overline{E} \text{ sy } a$ | $\underline{E} \text{ sy } a \Rightarrow \underline{E \text{ sy } a}$ |
| $\overline{[E * F * K]} \Rightarrow [\overline{E} * F * K]$ | $[\underline{E} * F * K] \Rightarrow [E * \overline{F} * K]$ | $[E * \underline{F} * K] \Rightarrow [E * \overline{F} * K]$ |
| $[E * \underline{F} * K] \Rightarrow [E * F * \overline{K}]$ | $[E * F * \underline{K}] \Rightarrow \underline{[E * F * K]}$ | |

Table 2. *Inaction rules for arbitrary regular dynamic expressions.*

| | | |
|---|---|---|
| $\dfrac{G \Rightarrow \widetilde{G},\ \circ \in \{;, []\}}{G \circ E \Rightarrow \widetilde{G} \circ E}$ | $\dfrac{G \Rightarrow \widetilde{G},\ \circ \in \{;, []\}}{E \circ G \Rightarrow E \circ \widetilde{G}}$ | $\dfrac{G \Rightarrow \widetilde{G}}{G\|H \Rightarrow \widetilde{G}\|H}$ |
| $\dfrac{H \Rightarrow \widetilde{H}}{G\|H \Rightarrow G\|\widetilde{H}}$ | $\dfrac{G \Rightarrow \widetilde{G}}{G[f] \Rightarrow \widetilde{G}[f]}$ | $\dfrac{G \Rightarrow \widetilde{G},\ \circ \in \{\text{rs}, \text{sy}\}}{G \circ a \Rightarrow \widetilde{G} \circ a}$ |
| $\dfrac{G \Rightarrow \widetilde{G}}{[G * E * F] \Rightarrow [\widetilde{G} * E * F]}$ | $\dfrac{G \Rightarrow \widetilde{G}}{[E * G * F] \Rightarrow [E * \widetilde{G} * F]}$ | $\dfrac{G \Rightarrow \widetilde{G}}{[E * F * G] \Rightarrow [E * F * \widetilde{G}]}$ |

### 3.1. *Inaction rules*

The inaction rules for dynamic expressions describe the structural transformations that do not change the states of the specified processes. The goal of these syntactic transformations is to obtain the well-structured terminal expressions, called operative expressions, to which no inaction rules can be further applied. As we shall see, the application of an inaction rule to a dynamic expression does not lead to any discrete time step in the corresponding LDTSPN, so no transitions fire, and its current marking remains unchanged.

Thus, an application of any inaction rule requires no time delay, that is, the dynamic expression transformation described by the rule is accomplished instantly.

Table 1 defines the inaction rules for regular dynamic expressions in the form of overlined and underlined static expressions. In this table, $E, F, K \in RegStatExpr$ and $a \in Act$.

Table 2 defines the inaction rules for arbitrary regular dynamic expressions. In this table, $E, F \in RegStatExpr,\ G, H, \widetilde{G}, \widetilde{H} \in RegDynExpr$ and $a \in Act$.

**Definition 3.1.** A regular dynamic expression $G$ is *operative* if no inaction rule can be applied to it.

$OpRegDynExpr$ denotes the set of *all the operative regular dynamic expressions* of dtsPBC. Any regular dynamic expression can be transformed into a (possibly not unique)

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

operative one using the inaction rules. Since in the following we shall only consider regular expressions, the word 'regular' will be omitted from now on.

**Definition 3.2.** Let $\approx = (\Rightarrow \cup \Leftarrow)^*$ be a structural equivalence of the dynamic expressions in dtsPBC. Thus, two dynamic expressions $G$ and $G'$ are *structurally equivalent*, denoted by $G \approx G'$, if they can be reached from each other by applying the inaction rules in a forward or backward direction.

### 3.2. *Action and empty loop rules*

The action rules are applied when some activities are executed. We also have the empty loop rule, which is used to capture a delay of one time unit in the same state when the empty multiset of activities is executed. The action and empty loop rules will be used later to determine all multisets of activities that can be executed from the structural equivalence class of any dynamic expression (that is, from the state of the corresponding process). This information, together with the information about the probabilities of the activities to be executed from the process state, will be used to calculate the probabilities of such executions.

The action rules describe dynamic expression transformations due to the execution of non-empty multisets of activities. The rules represent possible state changes of the specified processes when some non-empty multisets of activities are executed. As we shall see, the application of an action rule to a dynamic expression leads to a discrete time step in the corresponding LDTSPN at which some transitions fire and the current marking is changed, unless there is a self-loop produced by the iterative execution of a non-empty multiset (which should have one element, that is, the single activity, since we do not allow concurrency at the highest level of the second argument of iteration).

The empty loop rule $G \stackrel{\emptyset}{\rightarrow} G$ describes dynamic expression transformations due to the execution of the empty multiset of activities at a discrete time step. The rule reflects a non-zero probability of staying in the current state at the next time moment, which is an essential feature of discrete time stochastic processes. As we shall see, the application of the empty loop rule to a dynamic expression leads to a discrete time step in the corresponding LDTSPN at which no transitions fire and the current marking is not changed. This is a new rule, which has no prototype among the inaction rules of PBC since it represents a time delay. In our setting, the PBC rule $G \stackrel{\emptyset}{\rightarrow} G$ from Best *et al.* (2001) would correspond to the rule $G \Rightarrow G$ describing the stay in the current state when no time elapses. Since we do not need the latter rule to transform dynamic expressions into operative ones, and it may even destroy the definition of operative expressions, we do not include it in dtsPBC.

Thus, an application of any action rule or the empty loop rule requires one discrete time unit delay, that is, the execution of a (possibly empty) multiset of activities resulting in the dynamic expression transformation described by the rule is accomplished instantly after one unit of time elapses.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Table 3. *Action and empty loop rules.*

| | | |
|---|---|---|
| **El** $G \xrightarrow{\emptyset} G$ | **B** $\overline{(\alpha,\rho)} \xrightarrow{\{(\alpha,\rho)\}} \underline{(\alpha,\rho)}$ | |
| **SC1** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G},\ \circ \in \{;,[]\}}{G \circ E \xrightarrow{\Gamma} \widetilde{G} \circ E}$ | **SC2** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G},\ \circ \in \{;,[]\}}{E \circ G \xrightarrow{\Gamma} E \circ \widetilde{G}}$ | |
| **P1** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G}}{G \| H \xrightarrow{\Gamma} \widetilde{G} \| H}$ | **P2** $\dfrac{H \xrightarrow{\Gamma} \widetilde{H}}{G \| H \xrightarrow{\Gamma} G \| \widetilde{H}}$ | **P3** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G},\ H \xrightarrow{\Delta} \widetilde{H}}{G \| H \xrightarrow{\Gamma+\Delta} \widetilde{G} \| \widetilde{H}}$ |
| **L** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G}}{G[f] \xrightarrow{f(\Gamma)} \widetilde{G}[f]}$ | **Rs** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G},\ a, \hat{a} \notin \mathcal{A}(\Gamma)}{G \text{ rs } a \xrightarrow{\Gamma} \widetilde{G} \text{ rs } a}$ | |
| **I1** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G}}{[G * E * F] \xrightarrow{\Gamma} [\widetilde{G} * E * F]}$ | **I2** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G}}{[E * G * F] \xrightarrow{\Gamma} [E * \widetilde{G} * F]}$ | **I3** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G}}{[E * F * G] \xrightarrow{\Gamma} [E * F * \widetilde{G}]}$ |
| **Sy1** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G}}{G \text{ sy } a \xrightarrow{\Gamma} \widetilde{G} \text{ sy } a}$ | **Sy2** $\dfrac{G \text{ sy } a \xrightarrow{\Gamma'+\{(\alpha,\rho)\}+\{(\beta,\chi)\}} \widetilde{G} \text{ sy } a,\ a \in \alpha,\ \hat{a} \in \beta}{G \text{ sy } a \xrightarrow{\Gamma'+\{(\alpha \oplus_a \beta, \rho \cdot \chi)\}} \widetilde{G} \text{ sy } a}$ | |

Table 3 defines the action and empty loop rules. In this table,

$$\begin{aligned}(\alpha,\rho),(\beta,\chi) &\in \mathcal{SL}\\ E,F &\in RegStatExpr\\ G,H &\in OpRegDynExpr\\ \widetilde{G},\widetilde{H} &\in RegDynExpr\\ a &\in Act.\end{aligned}$$

Moreover,

$$\begin{aligned}\Gamma,\Delta &\in I\!N_f^{\mathcal{SL}} \setminus \{\emptyset\}\\ \Gamma' &\in I\!N_f^{\mathcal{SL}}.\end{aligned}$$

Almost all of the rules in Table 3 (the exceptions are **El**, **P3** and **Sy2**) resemble those of sPBC (Macià *et al.* 2001), though the rules in Table 3 correspond to the execution of multisets of activities, and not single activities as in sPBC.

Rule **El** corresponds to one discrete time unit delay while executing no activities, so it has no analogues among the rules of sPBC, which adopts the continuous time model.

Rule **P3** has no analogous rules in sPBC since the interleaving semantics of the algebra forbids the simultaneous execution of activities. On the other hand, **P3** does have the analogous rule **PAR** in PBC, which is used to construct the step semantics of the calculus, though **P3** corresponds to the execution of multisets of activities, as opposed to multisets of multiactions as in the **PAR** rule.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Rule **Sy2** differs from the corresponding synchronisation rule in sPBC since the probability of synchronisation in the **Sy2** rule and the rate of synchronisation in the sPBC rule are calculated differently.

Rule **Sy2** says that the synchronisation of two stochastic multiactions is achieved by taking the product of their probabilities since we consider that both must occur for the synchronisation to happen. So this corresponds, in some sense, to the probability of the intersection of the independent events, though the real situation is more complex since these stochastic multiactions can be also executed in parallel. Nevertheless, when scoping (the combined operation consisting of synchronisation followed by restriction over the same action (Best *et al.* 2001)) is applied over a parallel execution, we just get the simple product of the probabilities as the final result since no normalisation is required. Multiplication is an associative and commutative binary operation that is distributive over addition, that is, it fulfills all the practical conditions imposed on the synchronisation operator in Hillston (1994). Furthermore, if both arguments of the multiplication lie in $(0;1)$, the result belongs to the same interval, so multiplication naturally maintains probabilistic compositionality in our model. Our approach is similar to the multiplication of the rates of the synchronised actions in MTIPP (Hermanns and Rettelbach 1994) for the case where the rates are less than 1. Moreover, for the probabilities $\rho$ and $\chi$ of two stochastic multiactions to be synchronised, we have $\rho \cdot \chi < \min\{\rho, \chi\}$, that is, multiplication meets the performance requirement stating that the probability of the resulting synchronised stochastic multiaction should be less than the probabilities of each of the actions to be synchronised. In performance evaluation, it is usually assumed that the execution of two components together requires more system resources and time than the execution of each of them independently. This resembles the *bounded capacity* assumption in Hillston (1994). Thus, multiplication is easy to handle and satisfies the algebraic, probabilistic, time and performance requirements. For these reasons, we have chosen to use the product of the probabilities for the synchronisation. See also Brinksma *et al.* (1995) and Brinksma and Hermanns (2001) for a discussion of the use of binary operations in producing the rates of synchronisation in the continuous time setting.

As we shall see, for every LDTSPN obtained by synchronisation of two LDTSPNs, this approach allows us to calculate the transition firing probabilities using the standard transition probability function for that net class. If concurrency aspects are irrelevant, interleaving semantics is used, and this abstracts from steps with more than one element. After the abstraction, the probabilities of the remaining single-element steps are normalised to keep the sums of outgoing probabilities equal to one. For two synchronised LDTSPNs, our approach allows us to extract the interleaving probabilities from the step probabilities in the same way as for two non-synchronised parallel LDTSPNs.

We do not allow self-synchronisation, that is, the synchronisation of an activity with itself, to avoid unexpected behaviour and many technical difficulties (Best *et al.* 2001).

### 3.3. *Transition systems*

We will now construct the labelled probabilistic transition systems associated with dynamic expressions, which will be used to define the operational semantics of dtsPBC.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

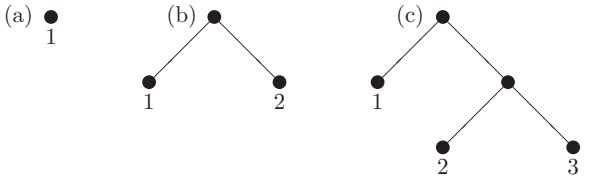



Fig. 1. The binary trees encoded with the numberings 1, (1)(2) and (1)((2)(3)).

The expressions of dtsPBC can contain identical activities. To avoid technical difficulties, we must enumerate coinciding activities, for instance, from left to right in the syntax of expressions. The new activities resulting from synchronisation will be annotated with the concatenation of the numberings of the activities they come from, so the numbering should have a tree structure to reflect the effect of multiple synchronisations. We will now define the numbering that encodes a binary tree, with the leaves labelled by natural numbers.

**Definition 3.3.** Let $n \in \mathbb{N}$. The *numbering* of expressions is defined by $\iota ::= n \mid (\iota)(\iota)$.

*Num* denotes the set of *all numberings* of expressions.

**Example 3.4.** The numbering 1 encodes the binary tree in Figure 1(a) with the root labelled by 1. The numbering (1)(2) corresponds to the binary tree in Figure 1(b) without internal nodes and with two leaves labelled by 1 and 2. The numbering (1)((2)(3)) represents the binary tree in Figure 1(c) with one internal node, which is the root for the subtree (2)(3), and three leaves labelled by 1, 2 and 3.

The new activities resulting from applications of the second rule for synchronisation **Sy2** in different orders should be considered up to permutation of their numbering. In this way, we can recognise different instances of the same activity. If we compare the contents of different numberings, that is, the sets of natural numbers in them, we can identify the mentioned instances. The *content* of a numbering $\iota \in Num$ is defined by

$$Cont(\iota) = \begin{cases} \{\iota\} & \text{if } \iota \in \mathbb{N} \\ Cont(\iota_1) \cup Cont(\iota_2) & \text{if } \iota = (\iota_1)(\iota_2). \end{cases}$$

After we have applied the enumeration, the multisets of activities from the expressions become the proper sets. In the following, we assume that the identical activities are enumerated when needed to avoid ambiguity. This enumeration is considered to be implicit.

**Definition 3.5.** Let $G$ be a dynamic expression. Then $[G]_{\approx} = \{H \mid G \approx H\}$ is the equivalence class of $G$ with respect to the structural equivalence. The *derivation set* of a dynamic expression $G$, denoted by $DR(G)$, is the minimal set such that:

— $[G]_{\approx} \in DR(G)$;

— if $[H]_{\approx} \in DR(G)$ and there exists $\Gamma$ such that $H \xrightarrow{\Gamma} \widetilde{H}$, then $[\widetilde{H}]_{\approx} \in DR(G)$.

Let $G$ be a dynamic expression and $s, \tilde{s} \in DR(G)$.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

The set of *all multisets of activities executable in* $s$ is

$$Exec(s) = \{\Gamma \mid \exists H \in s \; \exists \widetilde{H} H \xrightarrow{\Gamma} \widetilde{H}\}.$$

Let $\Gamma \in Exec(s) \setminus \{\emptyset\}$. The *probability that the multiset of activities* $\Gamma$ *is ready for execution in* $s$ is

$$PF(\Gamma, s) = \prod_{(\alpha,\rho)\in\Gamma} \rho \cdot \prod_{\{\{(\beta,\chi)\}\in Exec(s) \mid (\beta,\chi)\notin\Gamma\}} (1-\chi).$$

When $\Gamma = \emptyset$, we define

$$PF(\emptyset, s) = \begin{cases} \prod_{\{(\beta,\chi)\}\in Exec(s)}(1-\chi) & \text{if } Exec(s) \neq \{\emptyset\} \\ 1 & \text{otherwise.} \end{cases}$$

Thus, if $Exec(s) \neq \{\emptyset\}$, then $PF(\Gamma, s)$ can be interpreted as a *joint* probability of independent events. Each such event consists of the positive or negative decision to execute a particular activity. Every executable activity decides probabilistically (using its probabilistic part) and independently (from others) if it wants to be executed in $s$. If $\Gamma$ is a multiset of all executable activities that have decided to be executed in $s$ and $\Gamma \in Exec(s)$, then $\Gamma$ is ready for execution in $s$. The multiplication in the definition is used because it reflects the probability of the independent event intersection. Alternatively, when $\Gamma \neq \emptyset$, $PF(\Gamma, s)$ can be interpreted as the probability to execute *exclusively* the multiset of activities $\Gamma$ in $s$, that is, the probability of the *intersection* of two events calculated using the conditional probability formula in the form

$$\mathsf{P}(X \cap Y) = \mathsf{P}(X|Y)\mathsf{P}(Y).$$

The event $X$ consists of the execution of $\Gamma$ in $s$. The event $Y$ consists of the non-execution in $s$ of all the executable activities not belonging to $\Gamma$. Since these non-executions are obviously independent events, the probability of $Y$ is a product of the probabilities of the non-executions:

$$\mathsf{P}(Y) = \prod_{\{\{(\beta,\chi)\}\in Exec(s) \mid (\beta,\chi)\notin\Gamma\}} (1-\chi).$$

The conditioning of $X$ by $Y$ makes the executions of the activities from $\Gamma$ independent since all of them can be executed in parallel in $s$ by the definition of $Exec(s)$. Hence, the probability of executing $\Gamma$ *under the condition* that no executable activities not belonging to $\Gamma$ are executed in $s$ is the product of probabilities of these activities:

$$\mathsf{P}(X|Y) = \prod_{(\alpha,\rho)\in\Gamma} \rho.$$

Thus, the probability that $\Gamma$ is executed *and* no executable activities not belonging to $\Gamma$ are executed in $s$ is the probability of $X$ conditioned by $Y$ multiplied by the probability of $Y$:

$$\mathsf{P}(X \cap Y) = \mathsf{P}(X|Y)\mathsf{P}(Y) = \prod_{(\alpha,\rho)\in\Gamma} \rho \cdot \prod_{\{\{(\beta,\chi)\}\in Exec(s) \mid (\beta,\chi)\notin\Gamma\}} (1-\chi).$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

When $\Gamma = \emptyset$, $PF(\Gamma, s)$ can be interpreted as the probability not to execute in $s$ any executable activities, thus,

$$PF(\emptyset, s) = \prod_{\{(\beta,\chi)\} \in Exec(s)} (1 - \chi).$$

When only the empty multiset of activities can be executed in $s$, that is, $Exec(s) = \{\emptyset\}$, we have $PF(\emptyset, s) = 1$ since we stay in $s$ in this case.

Let $\Gamma \in Exec(s)$. In addition to $\Gamma$, some other multisets of activities may be ready for execution in $s$, hence, a kind of conditioning or normalisation is needed to calculate the execution probability. The *probability to execute the multiset of activities* $\Gamma$ *in* $s$ is

$$PT(\Gamma, s) = \frac{PF(\Gamma, s)}{\sum_{\Delta \in Exec(s)} PF(\Delta, s)}.$$

Thus, $PT(\Gamma, s)$ can be interpreted as the *conditional* probability to execute $\Gamma$ in $s$ calculated using the conditional probability formula in the form

$$\mathsf{P}(Z \,|\, W) = \frac{\mathsf{P}(Z \cap W)}{\mathsf{P}(W)}.$$

The event $Z$ consists of the exclusive execution of $\Gamma$ in $s$, hence, $\mathsf{P}(Z) = PF(\Gamma, s)$. The event $W$ consists of the exclusive execution of any multiset (including the empty one) $\Delta \in Exec(s)$ in $s$. Thus, $W = \cup_j Z_j$, where for all $j$, the $Z_j$ are mutually exclusive events and there exists $i$ such that $Z = Z_i$. We have

$$\begin{aligned} \mathsf{P}(W) &= \sum_j \mathsf{P}(Z_j) \\ &= \sum_{\Delta \in Exec(s)} PF(\Delta, s) \end{aligned}$$

because summation reflects the probability of the mutually exclusive event union. Since

$$\begin{aligned} Z \cap W &= Z_i \cap (\cup_j Z_j) \\ &= Z_i \\ &= Z, \end{aligned}$$

we have

$$\begin{aligned} \mathsf{P}(Z \,|\, W) &= \frac{\mathsf{P}(Z)}{\mathsf{P}(W)} \\ &= \frac{PF(\Gamma, s)}{\sum_{\Delta \in Exec(s)} PF(\Delta, s)}. \end{aligned}$$

$PF(\Gamma, s)$ can be also viewed as the *potential* probability to execute $\Gamma$ in $s$ since we have $PF(\Gamma, s) = PT(\Gamma, s)$ only when *all* multisets (including the empty one) consisting of the executable activities can be executed in $s$. In this case, all the mentioned activities can be executed in parallel in $s$ and we have

$$\sum_{\Delta \in Exec(s)} PF(\Delta, s) = 1$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

since this sum collects the products of *all* combinations of the probability parts of the activities and the negations of these parts. But, in general, for example, for two activities $(\alpha, \rho)$ and $(\beta, \chi)$ executable in $s$, it may happen that they cannot be executed in $s$ together, in parallel, that is,

$$\emptyset, \{(\alpha, \rho)\}, \{(\beta, \chi)\} \in Exec(s),$$

but

$$\{(\alpha, \rho), (\beta, \chi)\} \notin Exec(s).$$

Note that $PT(\emptyset, s) \in (0; 1]$, so there is a non-zero probability of staying in the state $s$ at the next time moment, and the residence time in $s$ is at least 1 discrete time unit.

Note that the sum of outgoing probabilities for the expressions belonging to the derivations of $G$ is equal to 1, that is, for all $s \in DR(G)$, we have

$$\sum_{\Gamma \in Exec(s)} PT(\Gamma, s) = 1.$$

This follows from the definition of $PT(\Gamma, s)$ and guarantees that $PT(\Gamma, s)$ defines a probability distribution.

The *probability of moving from s to* $\tilde{s}$ *by executing any multiset of activities* is

$$PM(s, \tilde{s}) = \sum_{\{\Gamma \mid \exists H \in s \ \exists \widetilde{H} \in \tilde{s} \ H \stackrel{\Gamma}{\rightarrow} \widetilde{H}\}} PT(\Gamma, s).$$

**Example 3.6.** Let $E = (\{a\}, \rho)[](\{a\}, \chi)$. Now $DR(\overline{E})$ consists of the equivalence classes $s_1 = [\overline{E}]_{\approx}$ and $s_2 = [\underline{E}]_{\approx}$. The execution probabilities are calculated as follows. Since

$$Exec(s_1) = \{\emptyset, \{(\{a\}, \rho)\}, \{(\{a\}, \chi)\}\},$$

we get

$$PF(\{(\{a\}, \rho)\}, s_1) = \rho(1 - \chi)$$
$$PF(\{(\{a\}, \chi)\}, s_1) = \chi(1 - \rho)$$
$$PF(\emptyset, s_1) = (1 - \rho)(1 - \chi).$$

Then

$$\sum_{\Delta \in Exec(s_1)} PF(\Delta, s_1) = \rho(1 - \chi) + \chi(1 - \rho) + (1 - \rho)(1 - \chi)$$
$$= 1 - \rho\chi.$$

Thus,

$$PT(\{(\{a\}, \rho)\}, s_1) = \frac{\rho(1 - \chi)}{1 - \rho\chi}$$
$$PT(\{(\{a\}, \chi)\}, s_1) = \frac{\chi(1 - \rho)}{1 - \rho\chi}$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

and

$$\begin{aligned} PT(\emptyset, s_1) &= PM(s_1, s_1) \\ &= \frac{(1-\rho)(1-\chi)}{1-\rho\chi}. \end{aligned}$$

Furthermore, $Exec(s_2) = \{\emptyset\}$, so

$$\begin{aligned} \sum_{\Delta \in Exec(s_2)} PF(\Delta, s_2) &= PF(\emptyset, s_2) \\ &= 1 \end{aligned}$$

and

$$\begin{aligned} PT(\emptyset, s_2) &= PM(s_2, s_2) \\ &= \tfrac{1}{1} \\ &= 1. \end{aligned}$$

Finally,

$$\begin{aligned} PM(s_1, s_2) &= PT(\{(\{a\}, \rho)\}, s_1) + PT(\{(\{a\}, \chi)\}, s_1) \\ &= \frac{\rho(1-\chi)}{1-\rho\chi} + \frac{\chi(1-\rho)}{1-\rho\chi} \\ &= \frac{\rho + \chi - 2\rho\chi}{1-\rho\chi}. \end{aligned}$$

**Definition 3.7.** Let $G$ be a dynamic expression. The *(labelled probabilistic) transition system* of $G$ is a quadruple $TS(G) = (S_G, L_G, \mathcal{T}_G, s_G)$, where:

— the set of *states* is $S_G = DR(G)$;
— the set of *labels* is $L_G \subseteq \mathbb{N}_f^{\mathcal{SL}} \times (0; 1]$;
— the set of *transitions* is

$$\mathcal{T}_G = \{(s, (\Gamma, PT(\Gamma, s)), \tilde{s}) \mid s \in DR(G),\ \exists H \in s\ \exists \widetilde{H} \in \tilde{s}\ H \xrightarrow{\Gamma} \widetilde{H}\};$$

— the *initial state* is $s_G = [G]_{\approx}$.

The definition of $TS(G)$ is correct, that is, for every state, the sum of the probabilities of all the transitions starting from it is 1. This is guaranteed by the note following the definition of $PT(\Gamma, s)$. Thus, we have defined a *generative* model of probabilistic processes (Jou and Smolka 1990) according to the classification in van Glabbeek *et al.* (1995). This is because the sum of the probabilities of the transitions with *all* possible labels should be equal to 1 (and not just the sum of those with the same labels, up to enumeration of the activities they include, as in *reactive* models (Larsen and Skou 1991)), and we do not have nested probabilistic choice as in *stratified* models (van Glabbeek *et al.* 1995).

The transition system $TS(G)$ of a dynamic expression $G$ describes all steps that occur at discrete time moments with some (one-step) probability and that consist of multisets of activities. Every step occurs instantly after one discrete time unit delay, and the step

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

can change the current state to another one. The states are the structural equivalence classes of dynamic expressions obtained by the application of action rules starting from the expressions belonging to $[G]_{\approx}$. A transition $(s, (\Gamma, \mathcal{P}), \tilde{s}) \in \mathcal{T}_G$ is written

$$s \xrightarrow{\Gamma}_{\mathcal{P}} \tilde{s}$$

and interpreted as 'the probability of changing the state $s$ to $\tilde{s}$ by executing $\Gamma$ is $\mathcal{P}$'.

Note that $\Gamma$ can be the empty multiset, and its execution then does not change the current state (the equivalence class) since we have a loop transition

$$s \xrightarrow{\emptyset}_{\mathcal{P}} s$$

from a state $s$ to itself as a result of executing the empty multiset. This corresponds to the application of the empty loop rule to expressions from the equivalence class. We have to keep track of such executions, called *empty loops*, because they have non-zero probabilities. This follows from the definition of $PF(\emptyset, s)$ and the fact that multiaction probabilities cannot be equal to 1 since they belong to the interval $(0; 1)$. The step probabilities belong to the interval $(0; 1]$. The step probability is 1 when we cannot leave a state $s$, so there exists only one transition from it, namely, the empty loop transition

$$s \xrightarrow{\emptyset}_{1} s.$$

We write

$$s \xrightarrow{\Gamma} \tilde{s}$$

if there exists $\mathcal{P}$ such that

$$s \xrightarrow{\Gamma}_{\mathcal{P}} \tilde{s},$$

and

$$s \rightarrow \tilde{s}$$

if there exists $\Gamma$ such that

$$s \xrightarrow{\Gamma} \tilde{s}.$$

For a one-element multiset of activities $\Gamma = \{(\alpha, \rho)\}$, we write

$$s \xrightarrow{(\alpha,\rho)}_{\mathcal{P}} \tilde{s}$$

and

$$s \xrightarrow{(\alpha,\rho)} \tilde{s}.$$

Isomorphism is a coincidence of systems up to the renaming of their components. Let $\simeq$ denote isomorphism between transition systems that binds their initial states.

**Definition 3.8.** Dynamic expressions $G$ and $G'$ are *equivalent with respect to transition systems*, denoted by $G =_{ts} G'$, if $TS(G) \simeq TS(G')$.

**Definition 3.9.** Let $G$ be a dynamic expression. The *underlying discrete time Markov chain (DTMC)* of $G$, denoted by $DTMC(G)$, has state space $DR(G)$, initial state $[G]_{\approx}$ and transitions $s \rightarrow_{\mathcal{P}} \tilde{s}$ if $s \rightarrow \tilde{s}$ and $\mathcal{P} = PM(s, \tilde{s})$.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

For a dynamic expression $G$, a discrete random variable is associated with every state of $DTMC(G)$. The variable captures a residence time in the state. We can interpret staying in a state at the next discrete time moment as a failure of some trial series, and leaving it as a success. It is easy to see that the random variables are geometrically distributed since the probability of staying in the state $s \in DR(G)$ for $k-1$ time moments and leaving it at the moment $k \geqslant 1$ is

$$PM(s,s)^{k-1}(1-PM(s,s))$$

(the residence time is $k$ in this case). The mean value formula for the geometrical distribution allows us to calculate the *average sojourn time in the state s* as

$$SJ(s) = \frac{1}{1-PM(s,s)}.$$

The *average sojourn time vector* of $G$, denoted by $SJ$, has elements $SJ(s)$, $s \in DR(G)$. Analogously, the *sojourn time variance in the state s* is

$$VAR(s) = \frac{PM(s,s)}{(1-PM(s,s))^2}.$$

The *sojourn time variance vector* of $G$, denoted by $VAR$, has elements $VAR(s)$, $s \in DR(G)$.

**Example 3.10.** Let

$$E_1 = (\{a\},\rho)[](\{a\},\rho)$$
$$E_2 = (\{b\},\chi)$$
$$E_3 = (\{c\},\theta)$$

and

$$E = [E_1 * E_2 * E_3].$$

The identical activities of the composite static expression are enumerated as follows:

$$E = [((\{a\},\rho)_1[](\{a\},\rho)_2) * (\{b\},\chi) * (\{c\},\theta)].$$

Figure 2 shows the transition system $TS(\overline{E})$ and the underlying DTMC $DTMC(\overline{E})$. For simplicity, the states are labelled by expressions belonging to the corresponding equivalence classes, and singleton multisets of activities are written without braces.

$DR(\overline{E})$ consists of the equivalence classes

$$s_1 = [\overline{[E_1 * E_2 * E_3]}]_{\approx}$$
$$s_2 = [[E_1 * \overline{E_2} * E_3]]_{\approx}$$
$$s_3 = [\underline{[E_1 * E_2 * E_3]}]_{\approx}.$$

To see how the transition probabilities are calculated, consider, for instance,

$$PF(\{(\{a\},\rho)_1\},s_1) = PF(\{(\{a\},\rho)_2\},s_1) = \rho(1-\rho)$$

and

$$PF(\emptyset,s_1) = (1-\rho)^2.$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

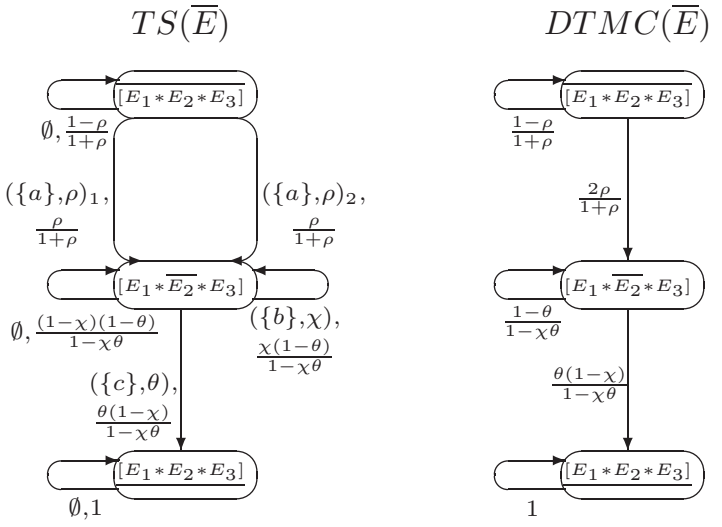



Fig. 2. The transition system and the underlying DTMC of $\overline{E}$ for

$$E = [((\{a\},\rho)_1 [] (\{a\},\rho)_2) * (\{b\},\chi) * (\{c\},\theta)].$$

Hence,

$$\sum_{\Delta \in Exec(s_1)} PF(\Delta, s_1) = 2\rho(1-\rho) + (1-\rho)^2 = 1 - \rho^2.$$

Thus,

$$\begin{aligned} PT(\{(\{a\},\rho)_1\}, s_1) &= PT(\{(\{a\},\rho)_2\}, s_1) \\ &= \frac{\rho(1-\rho)}{1-\rho^2} \\ &= \frac{\rho(1-\rho)}{(1-\rho)(1+\rho)} \\ &= \frac{\rho}{1+\rho} \end{aligned}$$

and

$$\begin{aligned} PT(\emptyset, s_1) &= \frac{(1-\rho)^2}{1-\rho^2} \\ &= \frac{(1-\rho)^2}{(1-\rho)(1+\rho)} \\ &= \frac{1-\rho}{1+\rho}. \end{aligned}$$

Other probabilities are calculated similarly. The average sojourn time vector of $\overline{E}$ is

$$SJ = \left( \frac{1+\rho}{2\rho}, \frac{1-\chi\theta}{\theta(1-\chi)}, \infty \right).$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

The sojourn time variance vector of $\overline{E}$ is

$$VAR = \left( \frac{1-\rho^2}{4\rho^2}, \frac{(1-\theta)(1-\chi\theta)}{\theta^2(1-\chi)^2}, \infty \right).$$

## 4. Denotational semantics

In this section, we define the denotational semantics in terms of a subclass of LDTSPNs, which are called discrete time stochastic Petri boxes (dts-boxes) – we present an example at the end of the section.

### 4.1. *Labelled DTSPNs*

We introduce a class of labelled discrete time stochastic PNs (LDTSPNs), which are a subclass of DTSPNs (Molloy 1985) (we do not allow the transition probabilities to be equal to 1) with transition labelling. LDTSPNs are similar to the labelled weighted DTSPNs (LWDTSPNs) of Buchholz and Tarasyuk (2001), but in LWDTSPNs, all transitions have weights, the transition probabilities may be equal to 1 and only maximal fireable subsets of the enabled transitions are fired.

**Definition 4.1.** A *labelled DTSPN (LDTSPN)* is a tuple

$$N = (P_N, T_N, W_N, \Omega_N, L_N, M_N),$$

where:

— $P_N$ and $T_N$ are finite sets of *places* and *transitions*, respectively, such that

$$P_N \cup T_N \neq \emptyset$$
$$P_N \cap T_N = \emptyset.$$

— $W_N : (P_N \times T_N) \cup (T_N \times P_N) \to \mathbb{N}$ is a function providing the *weights of arcs* between places and transitions.

— $\Omega_N : T_N \to (0;1)$ is the *transition probability* function associating transitions with probabilities.

— $L_N : T_N \to \mathcal{L}$ is the *transition labelling* function assigning multiactions to transitions.

— $M_N \in \mathbb{N}_f^{P_N}$ is the *initial marking*.

LDTSPNs can be represented graphically in the same way as standard labelled PNs, but with probabilities written near the corresponding transitions – the probabilities are not shown if they are considered to be of no importance. Also, the weights of arcs are shown near the arcs, and, when needed, the names of places and transitions are shown near them.

Let $N$ be an LDTSPN and $t \in T_N$, $U \in \mathbb{N}_f^{T_N}$. The *precondition* ${}^\bullet t$ and the *postcondition* $t^\bullet$ of $t$ are the multisets of places defined by

$$({}^\bullet t)(p) = W_N(p,t)$$
$$(t^\bullet)(p) = W_N(t,p).$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

The *precondition* ${}^\bullet U$ and the *postcondition* $U^\bullet$ of $U$ are the multisets of places defined by

$$ {}^\bullet U = \sum_{t \in U} {}^\bullet t $$

$$ U^\bullet = \sum_{t \in U} t^\bullet. $$

Note that for $U = \emptyset$, we have

$$ {}^\bullet \emptyset = \emptyset = \emptyset^\bullet. $$

A transition $t \in T_N$ is *enabled* in a marking $M \in I\!N_f^{P_N}$ of LDTSPN $N$ if ${}^\bullet t \subseteq M$. Let $Ena(M)$ be the set of *all the transitions (such that each of them is) enabled in a marking* $M$. A set of transitions $U \subseteq Ena(M)$ is *enabled* in a marking $M$ if ${}^\bullet U \subseteq M$. The firings of transitions are atomic operations, and transitions may fire concurrently in steps. We assume that all transitions participating in a step should differ, hence, only the sets (not multisets) of transitions may fire. Hence, we do not allow self-concurrency, that is, the firing of transitions concurrently with themselves. This restriction is introduced because we would like to avoid some technical difficulties when calculating probabilities for multisets of transitions, as we shall see after the following formal definitions. Moreover, we do not need to consider self-concurrency since the denotational semantics of expressions will be defined using dts-boxes, which are safe LDTSPNs (hence, no self-concurrency is possible).

Let $M$ be a marking of an LDTSPN $N$. A transition $t \in Ena(M)$ fires with probability $\Omega_N(t)$ when no other transitions conflicting with it are enabled.

Let $U \subseteq Ena(M)$, $U \neq \emptyset$ and ${}^\bullet U \subseteq M$. The *probability that the set of transitions* $U$ *is ready for firing in* $M$ is

$$ PF(U, M) = \prod_{t \in U} \Omega_N(t) \cdot \prod_{u \in Ena(M) \setminus U} (1 - \Omega_N(u)). $$

When $U = \emptyset$, we define

$$ PF(\emptyset, M) = \begin{cases} \prod_{u \in Ena(M)} (1 - \Omega_N(u)) & \text{if } Ena(M) \neq \emptyset \\ 1 & \text{otherwise.} \end{cases} $$

Let $U \subseteq Ena(M)$ and ${}^\bullet U \subseteq M$. Besides $U$, some other sets of transitions may be ready for firing in $M$, so a kind of conditioning or normalisation is needed to calculate the firing probability. The concurrent firing of the transitions from $U$ changes the marking $M$ to

$$ \widetilde{M} = M - {}^\bullet U + U^\bullet, $$

denoted by

$$ M \stackrel{U}{\rightarrow}_{\mathcal{P}} \widetilde{M}, $$

where $\mathcal{P} = PT(U, M)$ is the *probability that the set of transitions* $U$ *fires in* $M$ defined by

$$ PT(U, M) = \frac{PF(U, M)}{\sum_{\{V \mid {}^\bullet V \subseteq M\}} PF(V, M)}. $$

Note that when $U = \emptyset$, we have $M = \widetilde{M}$.

Let $Ena(M) = \{t_1, \ldots, t_n\}$ be a mutually exclusive set of transitions (that is, a firing of any transition from the set results in a marking in which no other transition from

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

the set is enabled) and $\rho_i = \Omega_N(t_i)$ $(1 \leqslant i \leqslant n)$. In this way, $PT(\{t_i\}, M)$ resembles the probabilistic function $P[E_i]$ in Molloy (1985), which defines the probability of the event $E_i$, *viz.* that transition $t_i$ in a mutually exclusive set of transitions $\{t_1, \ldots, t_n\}$ will fire in the marking $M$. We have

$$P[E_i] = \frac{\frac{\rho_i}{1-\rho_i}}{1 + \sum_{j=1}^{n} \frac{\rho_j}{1-\rho_j}} = \frac{\frac{\rho_i(1-\rho_1)\cdots(1-\rho_n)}{1-\rho_i}}{(1-\rho_1)\cdots(1-\rho_n) + \sum_{j=1}^{n} \frac{\rho_j(1-\rho_1)\cdots(1-\rho_n)}{1-\rho_j}},$$

where

$$\frac{\rho_i(1-\rho_1)\cdots(1-\rho_n)}{1-\rho_i}$$

corresponds to $PF(\{t_i\}, M)$ in our setting. Furthermore, $PT(\emptyset, M)$ resembles the probabilistic function $P[E_0]$, which defines the probability of the event $E_0$, that no transitions from the mutually exclusive set of transitions $\{t_1, \ldots, t_n\}$ will fire in the marking $M$. We have

$$P[E_0] = \frac{1}{1 + \sum_{j=1}^{n} \frac{\rho_j}{1-\rho_j}} = \frac{(1-\rho_1)\cdots(1-\rho_n)}{(1-\rho_1)\cdots(1-\rho_n) + \sum_{j=1}^{n} \frac{\rho_j(1-\rho_1)\cdots(1-\rho_n)}{1-\rho_j}},$$

where $(1-\rho_1)\cdots(1-\rho_n)$ corresponds to $PF(\emptyset, M)$ in our setting. If $Ena(M)$ is not a mutually exclusive set of transitions, our way of defining $PT(U, M)$ for $U \subseteq Ena(M)$, $U \neq \emptyset$, also extends the approach of Molloy (1985) and Molloy (1981). The advantage of our two-stage definition of $PT(U, M)$ is that it has a closed form and we do not need to consider which sets of transitions are exclusive – instead, we just consider the probability that $U$ fires in $M$ under the condition that only particular subsets of $Ena(M)$ can fire in $M$.

Note that for all markings of an LDTSPN $N$, the sum of the outgoing probabilities is equal to 1, that is, for all $M \in \mathbb{N}_f^{P_N}$, we have

$$PT(\emptyset, M) + \sum_{\{U \mid {}^\bullet U \subseteq M\}} PT(U, M) = 1.$$

This follows from the definition of $PT(U, M)$ and guarantees that it defines a probability distribution.

We write:

— $M \xrightarrow{U} \widetilde{M}$ if there exists $\mathcal{P}$ such that $M \xrightarrow{U}_{\mathcal{P}} \widetilde{M}$; and
— $M \to \widetilde{M}$ if there exists $U$ such that $M \xrightarrow{U} \widetilde{M}$.

**Definition 4.2.** Let $N$ be an LDTSPN.

— The *reachability set* $RS(N)$ of $N$ is the minimal set of markings such that:
  – $M_N \in RS(N)$.
  – If $M \in RS(N)$ and $M \to \widetilde{M}$, then $\widetilde{M} \in RS(N)$.
— The *reachability graph* $RG(N)$ of $N$ is a directed labelled graph with the set of nodes $RS(N)$ and an arc labelled with $(U, \mathcal{P})$ between nodes $M$ and $\widetilde{M}$ if $M \xrightarrow{U}_{\mathcal{P}} \widetilde{M}$.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

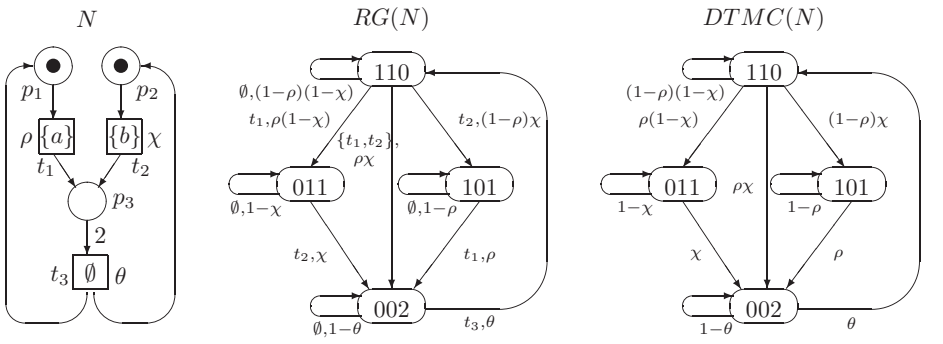



Fig. 3. LDTSPN, its reachability graph and the underlying DTMC.

— The *underlying discrete time Markov chain (DTMC)* $DTMC(N)$ of $N$ has state space $RS(N)$, initial state $M_N$ and transitions $M \to_{\mathcal{P}} \widetilde{M}$ if $M \to \widetilde{M}$, where $\mathcal{P} = PM(M, \widetilde{M})$ is the *probability of moving from $M$ to $\widetilde{M}$ by firing any set of transitions* defined by

$$PM(M, \widetilde{M}) = \sum_{\{U | M \xrightarrow{U} \widetilde{M}\}} PT(U, M).$$

Let $N$ be an LDTSPN and $M \in RS(N)$. The *average sojourn time in the marking $M$* is given by

$$SJ(M) = \frac{1}{1 - PM(M, M)}.$$

The *average sojourn time vector* of $N$, denoted by $SJ$, has elements $SJ(M)$, $M \in RS(N)$. The *sojourn time variance in the marking $M$* is

$$VAR(M) = \frac{PM(M, M)}{(1 - PM(M, M))^2}.$$

The *sojourn time variance vector* of $N$, denoted by $VAR$, has elements $VAR(M)$, $M \in RS(N)$.

**Example 4.3.** Figure 3 shows an LDTSPN $N$ with two visible transitions $t_1$ (labelled by $\{a\}$) and $t_2$ (labelled by $\{b\}$), and one invisible transition $t_3$ (labelled by $\emptyset$). The transition probabilities of $N$ are denoted by

$$\rho = \Omega_N(t_1)$$
$$\chi = \Omega_N(t_2)$$
$$\theta = \Omega_N(t_3).$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

The figure also shows both the reachability graph $RG(N)$ and the underlying DTMC $DTMC(N)$. The reachability set $RS(N)$ consists of the markings

$$M_1 = (1, 1, 0)$$
$$M_2 = (0, 1, 1)$$
$$M_3 = (1, 0, 1)$$
$$M_4 = (0, 0, 2).$$

The average sojourn time vector of $N$ is

$$SJ = \left(\frac{1}{\rho + \chi - \rho\chi}, \frac{1}{\chi}, \frac{1}{\rho}, \frac{1}{\theta}\right).$$

The sojourn time variance vector of $N$ is

$$VAR = \left(\frac{1 - \rho - \chi + \rho\chi}{(\rho + \chi - \rho\chi)^2}, \frac{1 - \chi}{\chi^2}, \frac{1 - \rho}{\rho^2}, \frac{1 - \theta}{\theta^2}\right).$$

### 4.2. *Algebra of dts-boxes*

We will now define the discrete time stochastic Petri boxes and associated algebraic operations, which define a net representation of dtsPBC expressions.

**Definition 4.4.** A *discrete time stochastic Petri box (dts-box)* is a tuple

$$N = (P_N, T_N, W_N, \Lambda_N),$$

where:

— $P_N$ and $T_N$ are finite sets of *places* and *transitions*, respectively, such that

$$P_N \cup T_N \neq \emptyset$$
$$P_N \cap T_N = \emptyset.$$

— $W_N : (P_N \times T_N) \cup (T_N \times P_N) \to \mathbb{N}$ is a function providing the *weights of arcs* between places and transitions.

— $\Lambda_N$ is the *place and transition labelling* function such that:

- $\Lambda_N|_{P_N} : P_N \to \{\mathsf{e}, \mathsf{i}, \mathsf{x}\}$ (this specifies *entry, internal* and *exit* places, respectively).
- $\Lambda_N|_{T_N} : T_N \to \{\varrho \mid \varrho \subseteq \mathbb{N}_f^{\mathcal{SL}} \times \mathcal{SL}\}$ (this associates transitions with the *relabelling relations* on activities).

We require that for all $t \in T_N$, we have ${}^\bullet t \neq \emptyset \neq t^\bullet$. In addition, for the set of *entry* places of $N$ defined by

$${}^\circ N = \{p \in P_N \mid \Lambda_N(p) = \mathsf{e}\}$$

and the set of *exit* places of $N$ defined by

$$N^\circ = \{p \in P_N \mid \Lambda_N(p) = \mathsf{x}\},$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

we have

$$^\circ N \neq \emptyset \neq N^\circ$$
$$^\bullet(^\circ N) = \emptyset = (N^\circ)^\bullet.$$

A dts-box is *plain* if for all $t \in T_N$, we have $\Lambda_N(t) \in \mathcal{SL}$, that is, $\Lambda_N(t)$ is a constant relabelling, which will be defined later. In the case of a constant relabelling, we will use a shorthand notation (using an activity) for $\Lambda_N(t)$. A *marked plain dts-box* is a pair $(N, M_N)$, where $N$ is a plain dts-box and $M_N \in \mathbb{N}_f^{P_N}$ is its marking. We use the notation $\overline{N} = (N, ^\circ N)$ and $\underline{N} = (N, N^\circ)$. A marked plain dts-box $(P_N, T_N, W_N, \Lambda_N, M_N)$ could be interpreted as the LDTSPN $(P_N, T_N, W_N, \Omega_N, L_N, M_N)$, where functions $\Omega_N$ and $L_N$ are defined as follows: for all $t \in T_N$, we have

$$\Omega_N(t) = \Omega(\Lambda_N(t))$$
$$L_N(t) = \mathcal{L}(\Lambda_N(t)).$$

The behaviour of the marked dts-boxes follows from the LDTSPNs firing rule. A plain dts-box $N$ is:

— *n-bounded* ($n \in \mathbb{N}$) if $\overline{N}$ is, that is, if for all $M \in RS(\overline{N})$ and all $p \in P_N$, we have $M(p) \leqslant n$;
— *safe* if it is 1-bounded;
— *clean* if for all $M \in RS(\overline{N})$, we have $^\circ N \subseteq M$ implies $M = {}^\circ N$ and $N^\circ \subseteq M$ implies $M = N^\circ$, that is, if there are tokens in any of its entry (exit) places, then no other places have tokens.

The structure of the plain dts-box corresponding to a static expression is constructed as in PBC (Best and Koutny 1995; Best *et al.* 2001), that is, we use a simultaneous refinement and relabelling meta-operator (net refinement) in addition to the *operator dts-boxes* corresponding to the algebraic operations of dtsPBC and featuring transformational transition relabellings. Thus, the resulting plain dts-boxes are safe and clean. In the definition of the denotational semantics, we shall apply the standard constructions used for PBC. Let $\Theta$ denote an *operator box* and $u$ denote a *transition name* from the PBC setting.

The relabelling relations $\varrho \subseteq \mathbb{N}_f^{\mathcal{SL}} \times \mathcal{SL}$ are defined as follows:

— $\varrho_{id} = \{(\{(\alpha, \rho)\}, (\alpha, \rho)) \mid (\alpha, \rho) \in \mathcal{SL}\}$ is the *identity relabelling* keeping the interface.
— $\varrho_{(\alpha,\rho)} = \{(\emptyset, (\alpha, \rho))\}$ is the *constant relabelling* identified with $(\alpha, \rho) \in \mathcal{SL}$ itself.
— $\varrho_{[f]} = \{(\{(\alpha, \rho)\}, (f(\alpha), \rho)) \mid (\alpha, \rho) \in \mathcal{SL}\}$.
— $\varrho_{\mathsf{rs}\ a} = \{(\{(\alpha, \rho)\}, (\alpha, \rho)) \mid (\alpha, \rho) \in \mathcal{SL},\ a, \hat{a} \notin \alpha\}$.
— $\varrho_{\mathsf{sy}\ a}$ is the least relabelling relation containing $\varrho_{id}$ such that if

$$(\Gamma, (\alpha, \rho)), (\Delta, (\beta, \chi)) \in \varrho_{\mathsf{sy}\ a}$$

and $a \in \alpha,\ \hat{a} \in \beta$, then

$$(\Gamma + \Delta, (\alpha \oplus_a \beta, \rho \cdot \chi)) \in \varrho_{\mathsf{sy}\ a}.$$

The plain and operator dts-boxes are shown in Figure 4, where the symbol i is often omitted.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

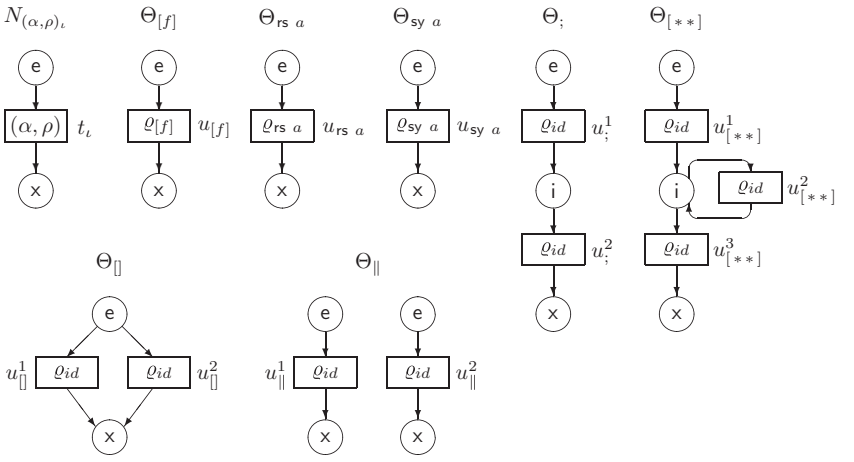



Fig. 4. The plain and operator dts-boxes.

To construct a semantic function associating a plain dts-box with every static expression of dtsPBC, we need to define the *enumeration* function $Enu : T_N \to Num$. This associates numberings with transitions of the plain dts-box $N$ according to those of the activities. In the case of synchronisation, the function associates the concatenation of the parenthesised numberings of the synchronised transitions with the resulting new transition.

To define the enumeration function $Enu$ for every operator of dtsPBC, we let

$$Box_{dts}(E) = (P_E, T_E, W_E, \Lambda_E)$$

be the plain dts-box corresponding to a static expression $E$, and $Enu_E : T_E \to Num$ be the enumeration function for $Box_{dts}(E)$, with analogous notation for static expressions $F$ and $K$. Then:

— $Box_{dts}(E \circ F) = \Theta_\circ(Box_{dts}(E), Box_{dts}(F)),\ \circ \in \{;, [], \|\}$.
Since we do not introduce any new transitions, we preserve the initial numbering:

$$Enu(t) = \begin{cases} Enu_E(t) & \text{if } t \in T_E \\ Enu_F(t) & \text{if } t \in T_F. \end{cases}$$

— $Box_{dts}(E[f]) = \Theta_{[f]}(Box_{dts}(E))$.
Since we only replace the labels of some multiactions by a bijection, we preserve the initial numbering:

$$Enu(t) = Enu_E(t),\ t \in T_E.$$

— $Box_{dts}(E\ \mathsf{rs}\ a) = \Theta_{\mathsf{rs}\ a}(Box_{dts}(E))$.
Since we remove all the transitions labelled with multiactions containing $a$ or $\hat{a}$, this does not change the numbering of the remaining transitions:

$$Enu(t) = Enu_E(t),\ t \in T_E,\ a, \hat{a} \notin \mathcal{L}(\Lambda_E(t)).$$

— $Box_{dts}(E\ \mathsf{sy}\ a) = \Theta_{\mathsf{sy}\ a}(Box_{dts}(E))$.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Note that for all $v, w \in T_E$ such that

$$\Lambda_E(v) = (\alpha, \rho)$$
$$\Lambda_E(w) = (\beta, \chi)$$
$$a \in \alpha$$
$$\hat{a} \in \beta,$$

the new transition $t$ resulting from the synchronisation of $v$ and $w$ has the label

$$\Lambda(t) = (\alpha \oplus_a \beta, \rho \cdot \chi)$$

and the numbering

$$Enu(t) = (Enu_E(v))(Enu_E(w)).$$

The enumeration function is

$$Enu(t) = \begin{cases} Enu_E(t) & \text{if } t \in T_E \\ (Enu_E(v))(Enu_E(w)) & \text{if } t \text{ results from the synchronisation of } v \text{ and } w. \end{cases}$$

When we synchronise the same set of transitions in different orders, we get several resulting transitions with the same label and probability, but with different numberings having the same content. So we shall only consider a single transition from the resulting ones in the plain dts-box to avoid introducing redundancy.

— $Box_{dts}([E * F * K]) = \Theta_{[**]}(Box_{dts}(E), Box_{dts}(F), Box_{dts}(K))$.
Since we do not introduce any new transitions, we preserve the initial numbering:

$$Enu(t) = \begin{cases} Enu_E(t) & \text{if } t \in T_E \\ Enu_F(t) & \text{if } t \in T_F \\ Enu_K(t) & \text{if } t \in T_K. \end{cases}$$

**Definition 4.5.** Let $(\alpha, \rho) \in \mathcal{SL}$, $a \in Act$ and $E, F, K \in RegStatExpr$. The *denotational semantics* of dtsPBC is a mapping $Box_{dts}$ from $RegStatExpr$ into the area of plain dts-boxes defined as follows:

$$Box_{dts}((\alpha, \rho)_\iota) = N_{(\alpha,\rho)_\iota}$$
$$Box_{dts}(E \circ F) = \Theta_\circ(Box_{dts}(E), Box_{dts}(F)), \ \circ \in \{;, [], \|\}$$
$$Box_{dts}(E[f]) = \Theta_{[f]}(Box_{dts}(E))$$
$$Box_{dts}(E \circ a) = \Theta_{\circ a}(Box_{dts}(E)), \ \circ \in \{\mathsf{rs}, \mathsf{sy}\}$$
$$Box_{dts}([E * F * K]) = \Theta_{[**]}(Box_{dts}(E), Box_{dts}(F), Box_{dts}(K)).$$

For $E \in RegStatExpr$, let

$$Box_{dts}(\overline{E}) = \overline{Box_{dts}(E)}$$
$$Box_{dts}(\underline{E}) = \underline{Box_{dts}(E)}.$$

This definition is compositional in the sense that, for any dynamic expression, we may decompose it into some inner dynamic and static expressions, for which we may apply

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

the definition, thus obtaining the corresponding plain dts-boxes, which can be joined according to the term structure (by the definition of $Box_{dts}$), the resulting plain box being marked in the places that were marked in the argument nets.

Let $\simeq$ denote the isomorphism between transition systems, or between DTMCs and reachability graphs, that binds their initial states. The names of the transitions of the dts-box corresponding to a static expression could be identified with the enumerated activities of the latter. For a dts-box $N$, we denote its *reachability graph* by $RG(N)$ and its *underlying DTMC* by $DTMC(N)$.

**Theorem 4.6 (Tarasyuk 2006).** For any static expression $E$,

$$TS(\overline{E}) \simeq RG(Box_{dts}(\overline{E})).$$

*Proof.* As with the qualitative behaviour, we have the same isomorphism as in PBC. The quantitative behaviour is the same since the activities of an expression have probability parts coinciding with the probabilities of the transitions belonging to the corresponding dts-box and, both in stochastic processes specified by expressions and dts-boxes, conflicts are resolved using analogous probability functions. □

**Proposition 4.7 (Tarasyuk 2006).** For any static expression $E$,

$$DTMC(\overline{E}) \simeq DTMC(Box_{dts}(\overline{E})).$$

*Proof.* The statement follows by Theorem 4.6 and the definitions of the underlying DTMCs for dynamic expressions and LDTSPNs since the transition probabilities of the associated DTMCs are the sums of those belonging to the transition systems or reachability graphs. □

**Example 4.8.** Let $E$ be as in Example 3.10. Figure 5 shows the marked dts-box $N = Box_{dts}(\overline{E})$, its reachability graph $RG(N)$ and the underlying DTMC $DTMC(N)$. It is easy to see that $TS(\overline{E})$ and $RG(N)$ are isomorphic, as well as $DTMC(\overline{E})$ and $DTMC(N)$.

The following example shows that without the syntactic restriction on the regularity of expressions, the corresponding marked dts-boxes may not be safe.

**Example 4.9.** Let

$$E = [((\{a\}, \tfrac{1}{2}) * ((\{b\}, \tfrac{1}{2}) \| (\{c\}, \tfrac{1}{2})) * (\{d\}, \tfrac{1}{2})].$$

Figure 6 shows the marked dts-box $N = Box_{dts}(\overline{E})$ and its reachability graph $RG(N)$. In the marking $(0, 1, 1, 2, 0, 0)$, there are 2 tokens in the place $p_4$. Symmetrically, in the marking $(0, 1, 1, 0, 2, 0)$, there are 2 tokens in the place $p_5$. Thus, allowing concurrency in the second argument of iteration in the expression $\overline{E}$ can lead to non-safeness of the corresponding marked dts-box $N$, though it is 2-bounded in the worst case (Best *et al.* 2001). The origin of the problem is that $N$ has as a self-loop with two subnets that can function independently. This is why we have decided to consider regular expressions only since the alternative, which is a safe version of the iteration operator with six arguments in the

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

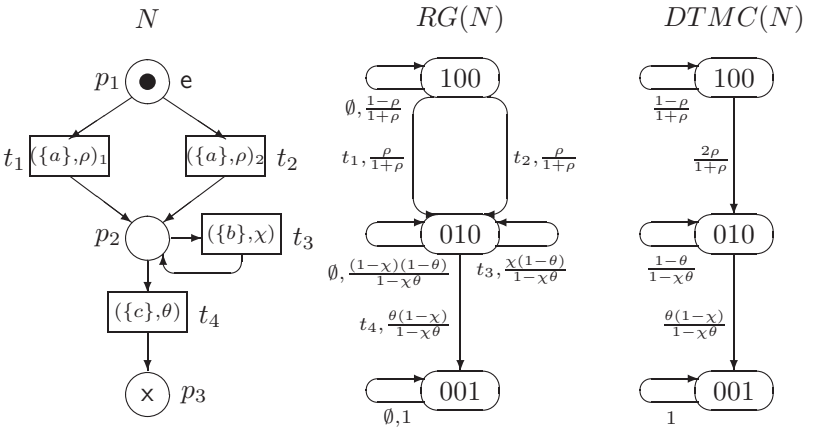



Fig. 5. The marked dts-box $N = Box_{dts}(\overline{E})$ for

$$E = [((\{a\},\rho)_1 [] (\{a\},\rho)_2) * (\{b\},\chi) * (\{c\},\theta)],$$

its reachability graph and the underlying DTMC.

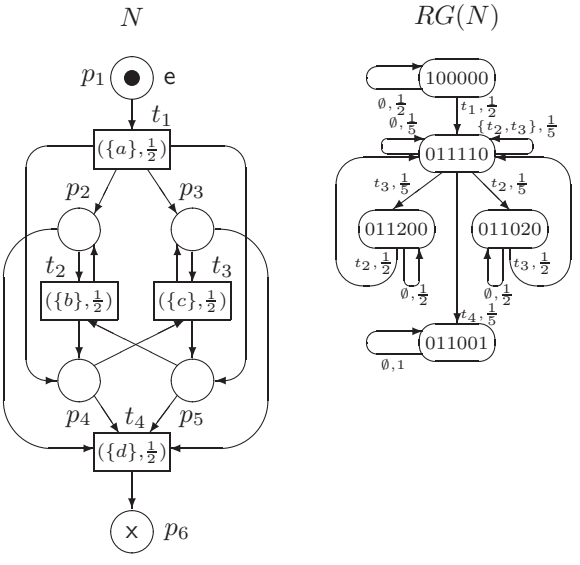



Fig. 6. The marked dts-box $N = Box_{dts}(\overline{E})$ for

$$E = [(\{a\}, \tfrac{1}{2}) * ((\{b\}, \tfrac{1}{2}) \| (\{c\}, \tfrac{1}{2})) * (\{d\}, \tfrac{1}{2})]$$

and its reachability graph.

corresponding dts-box, like the one in Best *et al.* (2001), is rather cumbersome and has an over-intricate Petri net interpretation – our motivation was to keep the algebraic and Petri net specifications as simple as possible.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

## 5. Stochastic equivalences

Consider the expressions

$$E = (\{a\}, \tfrac{1}{2})$$
$$E' = (\{a\}, \tfrac{1}{3})_1 [] (\{a\}, \tfrac{1}{3})_2,$$

for which $\overline{E} \neq_{ts} \overline{E'}$ since $TS(\overline{E})$ has only one transition from the initial to the final state (with probability $\frac{1}{2}$) while $TS(\overline{E'})$ has two such transitions (with probabilities $\frac{1}{4}$). On the other hand, all the mentioned transitions are labelled by activities with the same multiaction part $\{a\}$. Moreover, the overall probabilities of the $TS(\overline{E})$ and $TS(\overline{E'})$ transitions coincide: $\frac{1}{2} = \frac{1}{4} + \frac{1}{4}$. Furthermore, $TS(\overline{E})$ (as well as $TS(\overline{E'})$) has one empty loop transition from the initial state to itself with probability $\frac{1}{2}$, and one empty loop transition from the final state to itself with probability 1. The empty loop transitions are labelled by the empty multiset of activities. To calculate the transition probabilities of $TS(\overline{E'})$, take $\rho = \chi = \frac{1}{3}$ in Example 3.6. Unlike $=_{ts}$, most of the probabilistic and stochastic equivalences proposed in the literature do not differentiate between processes like those specified by $E$ and $E'$.

Since the semantic equivalence $=_{ts}$ is too discriminating in many cases, we need some weaker equivalence notions. These equivalences should possess the following necessary properties:

(1) Any two equivalent processes must have the same sequences of multisets of multiactions, which are the multiaction parts of the activities executed in steps starting from the initial states of the processes.
(2) For every such sequence, its execution probabilities within both processes must coincide.
(3) The equivalence should preserve the branching structure of computations, that is, the points at which an external observer can choose between several extensions of a particular computation should be taken into account.

In this section, we define two such notions: step stochastic bisimulation equivalence and stochastic isomorphism.

### 5.1. *Step stochastic bisimulation equivalence*

Bisimulation equivalences respect the particular choice points in the behaviour of a system. To define stochastic bisimulation equivalences, we have to consider a bisimulation as an *equivalence* relation that partitions the states of the *union* of the transition systems $TS(G)$ and $TS(G')$ of the two dynamic expressions $G$ and $G'$ to be compared. For $G$ and $G'$ to be bisimulation equivalent, the initial states of their transition systems, $[G]_{\approx}$ and $[G']_{\approx}$, must be related by a bisimulation having the following transfer property: two states are related if in each of them the same multisets of multiactions can occur, and the resulting states *belong to the same equivalence class*. In addition, the sums of probabilities for all such occurrences should be the same for both states.

In this way, we follow the approaches taken in Jou and Smolka (1990), Larsen and Skou (1991), Hermanns and Rettelbach (1994), Hillston (1996), Bernardo and

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Gorrieri (1998) and Bernardo (2007), but implement step semantics rather than the interleaving semantics considered in those papers. Recall also that we use generative probabilistic transition systems, as in Jou and Smolka (1990), in contrast to the reactive model treated in Larsen and Skou (1991), and we take transition probabilities rather than the transition rates used in Hermanns and Rettelbach (1994), Hillston (1996), Bernardo and Gorrieri (1998) and Bernardo (2007). Thus, the step stochastic bisimulation equivalence we will define is (in the probabilistic sense) only comparable with the interleaving probabilistic bisimulation equivalence of Jou and Smolka (1990), and our equivalence is obviously stronger.

In the definition below, we consider $\mathcal{L}(\Gamma) \in \mathbb{N}_f^{\mathcal{L}}$ for $\Gamma \in \mathbb{N}_f^{\mathcal{SL}}$, that is, the (possibly empty) multisets of multiactions. The multiactions can be empty, in which case $\mathcal{L}(\Gamma)$ contains the elements $\emptyset$, and is not itself empty.

Let $G$ be a dynamic expression and $\mathcal{H} \subseteq DR(G)$. Then, for any $s \in DR(G)$ and $A \in \mathbb{N}_f^{\mathcal{L}}$, we write $s \xrightarrow{A}_{\mathcal{P}} \mathcal{H}$, where $\mathcal{P} = PM_A(s, \mathcal{H})$ is the *overall probability of moving from s into the set of states* $\mathcal{H}$ *via steps with the multiaction part* $A$ defined by

$$PM_A(s, \mathcal{H}) = \sum_{\{\Gamma \mid \exists \tilde{s} \in \mathcal{H}\ s \xrightarrow{\Gamma} \tilde{s},\ \mathcal{L}(\Gamma) = A\}} PT(\Gamma, s).$$

We write $s \xrightarrow{A} \mathcal{H}$ if there exists $\mathcal{P}$ such that $s \xrightarrow{A}_{\mathcal{P}} \mathcal{H}$, and we write $s \rightarrow_{\mathcal{P}} \mathcal{H}$ if there exists $A$ such that $s \xrightarrow{A} \mathcal{H}$, where $\mathcal{P} = PM(s, \mathcal{H})$ is the *overall probability of moving from s into the set of states* $\mathcal{H}$ *via any steps* defined by

$$PM(s, \mathcal{H}) = \sum_{\{\Gamma \mid \exists \tilde{s} \in \mathcal{H}\ s \xrightarrow{\Gamma} \tilde{s}\}} PT(\Gamma, s).$$

To introduce a stochastic bisimulation between dynamic expressions $G$ and $G'$, we should consider the 'composite' set of states $DR(G) \cup DR(G')$ since we have to identify the probabilities of coming from any two equivalent states into the same 'composite' equivalence class (with respect to the stochastic bisimulation). Note that for $G \neq G'$, transitions starting from the states of $DR(G)$ (or $DR(G')$) always lead to those from the same set since $DR(G) \cap DR(G') = \emptyset$, and this allows us to 'mix' the sets of states in the definition of stochastic bisimulation.

**Definition 5.1.** Let $G$ and $G'$ be dynamic expressions. An *equivalence* relation

$$\mathcal{R} \subseteq (DR(G) \cup DR(G'))^2$$

is a *step stochastic bisimulation* between $G$ and $G'$, denoted by $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$, if:

(1) $([G]_{\approx}, [G']_{\approx}) \in \mathcal{R}$.

(2) $(s_1, s_2) \in \mathcal{R} \ \Rightarrow\ \forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}\ \forall A \in \mathbb{N}_f^{\mathcal{L}}$

$$s_1 \xrightarrow{A}_{\mathcal{P}} \mathcal{H} \ \Leftrightarrow\ s_2 \xrightarrow{A}_{\mathcal{P}} \mathcal{H}.$$

Dynamic expressions $G$ and $G'$ are *step stochastic bisimulation equivalent*, denoted by $G \underline{\leftrightarrow}_{ss} G'$, if there exists $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Let

$$\mathcal{R}_{ss}(G, G') = \bigcup \{\mathcal{R} \mid \mathcal{R} : G \underline{\leftrightarrow}_{ss} G'\}$$

be the *union of all step stochastic bisimulations* between $G$ and $G'$. The following proposition proves that $\mathcal{R}_{ss}(G, G')$ is also an *equivalence* and $\mathcal{R}_{ss}(G, G') : G \underline{\leftrightarrow}_{ss} G'$.

**Proposition 5.2.** Let $G$ and $G'$ be dynamic expressions and $G \underline{\leftrightarrow}_{ss} G'$. Then $\mathcal{R}_{ss}(G, G')$ is the largest step stochastic bisimulation between $G$ and $G'$.

*Proof.* See Appendix A.1. □

The algorithm in Paige and Tarjan (1987) for determining bisimulation of transition systems can be adapted for our framework. This algorithm has time complexity $O(m \log n)$, where $n$ is the number of states and $m$ is the number of transitions.

### 5.2. *Stochastic isomorphism*

Stochastic isomorphism is weaker than $=_{ts}$. The main idea is to collect the probabilities of all transitions between the same pair of states such that the transition labels have the same multiaction parts.

For a dynamic expression $G$, let $s, \tilde{s} \in DR(G)$ and $s \xrightarrow{A}_{\mathcal{P}} \{\tilde{s}\}$. Then we write $s \xrightarrow{A}_{\mathcal{P}} \tilde{s}$.

**Definition 5.3.** Let $G$ and $G'$ be dynamic expressions. A mapping $\beta : DR(G) \to DR(G')$ is a *stochastic isomorphism* between $G$ and $G'$, denoted by $\beta : G =_{sto} G'$, if:

(1) $\beta$ is a bijection such that $\beta([G]_{\approx}) = [G']_{\approx}$.
(2) $\forall s, \tilde{s} \in DR(G)\ \forall A \in \mathbb{N}_f^{\mathcal{L}}\ s \xrightarrow{A}_{\mathcal{P}} \tilde{s} \Leftrightarrow \beta(s) \xrightarrow{A}_{\mathcal{P}} \beta(\tilde{s})$.

Dynamic expressions $G$ and $G'$ are *stochastically isomorphic*, denoted by $G =_{sto} G'$, if there exists $\beta : G =_{sto} G'$.

### 5.3. *Relationship between the stochastic equivalences*

We will now compare the discrimination power of the stochastic equivalences.

**Theorem 5.4.** For dynamic expressions $G$ and $G'$ the following *strict* implications hold:

$$G \approx G' \Rightarrow G =_{ts} G' \Rightarrow G =_{sto} G' \Rightarrow G \underline{\leftrightarrow}_{ss} G'.$$

*Proof.* We first check the validity of the forward implications:

— To prove the implication $=_{sto} \Rightarrow \underline{\leftrightarrow}_{ss}$, let $\beta : G =_{sto} G'$. It is then easy to see that $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$, where $\mathcal{R} = \{(s, \beta(s)) \mid s \in DR(G)\}$.
— The implication $=_{ts} \Rightarrow =_{sto}$ is valid since stochastic isomorphism is an isomorphism of transition systems up to the merging of transitions with labels having identical multiaction parts.
— The implication $\approx \Rightarrow =_{ts}$ is valid since the transition system of a dynamic formula is defined on the basis of its structural equivalence class.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

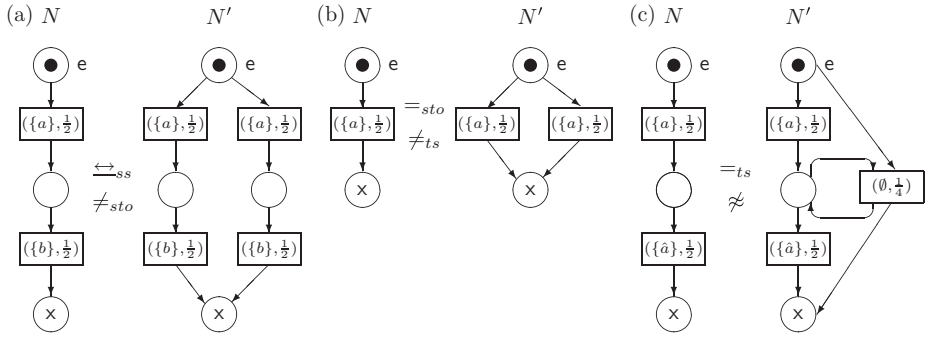



Fig. 7. Dts-boxes of the dynamic expressions from equivalence examples of Theorem 5.4.

We will now show that that the implications are strict, that is, the reverse implications do not hold, using the following counterexamples:

(a) Let

$$E = (\{a\}, \tfrac{1}{2});(\{b\}, \tfrac{1}{2})$$
$$E' = (\{a\}, \tfrac{1}{3});(\{b\}, \tfrac{1}{2})[](\{a\}, \tfrac{1}{3});(\{b\}, \tfrac{1}{2}).$$

Then $\overline{E} \underline{\leftrightarrow}_{ss} \overline{E'}$, but $\overline{E} \neq_{sto} \overline{E'}$ since $TS(\overline{E'})$ has more states than $TS(\overline{E})$.

(b) Let

$$E = (\{a\}, \tfrac{1}{2})$$
$$E' = (\{a\}, \tfrac{1}{3})_1[](\{a\}, \tfrac{1}{3})_2.$$

Then $\overline{E} =_{sto} \overline{E'}$, but $\overline{E} \neq_{ts} \overline{E'}$ since $TS(\overline{E})$ has only one transition from the initial to the final state while $TS(\overline{E'})$ has two such transitions.

(c) Let

$$E = (\{a\}, \tfrac{1}{2});(\{\hat{a}\}, \tfrac{1}{2})$$
$$E' = ((\{a\}, \tfrac{1}{2});(\{\hat{a}\}, \tfrac{1}{2}))\ \mathsf{sy}\ a.$$

Then $\overline{E} =_{ts} \overline{E'}$, but $\overline{E} \not\approx \overline{E'}$ since $\overline{E}$ and $\overline{E'}$ cannot be reached from each other by inaction rules. □

**Example 5.5.** Figure 7 shows the marked dts-boxes corresponding to the dynamic expressions for the equivalence examples of Theorem 5.4, that is, $N = Box_{dts}(\overline{E})$ and $N' = Box_{dts}(\overline{E'})$ for each of the pictures (a)–(c).

## 6. Reduction modulo equivalences

The equivalences we have proposed can be used to reduce transition systems and DTMCs of expressions (reachability graphs and DTMCs of dts-boxes). Reductions of graph-based models, such as transition systems, reachability graphs and DTMCs, produce models with fewer states (the graph nodes). The goal of the reduction is to decrease the number of states in the semantic representation of the modelled system while preserving its important

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

qualitative and quantitative properties. Thus, the reduction allows us to simplify the behaviour and performance analysis of systems.

An *autobisimulation* is a bisimulation between an expression and itself. Given a dynamic expression $G$ and a step stochastic autobisimulation on it $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G$, we let $\mathcal{K} \in DR(G)/_{\mathcal{R}}$ and $s_1, s_2 \in \mathcal{K}$. Then , for all $\widetilde{\mathcal{K}} \in DR(G)/_{\mathcal{R}}$ and all $A \in \mathbb{N}_f^{\mathcal{L}}$, we have

$$s_1 \stackrel{A}{\rightarrow}_{\mathcal{P}} \widetilde{\mathcal{K}} \text{ iff } s_2 \stackrel{A}{\rightarrow}_{\mathcal{P}} \widetilde{\mathcal{K}}.$$

This statement is valid for all $s_1, s_2 \in \mathcal{K}$, so we can rewrite it as $\mathcal{K} \stackrel{A}{\rightarrow}_{\mathcal{P}} \widetilde{\mathcal{K}}$, where

$$\mathcal{P} = PM_A(\mathcal{K}, \widetilde{\mathcal{K}}) = PM_A(s_1, \widetilde{\mathcal{K}}) = PM_A(s_2, \widetilde{\mathcal{K}}).$$

We write $\mathcal{K} \stackrel{A}{\rightarrow} \widetilde{\mathcal{K}}$ if there exists $\mathcal{P}$ such that $\mathcal{K} \stackrel{A}{\rightarrow}_{\mathcal{P}} \widetilde{\mathcal{K}}$, and $\mathcal{K} \rightarrow \widetilde{\mathcal{K}}$ if there exists $A$ such that $\mathcal{K} \stackrel{A}{\rightarrow} \widetilde{\mathcal{K}}$. Similar arguments allow us to write $\mathcal{K} \rightarrow_{\mathcal{P}} \widetilde{\mathcal{K}}$, where

$$\mathcal{P} = PM(\mathcal{K}, \widetilde{\mathcal{K}}) = PM(s_1, \widetilde{\mathcal{K}}) = PM(s_2, \widetilde{\mathcal{K}}).$$

The *average sojourn time in the equivalence class (with respect to $\mathcal{R}$) of states* $\mathcal{K}$ is

$$SJ_{\mathcal{R}}(\mathcal{K}) = \frac{1}{1 - PM(\mathcal{K}, \mathcal{K})}.$$

The *average sojourn time vector for the equivalence classes (with respect to $\mathcal{R}$) of states* of $G$, denoted by $SJ_{\mathcal{R}}$, has elements $SJ_{\mathcal{R}}(\mathcal{K})$, with $\mathcal{K} \in DR(G)/_{\mathcal{R}}$. The *sojourn time variance in the equivalence class (with respect to $\mathcal{R}$) of states* $\mathcal{K}$ is

$$VAR_{\mathcal{R}}(\mathcal{K}) = \frac{PM(\mathcal{K}, \mathcal{K})}{(1 - PM(\mathcal{K}, \mathcal{K}))^2}.$$

The *sojourn time variance vector for the equivalence classes (with respect to $\mathcal{R}$) of states* of $G$, denoted by $VAR_{\mathcal{R}}$, has elements $VAR_{\mathcal{R}}(\mathcal{K})$, with $\mathcal{K} \in DR(G)/_{\mathcal{R}}$.

Let

$$\mathcal{R}_{ss}(G) = \bigcup \{\mathcal{R} \mid \mathcal{R} : G \underline{\leftrightarrow}_{ss} G\}$$

be the *union of all step stochastic autobisimulations* on $G$. By Proposition 5.2, $\mathcal{R}_{ss}(G)$ is the largest step stochastic autobisimulation on $G$. Based on the equivalence classes with respect to $\mathcal{R}_{ss}(G)$, the quotient (by $\underline{\leftrightarrow}_{ss}$) transition systems and the quotient (by $\underline{\leftrightarrow}_{ss}$) underlying DTMCs of expressions can be defined. These equivalence classes then become the quotient states. Every quotient transition between two such composite states represents all steps (having the same multiaction part in case of the transition system quotient) from the first state to the second.

**Definition 6.1.** Let $G$ be a dynamic expression. The *quotient (by $\underline{\leftrightarrow}_{ss}$) (labelled probabilistic) transition system* of $G$ is a quadruple

$$TS_{\underline{\leftrightarrow}_{ss}}(G) = (S_{\underline{\leftrightarrow}_{ss}}, L_{\underline{\leftrightarrow}_{ss}}, \mathcal{T}_{\underline{\leftrightarrow}_{ss}}, s_{\underline{\leftrightarrow}_{ss}}),$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

where

$$S_{\underline{\leftrightarrow}_{ss}} = DR(G)/_{\mathcal{R}_{ss}(G)}$$
$$L_{\underline{\leftrightarrow}_{ss}} \subseteq I\!N_f^{\mathcal{L}} \times (0;1]$$
$$\mathcal{T}_{\underline{\leftrightarrow}_{ss}} = \{(\mathcal{K}, (A, PM_A(\mathcal{K}, \widetilde{\mathcal{K}})), \widetilde{\mathcal{K}}) \mid \mathcal{K} \in DR(G)/_{\mathcal{R}_{ss}(G)},\ \mathcal{K} \stackrel{A}{\rightarrow} \widetilde{\mathcal{K}}\}$$
$$s_{\underline{\leftrightarrow}_{ss}} = [[G]_{\approx}]_{\mathcal{R}_{ss}(G)}.$$

The transition $(\mathcal{K}, (A, \mathcal{P}), \widetilde{\mathcal{K}}) \in \mathcal{T}_{\underline{\leftrightarrow}_{ss}}$ will be denoted by $\mathcal{K} \stackrel{A}{\rightarrow}_{\mathcal{P}} \widetilde{\mathcal{K}}$.

**Definition 6.2.** Let $G$ be a dynamic expression. The *quotient (by $\underline{\leftrightarrow}_{ss}$) underlying DTMC* of $G$, denoted by $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$, has state space $DR(G)/_{\mathcal{R}_{ss}(G)}$, initial state $[[G]_{\approx}]_{\mathcal{R}_{ss}(G)}$ and transitions $\mathcal{K} \rightarrow_{\mathcal{P}} \widetilde{\mathcal{K}}$, where $\mathcal{P} = PM(\mathcal{K}, \widetilde{\mathcal{K}})$.

The *quotient (by $\underline{\leftrightarrow}_{ss}$) average sojourn time vector* of $G$ is

$$SJ_{\underline{\leftrightarrow}_{ss}} = SJ_{\mathcal{R}_{ss}(G)}.$$

The *quotient (by $\underline{\leftrightarrow}_{ss}$) sojourn time variance vector* of $G$ is

$$VAR_{\underline{\leftrightarrow}_{ss}} = VAR_{\mathcal{R}_{ss}(G)}.$$

The quotients of both the transition systems and the underlying DTMCs are the minimal reductions of these objects modulo $\underline{\leftrightarrow}_{ss}$. The quotients can be used to simplify the analysis of system properties preserved by $\underline{\leftrightarrow}_{ss}$ since fewer states need to be examined for it. Such a reduction method resembles the one in Autant and Schnoebelen (1992) based on place bisimulation equivalence for PNs, except that our method merges states, while Autant and Schnoebelen's merges places.

Existing algorithms for constructing the quotients of transition systems by bisimulation (Paige and Tarjan 1987) and for (discrete or continuous time) Markov chains by ordinary lumping (Derisavi *et al.* 2003) can be adapted for our framework. These algorithms have time complexity $O(m \log n)$ and space complexity $O(m+n)$ (for the case of Markov chains), where $n$ is the number of states and $m$ is the number of transitions. As mentioned in Wimmer *et al.* (2010), it is easy to adapt the algorithm in Derisavi *et al.* (2003) to produce quotients of labelled probabilistic transition systems using the probabilistic bisimulation equivalence. Wimmer *et al.* (2010) proposed a symbolic partition refinement algorithm on the state space of CTMCs, and it is straightforward to adapt it to accommodate DTMCs and other Markovian models, Kripke structures and labelled probabilistic transition systems. Such a symbolic lumping uses memory efficiently because of the compact representation of the state space partition. The symbolic lumping is time efficient since a fast algorithm for the partition representation and refinement is applied.

We will present some comprehensive reduction examples in Section 8.

## 7. Stationary behaviour

We will now examine how the proposed equivalences can be used to compare the behaviour of stochastic processes in their steady states. We shall only consider formulas specifying

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

stochastic processes with infinite behaviour, that is, expressions containing the iteration operator. Note that the iteration operator does not guarantee infiniteness of behaviour since there may be a deadlock within the body (the second argument) of the iteration when the corresponding subprocess does not reach its final state for some reason. We will define the expression $\mathsf{Stop} = (\{c\}, \frac{1}{2})\ \mathsf{rs}\ c$ to specify the non-terminating process that only performs empty loops with probability 1. In particular, if the body of the iteration contains the Stop expression, the iteration will be 'broken'. On the other hand, the iteration body can be exited after a finite number of repeated executions, at which point the iteration termination is started. To avoid executing any activities after the iteration body, we take Stop as the termination argument for iteration.

As in the framework of DTMCs, the most common systems for performance analysis in DTSPNs are *ergodic* (recurrent non-null, aperiodic and irreducible). For ergodic DTSPNs, steady-state marking probabilities exist and can be determined. Molloy (1985) stated the following sufficient (but not necessary) conditions for the ergodicity of DTSPNs:

— *liveness* – for each transition and any reachable marking there exist a sequence of markings from it leading to the marking enabling that transition;
— *boundedness* – the number of tokens in every place is not greater than some fixed number for any reachable marking;
— *non-determinism* – the transition probabilities are strictly less than 1.

Given a dts-box of a dynamic expression with no deadlocks in the body of some iteration operator it contains and Stop as the termination argument of this operator, the three ergodicity conditions are satisfied: the subnet corresponding to such an iteration body is live, safe (1-bounded) and non-deterministic (since all markings of the live subnet are non-terminal, and the probabilities of transitions from them are strictly less than 1). Hence, the DTMC restricted to the states between the initial and final states of this iteration body is ergodic. The isomorphism between the DTMCs of expressions and the DTMCs of the corresponding dts-boxes, which is stated by Proposition 4.7, guarantees that the underlying DTMC of an expression with infinite behaviour is ergodic if it is restricted to the states in which such an iteration body is executed.

In this section, we only consider expressions for which the underlying DTMCs contain a single ergodic subset of states to guarantee that a single steady state exists.

### 7.1. *Theoretical background*

Let $G$ be a dynamic expression. The elements

$$\mathcal{P}_{ij}\ (1 \leqslant i, j \leqslant n = |DR(G)|)$$

of the (one-step) transition probability matrix (TPM) $\mathbf{P}$ for $DTMC(G)$ are defined by

$$\mathcal{P}_{ij} = \begin{cases} PM(s_i, s_j) & \text{if } s_i \to s_j \\ 0 & \text{otherwise.} \end{cases}$$

The transient ($k$-step, $k \in \mathbb{N}$) probability mass function (PMF)

$$\psi[k] = (\psi_1[k], \ldots, \psi_n[k])$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

for $DTMC(G)$ is a solution of the equation system

$$\psi[k] = \psi[0]\mathbf{P}^k,$$

where $\psi[0] = (\psi_1[0], \ldots, \psi_n[0])$ is the initial PMF defined by

$$\psi_i[0] = \begin{cases} 1 & \text{if } s_i = [G]_{\approx} \\ 0 & \text{otherwise.} \end{cases}$$

Note also that $\psi[k+1] = \psi[k]\mathbf{P}$ $(k \in \mathbb{N})$.

The steady-state PMF $\psi = (\psi_1, \ldots, \psi_n)$ for $DTMC(G)$ is a solution of the equation system

$$\begin{cases} \psi(\mathbf{P} - \mathbf{I}) = \mathbf{0} \\ \psi \mathbf{1}^T = 1 \end{cases}$$

where $\mathbf{I}$ is the identity matrix of size $n$, $\mathbf{0}$ is a row vector of $n$ values 0 and $\mathbf{1}$ is a row vector of $n$ values 1.

If $DTMC(G)$ has a single steady state, then $\psi = \lim_{k\to\infty} \psi[k]$. For

$$s = s_i \in DR(G) \quad (1 \leqslant i \leqslant n),$$

let

$$\psi[k](s) = \psi_i[k] \quad (k \in \mathbb{N})$$
$$\psi(s) = \psi_i.$$

Let $G$ be a dynamic expression and $s, \tilde{s} \in DR(G)$, $S, \widetilde{S} \subseteq DR(G)$. The following *performance indices (measures)* can be calculated based on the steady-state PMF for $DTMC(G)$:

— The *average recurrence (return) time in the state s* (that is, the number of discrete time units or steps required for this) is

$$\frac{1}{\psi(s)}.$$

— The *fraction of residence time in the state s* is $\psi(s)$.

— The *fraction of residence time in the set of states* $S \subseteq DR(G)$ or the *probability of the event determined by a condition that is true for all states from S* is

$$\sum_{s\in S} \psi(s).$$

— The *relative fraction of residence time in the set of states S with respect to that in* $\widetilde{S}$ is

$$\frac{\sum_{s\in S} \psi(s)}{\sum_{\tilde{s}\in\widetilde{S}} \psi(\tilde{s})}.$$

— The *steady-state probability of performing a step with an activity* $(\alpha, \rho)$ is

$$\sum_{s\in DR(G)} \psi(s) \sum_{\{\Gamma|(\alpha,\rho)\in\Gamma\}} PT(\Gamma, s).$$

— The *probability of the event determined by a reward function r* is

$$\sum_{s\in DR(G)} \psi(s) r(s).$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

### 7.2. *Steady state and equivalences*

The following proposition states that for two dynamic expressions related by $\underline{\leftrightarrow}_{ss}$, the steady-state probabilities of arriving in an equivalence class coincide. This result can also be interpreted as stating that the mean recurrence time for an equivalence class is the same for both expressions.

**Proposition 7.1.** Let $G, G'$ be dynamic expressions with $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$. Let $\psi$ be the steady-state PMF for $DTMC(G)$ and $\psi'$ be the steady-state PMF for $DTMC(G')$. Then for all $\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$ we have

$$\sum_{s \in \mathcal{H} \cap DR(G)} \psi(s) = \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'(s').$$

*Proof.* See Appendix A.2. □

Let $G$ be a dynamic expression. The steady-state PMF $\psi_{\underline{\leftrightarrow}_{ss}}$ for $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ is defined like the corresponding notion $\psi$ for $DTMC(G)$. By Proposition 7.1, for all $\mathcal{H} \in DR(G)/_{\mathcal{R}_{ss}(G)}$, we have

$$\psi_{\underline{\leftrightarrow}_{ss}}(\mathcal{H}) = \sum_{s \in \mathcal{H}} \psi(s).$$

Thus, for every equivalence class $\mathcal{H} \in DR(G)/_{\mathcal{R}_{ss}(G)}$, the value of $\psi_{\underline{\leftrightarrow}_{ss}}$ corresponding to $\mathcal{H}$ is the sum of all the values of $\psi$ corresponding to the states from $\mathcal{H}$. Hence, using $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ instead of $DTMC(G)$ simplifies the analytical solution since we have fewer states, though constructing the TPM for $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$, denoted by $\mathbf{P}_{\underline{\leftrightarrow}_{ss}}$, also requires some effort, including determining $\mathcal{R}_{ss}(G)$ and calculating the probabilities of moving from one equivalence class to another. The behaviour of $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ stabilises more quickly than that of $DTMC(G)$ (if each of them has a single steady state) since $\mathbf{P}_{\underline{\leftrightarrow}_{ss}}$ is a denser matrix than $\mathbf{P}$ because it is smaller and the transitions between the equivalence classes 'include' all the transitions between the states belonging to these equivalence classes.

By Proposition 7.1, $\underline{\leftrightarrow}_{ss}$ preserves the quantitative properties of the stationary behaviour. We will now demonstrate that the qualitative properties of the stationary behaviour based on the multiaction labels are also preserved.

**Definition 7.2.** A *derived step trace* of a dynamic expression $G$ is a chain

$$\Sigma = A_1 \cdots A_n \in (\mathbb{N}_f^{\mathcal{L}})^*$$

where there exists $s \in DR(G)$ such that $s \xrightarrow{\Gamma_1} s_1 \xrightarrow{\Gamma_2} \cdots \xrightarrow{\Gamma_n} s_n$ and $\mathcal{L}(\Gamma_i) = A_i$ $(1 \leqslant i \leqslant n)$. Then the *probability of executing the derived step trace* $\Sigma$ *in* $s$ is

$$PT(\Sigma, s) = \sum_{\{\Gamma_1, \ldots, \Gamma_n \mid s = s_0 \xrightarrow{\Gamma_1} s_1 \xrightarrow{\Gamma_2} \cdots \xrightarrow{\Gamma_n} s_n,\ \mathcal{L}(\Gamma_i) = A_i\ (1 \leqslant i \leqslant n)\}} \prod_{i=1}^{n} PT(\Gamma_i, s_{i-1}).$$

The following theorem demonstrates that for two dynamic expressions related by $\underline{\leftrightarrow}_{ss}$, the steady-state probabilities of arriving in an equivalence class and starting a derived step trace from it coincide.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

**Theorem 7.3.** Let $G, G'$ be dynamic expressions with $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$. Let $\psi$ be the steady-state PMF for $DTMC(G)$ and $\psi'$ be the steady-state PMF for $DTMC(G')$. Let $\Sigma$ be a derived step trace of $G$ and $G'$. Then for all $\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$, we have

$$\sum_{s \in \mathcal{H} \cap DR(G)} \psi(s) PT(\Sigma, s) = \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'(s') PT(\Sigma, s').$$

*Proof.* See Appendix A.3. □

**Example 7.4.** Let

$$E = [(\{a\}, \tfrac{1}{2}) * ((\{b\}, \tfrac{1}{2}); ((\{c\}, \tfrac{1}{3})_1 \,[]\, (\{c\}, \tfrac{1}{3})_2)) * \mathsf{Stop}]$$
$$E' = [(\{a\}, \tfrac{1}{2}) * (((\{b\}, \tfrac{1}{3})_1 ; (\{c\}, \tfrac{1}{2})_1) [] ((\{b\}, \tfrac{1}{3})_2 ; (\{c\}, \tfrac{1}{2})_2)) * \mathsf{Stop}].$$

We have $\overline{E} =_{sto} \overline{E'}$, hence, $\overline{E} \underline{\leftrightarrow}_{ss} \overline{E'}$.

$DR(\overline{E})$ consists of the equivalence classes

$$s_1 = [[\overline{(\{a\}, \tfrac{1}{2})} * ((\{b\}, \tfrac{1}{2}); ((\{c\}, \tfrac{1}{3})_1 \,[]\, (\{c\}, \tfrac{1}{3})_2)) * \mathsf{Stop}]]_{\approx}$$
$$s_2 = [[(\{a\}, \tfrac{1}{2}) * \overline{((\{b\}, \tfrac{1}{2}); ((\{c\}, \tfrac{1}{3})_1 \,[]\, (\{c\}, \tfrac{1}{3})_2))} * \mathsf{Stop}]]_{\approx}$$
$$s_3 = [[(\{a\}, \tfrac{1}{2}) * ((\{b\}, \tfrac{1}{2}); \overline{((\{c\}, \tfrac{1}{3})_1 \,[]\, (\{c\}, \tfrac{1}{3})_2)}) * \mathsf{Stop}]]_{\approx}.$$

$DR(\overline{E'})$ consists of the equivalence classes

$$s'_1 = [[\overline{(\{a\}, \tfrac{1}{2})} * (((\{b\}, \tfrac{1}{3})_1 ; (\{c\}, \tfrac{1}{2})_1) [] ((\{b\}, \tfrac{1}{3})_2 ; (\{c\}, \tfrac{1}{2})_2)) * \mathsf{Stop}]]_{\approx}$$
$$s'_2 = [[(\{a\}, \tfrac{1}{2}) * \overline{(((\{b\}, \tfrac{1}{3})_1 ; (\{c\}, \tfrac{1}{2})_1) [] ((\{b\}, \tfrac{1}{3})_2 ; (\{c\}, \tfrac{1}{2})_2))} * \mathsf{Stop}]]_{\approx}$$
$$s'_3 = [[(\{a\}, \tfrac{1}{2}) * (((\{b\}, \tfrac{1}{3})_1 ; \overline{(\{c\}, \tfrac{1}{2})_1}) [] ((\{b\}, \tfrac{1}{3})_2 ; (\{c\}, \tfrac{1}{2})_2)) * \mathsf{Stop}]]_{\approx}$$
$$s'_4 = [[(\{a\}, \tfrac{1}{2}) * (((\{b\}, \tfrac{1}{3})_1 ; (\{c\}, \tfrac{1}{2})_1) [] ((\{b\}, \tfrac{1}{3})_2 ; \overline{(\{c\}, \tfrac{1}{2})_2})) * \mathsf{Stop}]]_{\approx}.$$

The steady-state PMFs $\psi$ for $DTMC(\overline{E})$ and $\psi'$ for $DTMC(\overline{E'})$ are

$$\psi = \left(0, \tfrac{1}{2}, \tfrac{1}{2}\right)$$
$$\psi' = \left(0, \tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{4}\right).$$

Consider the equivalence class (with respect to $\mathcal{R}_{ss}(\overline{E}, \overline{E'})$) $\mathcal{H} = \{s_3, s'_3, s'_4\}$. It is easy to see that the steady-state probabilities for $\mathcal{H}$ coincide:

$$\begin{aligned}
\sum_{s \in \mathcal{H} \cap DR(\overline{E})} \psi(s) &= \psi(s_3) \\
&= \tfrac{1}{2} \\
&= \tfrac{1}{4} + \tfrac{1}{4} \\
&= \psi'(s'_3) + \psi'(s'_4) \\
&= \sum_{s' \in \mathcal{H} \cap DR(\overline{E'})} \psi'(s').
\end{aligned}$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

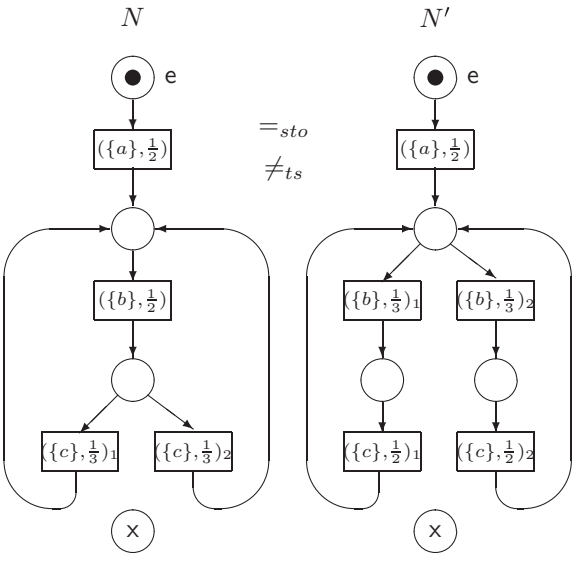



Fig. 8. $\underline{\leftrightarrow}_{ss}$ implies a coincidence of the steady-state probabilities of arriving in an equivalence class and starting a derived step trace from it.

Let $\Sigma = \{\{c\}\}$. The steady-state probabilities of arriving in the equivalence class $\mathcal{H}$ and starting the derived step trace $\Sigma$ from it also coincide:

$$\begin{aligned}\psi(s_3)(PT(\{(\{c\},\tfrac{1}{3})_1\},s_3) \; + PT(\{(\{c\},\tfrac{1}{3})_2\},s_3)) &= \\ &= \tfrac{1}{2}\left(\tfrac{1}{4}+\tfrac{1}{4}\right)\\ &= \tfrac{1}{4}\\ &= \tfrac{1}{4}\cdot\tfrac{1}{2}+\tfrac{1}{4}\cdot\tfrac{1}{2}\\ &= \psi'(s'_3)PT(\{(\{c\},\tfrac{1}{2})_1\},s'_3)\\ &\quad + \psi'(s'_4)PT(\{(\{c\},\tfrac{1}{2})_2\},s'_4).\end{aligned}$$

Figure 8 shows the marked dts-boxes corresponding to these dynamic expressions, that is, $N = Box_{dts}(\overline{E})$ and $N' = Box_{dts}(\overline{E'})$.

### 7.3. *The preservation of performance and the simplification of its analysis*

Many performance indices are based on the steady-state probabilities of arriving in a set of similar states or, after arriving in it, starting a derived step trace from this set. The similarity of states is usually captured by an equivalence relation, hence, the sets are often the equivalence classes. Proposition 7.1 and Theorem 7.3 guarantee the coincidence of these indices for the expressions related by $\underline{\leftrightarrow}_{ss}$. Hence, $\underline{\leftrightarrow}_{ss}$ (and thus all the stronger equivalences we have considered) preserves performance of stochastic systems modelled by expressions of dtsPBC.

In addition, it is easier to evaluate performance using a DTMC with fewer states since in this case the size of the transition probability matrix is smaller, and we need to solve

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

systems with fewer equations to calculate the steady-state probabilities. This reasoning validates the following method of performance analysis simplification:

(1) Specify the system under investigation using a static expression of dtsPBC.
(2) Construct the transition system for the expression.
(3) After treating the transition system for self-similarity, determine a step stochastic autobisimulation equivalence for the expression.
(4) Construct the quotient underlying DTMC from the quotient transition system.
(5) Calculate the stationary probabilities and performance indices using the DTMC.

The limitation of this method is that it is only applicable to expressions for which the corresponding DTMCs contain a single irreducible subset of states, that is, the existence of exactly one stationary state is required. If a DTMC contains several irreducible subsets of states, there may be several steady states that depend on the initial PMF. There is also an analytical method to determine the stable states for DTMCs of this kind (Kulkarni 2009). Note that for every expression, the underlying DTMC has, by definition, only one initial PMF (the one at the time moment 0), so the stationary state will be unique in this case too. The general steady-state probability will be calculated as a sum of the stationary probabilities of the irreducible subsets of states weighted by the probabilities of entering these subsets starting from the initial state and passing through some transient states. Furthermore, it is only worth applying the method to systems with similar subprocesses.

An alternative for reducing transition systems is to use an analogue of the approach in Katoen *et al.* (2011): first perform a fast symmetry reduction based on the method in Kwiatkowska *et al.* (2006), then construct a quotient of the resulting transition system using bisimulation equivalence by applying the time-optimal partition refinement algorithm in Derisavi *et al.* (2003) to the state space of this system. As mentioned in Katoen *et al.* (2011), minimisation by bisimulation in a number of case studies results in a more significant state space reduction than symmetry reduction, but the latter is much faster than the former since symmetries are determined on a syntactical level. Baarir *et al.* (2011) proposed some effective analysis methods for partially symmetric models.

## 8. Dining philosophers system

### 8.1. *The standard system*

In this section, we consider a model of five dining philosophers, for which a Petri net interpretation was proposed in Peterson (1981). We will investigate this dining philosophers system in the discrete time stochastic setting of dtsPBC. The philosophers occupy a round table, and there is one fork lying between each neighbour, so there are five forks on the table. A philosopher needs two forks to eat, namely, both the fork to his left and the fork to his right. This means that it is impossible for all five philosophers to eat at the same time since there are not enough forks available: only one of the philosophers or two non-neighbouring philosophers can eat at the same time. The model works as follows. After activation of the system (when the philosophers come into the dining room), five forks are placed on the table. If both the left and right forks are available for a philosopher, he takes them simultaneously and begins eating. After eating, the philosopher places both

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

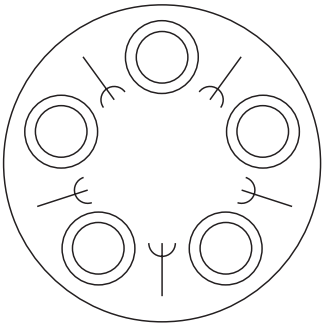


Fig. 9. The diagram of the dining philosophers system.

his forks simultaneously back on the table. The strategy of picking up and releasing two forks simultaneously prevents the situation where a philosopher takes one fork but is not able to pick up the second one because his neighbour has already done so. In particular, we avoid a deadlock when all the philosophers take their left (or right) forks and wait until their right (or left) forks become available. Figure 9 illustrates the system.

One can explore what happens for different numbers of philosophers at the table. The most interesting question is to find the maximal sets of philosophers that can eat at the same time since all other combinations of diners will be subsets of these maximal sets:

— For the system with 1 philosopher, the only maximal set is $\emptyset$.
— For 2 philosophers, the maximal sets are $\{1\}$, $\{2\}$.
— For 3 philosophers, the maximal sets are $\{1\}$, $\{2\}$, $\{3\}$.
— For 4 philosophers, the maximal sets are $\{1,3\}$, $\{2,4\}$.
— For 5 philosophers, the maximal sets are $\{1,3\}$, $\{1,4\}$, $\{2,4\}$, $\{2,5\}$, $\{3,5\}$.
— For 6 philosophers, the maximal sets are $\{1,4\}$, $\{2,5\}$, $\{3,6\}$, $\{1,3,5\}$, $\{2,4,6\}$.
— For 7 philosophers, the maximal sets are $\{1,3,5\}$, $\{1,3,6\}$, $\{1,4,6\}$, $\{2,4,6\}$, $\{2,4,7\}$, $\{2,5,7\}$, $\{3,5,7\}$.

Thus, the system demonstrates a non-trivial behaviour when there are at least 5 philosophers.

Since neighbours cannot eat at the same time, the maximal number of diners for the system with $n$ philosophers will be $\lfloor \frac{n}{2} \rfloor$, that is, the maximal natural number that is not greater than $\frac{n}{2}$. If philosopher $i$ belongs to some maximal set, then philosopher $i(\mathsf{mod}\ n)+1$ will belong to the next one. Let us calculate the number of such maximal sets consisting of the maximal number of philosophers ($\lfloor \frac{n}{2} \rfloor$). If $n$ is an even number, there will only be 2 such maximal sets of $\frac{n}{2}$ diners, namely, the philosophers numbered with all odd natural numbers that are not greater than $n$ and those numbered with all even natural numbers that are not greater than $n$. If $n$ is an odd number, there will be $n$ such maximal sets of $\frac{n-1}{2}$ diners since, starting from some maximal set, we can 'shift' clockwise $n-1$ times by one element modulo $n$ until the next maximal set coincides with the initial one.

Returning to the 5 dining philosophers system, we will now explain the meaning of the actions in the syntax of dtsPBC expressions specifying the system modules. The action $a$ corresponds to the system activation, and the actions $b_i$ and $e_i$ correspond to the beginning and end, respectively, of philosopher $i$ ($1 \leqslant i \leqslant 5$) eating. The other actions

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

are only used for communication purposes through synchronisation, and we will abstract from them later using restriction. Note that the expression for each philosopher includes two alternative subexpressions such that the second of them specifies a resource (fork) sharing with the right-hand neighbour.

The static expression for philosopher $i$ ($1 \leqslant i \leqslant 4$) is

$$E_i = [(\{x_i\}, \tfrac{1}{2}) * ((((\{b_i, \widehat{y_i}\}, \tfrac{1}{2}); (\{e_i, \widehat{z_i}\}, \tfrac{1}{2}))[]((\{y_{i+1}\}, \tfrac{1}{2}); (\{z_{i+1}\}, \tfrac{1}{2}))) * \mathsf{Stop}].$$

And the static expression for philosopher 5 is

$$E_5 = [(\{a, \widehat{x_1}, \widehat{x_2}, \widehat{x_2}, \widehat{x_4}\}, \tfrac{1}{2}) * ((((\{b_5, \widehat{y_5}\}, \tfrac{1}{2}); (\{e_5, \widehat{z_5}\}, \tfrac{1}{2}))[]((\{y_1\}, \tfrac{1}{2}); (\{z_1\}, \tfrac{1}{2}))) * \mathsf{Stop}].$$

For $a_1, \ldots, a_n \in Act$ ($n \in \mathbb{N}$), we shall write $\mathsf{sr}\ (a_1, \ldots, a_n)$ as an abbreviation for $\mathsf{sy}\ a_1 \cdots \mathsf{sy}\ a_n\ \mathsf{rs}\ a_1 \cdots \mathsf{rs}\ a_n$. The static expression of the dining philosophers system is then

$$E = (E_1 \| E_2 \| E_3 \| E_4 \| E_5)\ \mathsf{sr}\ (x_1, x_2, x_3, x_4, y_1, y_2, y_3, y_4, y_5, z_1, z_2, z_3, z_4, z_5).$$

To illustrate the effect of synchronisation, we have:

— Synchronising $(\{b_i, y_i\}, \frac{1}{2})$ and $(\{\widehat{y_i}\}, \frac{1}{2})$ gives the new activity $(\{b_i\}, \frac{1}{4})$ $(1 \leqslant i \leqslant 5)$.
— Synchronising $(\{e_i, z_i\}, \frac{1}{2})$ and $(\{\widehat{z_i}\}, \frac{1}{2})$ gives $(\{e_i\}, \frac{1}{4})$ $(1 \leqslant i \leqslant 5)$.
— Synchronising $(\{a, \widehat{x_1}, \widehat{x_2}, \widehat{x_3}, \widehat{x_4}\}, \frac{1}{2})$ and $(\{x_1\}, \frac{1}{2})$ gives $(\{a, \widehat{x_2}, \widehat{x_3}, \widehat{x_4}\}, \frac{1}{4})$.
— Synchronising $(\{a, \widehat{x_2}, \widehat{x_3}, \widehat{x_4}\}, \frac{1}{4})$ and $(\{x_2\}, \frac{1}{2})$ gives $(\{a, \widehat{x_3}, \widehat{x_4}\}, \frac{1}{8})$.
— Synchronising $(\{a, \widehat{x_3}, \widehat{x_4}\}, \frac{1}{8})$ and $(\{x_3\}, \frac{1}{2})$ gives $(\{a, \widehat{x_4}\}, \frac{1}{16})$.
— Synchronising $(\{a, \widehat{x_4}\}, \frac{1}{16})$ and $(\{x_4\}, \frac{1}{2})$ gives $(\{a\}, \frac{1}{32})$.

$DR(\overline{E})$ has 12 states, which are interpreted as follows:

$s_1$: the initial state
$s_2$: the system is activated and no philosophers are dining yet
$s_3$: philosopher 1 dines
$s_4$: philosophers 1 and 4 dine
$s_5$: philosophers 1 and 3 dine
$s_6$: philosopher 4 dines
$s_7$: philosopher 3 dines
$s_8$: philosophers 2 and 4 dine
$s_9$: philosophers 3 and 5 dine
$s_{10}$: philosopher 2 dines
$s_{11}$: philosopher 5 dines
$s_{12}$: philosophers 2 and 5 dine.

Figure 10 shows the transition system $TS(\overline{E})$.

The average sojourn time vector of $\overline{E}$ is

$$SJ = \left(32, \tfrac{29}{20}, \tfrac{20}{11}, \tfrac{16}{7}, \tfrac{16}{7}, \tfrac{20}{11}, \tfrac{20}{11}, \tfrac{16}{7}, \tfrac{16}{7}, \tfrac{20}{11}, \tfrac{20}{11}, \tfrac{16}{7}\right).$$

The sojourn time variance vector of $\overline{E}$ is

$$VAR = \left(992, \tfrac{261}{400}, \tfrac{180}{121}, \tfrac{144}{49}, \tfrac{144}{49}, \tfrac{180}{121}, \tfrac{180}{121}, \tfrac{144}{49}, \tfrac{144}{49}, \tfrac{180}{121}, \tfrac{180}{121}, \tfrac{144}{49}\right).$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

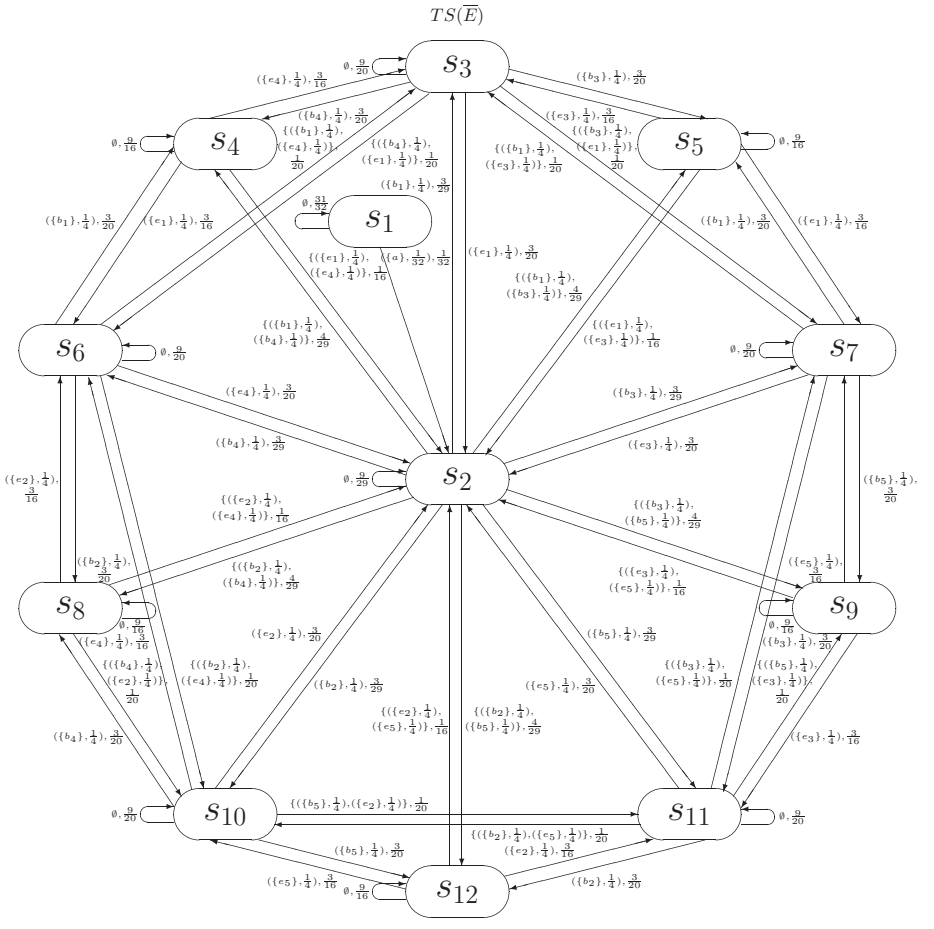



Fig. 10. The transition system for the dining philosophers system.

The transition probability matrix (TPM) for $DTMC(\overline{E})$ is

$$\mathbf{P} = \begin{pmatrix}
\frac{31}{32} & \frac{1}{32} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{9}{29} & \frac{3}{29} & \frac{1}{29} & \frac{1}{29} & \frac{3}{29} & \frac{3}{29} & \frac{1}{29} & \frac{1}{29} & \frac{3}{29} & \frac{3}{29} & \frac{1}{29} \\
0 & \frac{3}{20} & \frac{9}{20} & \frac{3}{20} & \frac{3}{20} & \frac{1}{20} & \frac{1}{20} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{16} & \frac{3}{16} & \frac{9}{16} & 0 & \frac{3}{16} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{16} & \frac{3}{16} & 0 & \frac{9}{16} & 0 & \frac{3}{16} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{3}{20} & \frac{1}{20} & \frac{3}{20} & 0 & \frac{9}{20} & 0 & \frac{3}{20} & 0 & \frac{1}{20} & 0 & 0 \\
0 & \frac{3}{20} & \frac{1}{20} & 0 & \frac{3}{20} & 0 & \frac{9}{20} & 0 & \frac{3}{20} & 0 & \frac{1}{20} & 0 \\
0 & \frac{1}{16} & 0 & 0 & 0 & \frac{3}{16} & 0 & \frac{9}{16} & 0 & \frac{3}{16} & 0 & 0 \\
0 & \frac{1}{16} & 0 & 0 & 0 & 0 & \frac{3}{16} & 0 & \frac{9}{16} & 0 & \frac{3}{16} & 0 \\
0 & \frac{3}{20} & 0 & 0 & 0 & \frac{1}{20} & 0 & \frac{3}{20} & 0 & \frac{9}{20} & \frac{1}{20} & \frac{3}{20} \\
0 & \frac{3}{20} & 0 & 0 & 0 & 0 & \frac{1}{20} & 0 & \frac{3}{20} & \frac{1}{20} & \frac{9}{20} & \frac{3}{20} \\
0 & \frac{1}{16} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3}{16} & \frac{3}{16} & \frac{9}{16}
\end{pmatrix}.$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Table 4. *Transient and steady-state probabilities for the dining philosophers system.*

| k | 0 | 20 | 40 | 60 | 80 | 100 | 120 | 140 | 160 | 180 | 200 | $\infty$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\psi_1[k]$ | 1 | 0.5299 | 0.2808 | 0.1488 | 0.0789 | 0.0418 | 0.0222 | 0.0117 | 0.0062 | 0.0033 | 0.0017 | 0 |
| $\psi_2[k]$ | 0 | 0.0842 | 0.1098 | 0.1234 | 0.1306 | 0.1345 | 0.1365 | 0.1375 | 0.1381 | 0.1384 | 0.1386 | 0.1388 |
| $\psi_3[k]$ | 0 | 0.0437 | 0.0681 | 0.0811 | 0.0880 | 0.0916 | 0.0935 | 0.0945 | 0.0951 | 0.0954 | 0.0955 | 0.0957 |
| $\psi_4[k]$ | 0 | 0.0335 | 0.0537 | 0.0645 | 0.0701 | 0.0732 | 0.0748 | 0.0756 | 0.0760 | 0.0763 | 0.0764 | 0.0766 |

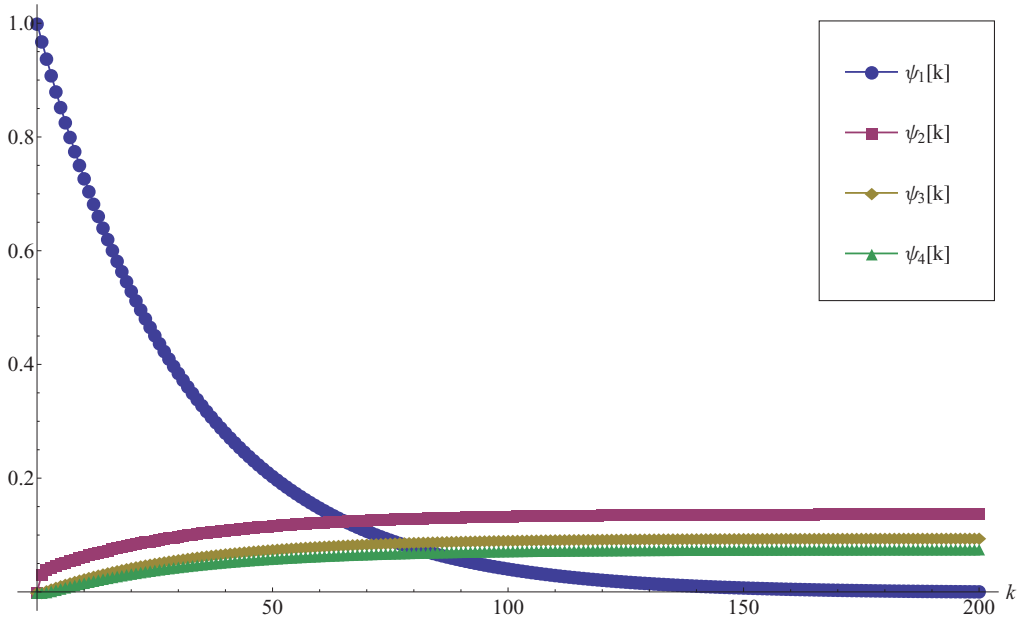



Fig. 11. (Colour online) Transient probabilities alteration diagram for the dining philosophers system.

Table 4 shows the transient and steady-state probabilities $\psi_i[k]$ $(1 \leqslant i \leqslant 4)$ for the dining philosophers system at the time moments $k \in \{0, 20, 40, \ldots, 200\}$ and $k = \infty$, and Figure 11 shows the alteration diagram (evolution in time) for the transient probabilities. It is sufficient just to consider the probabilities for the states $s_1, \ldots, s_4$ since the corresponding values are the same for $s_3, s_6, s_7, s_{10}, s_{11}$, and for $s_4, s_5, s_8, s_9, s_{12}$.

The steady-state PMF for $DTMC(\overline{E})$ is

$$\psi = \left(0, \tfrac{29}{209}, \tfrac{20}{209}, \tfrac{16}{209}, \tfrac{16}{209}, \tfrac{20}{209}, \tfrac{20}{209}, \tfrac{16}{209}, \tfrac{16}{209}, \tfrac{20}{209}, \tfrac{20}{209}, \tfrac{16}{209}\right).$$

We can now calculate the main performance indices as follows:

— The average recurrence time in the state $s_2$, where all the forks are available, which is called the *average system run-through*, is

$$\frac{1}{\psi_2} = \tfrac{209}{29} = 7\tfrac{6}{29}.$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

— Nobody eats in the state $s_2$, so the *fraction of time when no philosophers dine* is

$$\psi_2 = \tfrac{29}{209}.$$

Only one philosopher eats in the states $s_3, s_6, s_7, s_{10}, s_{11}$, so the *fraction of time when only one philosopher dines* is

$$\psi_3 + \psi_6 + \psi_7 + \psi_{10} + \psi_{11} = \tfrac{20}{209} + \tfrac{20}{209} + \tfrac{20}{209} + \tfrac{20}{209} + \tfrac{20}{209} = \tfrac{100}{209}.$$

Two philosophers eat together in the states $s_4, s_5, s_8, s_9, s_{12}$, so the *fraction of time when two philosophers dine* is

$$\psi_4 + \psi_5 + \psi_8 + \psi_9 + \psi_{12} = \tfrac{16}{209} + \tfrac{16}{209} + \tfrac{16}{209} + \tfrac{16}{209} + \tfrac{16}{209} = \tfrac{80}{209}.$$

The *relative fraction of time when two philosophers dine compared with when only one philosopher dines* is

$$\tfrac{80}{209} \cdot \tfrac{209}{100} = \tfrac{4}{5}.$$

— The action of the first philosopher beginning to eat, $(\{b_1\}, \frac{1}{4})$, is only possible from the states $s_2, s_6, s_7$. In each of these states, the probability of the first philosopher beginning to eat is the sum of the execution probabilities for all multisets of activities containing $(\{b_1\}, \frac{1}{4})$. Thus, the *steady-state probability of the first philosopher beginning to eat* is

$$\psi_2 \sum_{\{\Gamma|(\{b_1\},\frac{1}{4})\in\Gamma\}} PT(\Gamma, s_2) \; + \psi_6 \sum_{\{\Gamma|(\{b_1\},\frac{1}{4})\in\Gamma\}} PT(\Gamma, s_6) \; + \psi_7 \sum_{\{\Gamma|(\{b_1\},\frac{1}{4})\in\Gamma\}} PT(\Gamma, s_7)$$

$$= \tfrac{29}{209}\left(\tfrac{3}{29} + \tfrac{1}{29} + \tfrac{1}{29}\right) + \tfrac{20}{209}\left(\tfrac{3}{20} + \tfrac{1}{20}\right) + \tfrac{20}{209}\left(\tfrac{3}{20} + \tfrac{1}{20}\right)$$
$$= \tfrac{13}{209}.$$

Figure 12 shows the marked dts-boxes corresponding to the dynamic expressions for the individual dining philosophers, that is, $N_i = Box_{dts}(\overline{E_i})$ $(1 \leqslant i \leqslant 5)$. Figure 13 shows the marked dts-box corresponding to the dynamic expression for the dining philosophers system, that is, $N = Box_{dts}(\overline{E})$.

### 8.2. *The abstract system and its reduction*

We will now consider a modification of the dining philosophers system that abstracts from personalities so that all the philosophers are indistinguishable. For example, we can tell that one or two philosophers are eating, but not who they are. We call this system the abstract dining philosophers system. To implement the abstraction, we replace the actions $b_i$ and $e_i$ $(1 \leqslant i \leqslant 5)$ in the system specification by $b$ and $e$, respectively.

The static expression for philosopher $i$ $(1 \leqslant i \leqslant 4)$ is

$$F_i = [(\{x_i\}, \tfrac{1}{2}) * ((((\{b, \widehat{y_i}\}, \tfrac{1}{2}); (\{e, \widehat{z_i}\}, \tfrac{1}{2}))[]((\{y_{i+1}\}, \tfrac{1}{2}); (\{z_{i+1}\}, \tfrac{1}{2}))) * \mathsf{Stop}].$$

And the static expression for philosopher 5 is

$$F_5 = [(\{a, \widehat{x_1}, \widehat{x_2}, \widehat{x_2}, \widehat{x_4}\}, \tfrac{1}{2}) * ((((\{b, \widehat{y_5}\}, \tfrac{1}{2}); (\{e, \widehat{z_5}\}, \tfrac{1}{2}))[]((\{y_1\}, \tfrac{1}{2}); (\{z_1\}, \tfrac{1}{2}))) * \mathsf{Stop}].$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

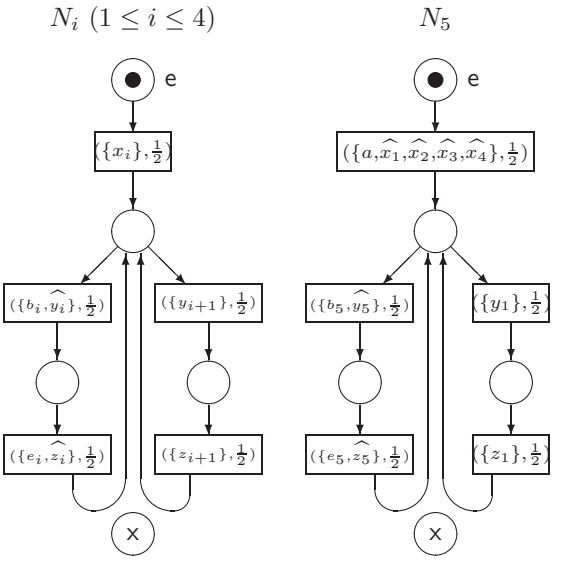



Fig. 12. The marked dts-boxes for the dining philosophers.

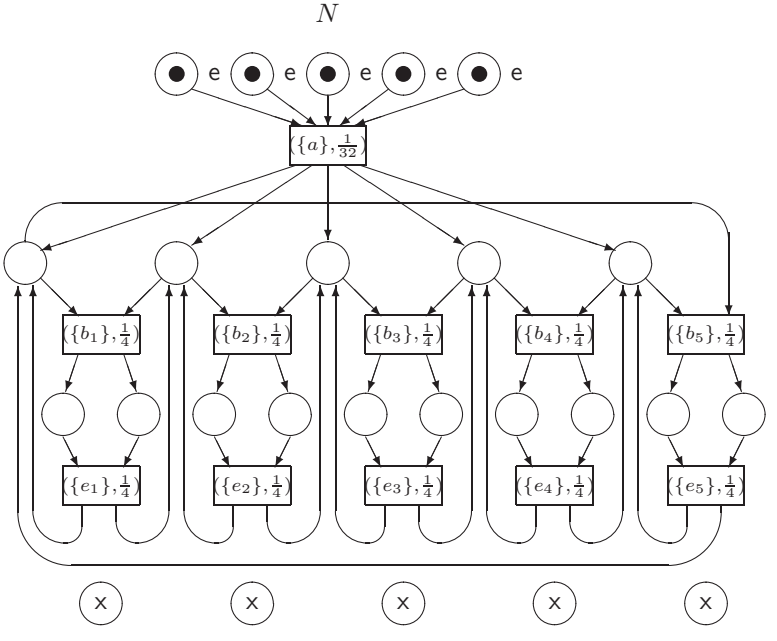



Fig. 13. The marked dts-box for the dining philosophers system.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

The static expression for the abstract dining philosophers system is

$$F = (F_1 \| F_2 \| F_3 \| F_4 \| F_5) \text{ sr } (x_1, x_2, x_3, x_4, y_1, y_2, y_3, y_4, y_5, z_1, z_2, z_3, z_4, z_5).$$

$DR(\overline{F})$ resembles $DR(\overline{E})$, and $TS(\overline{F})$ is similar to $TS(\overline{E})$. Also,

$$DTMC(\overline{F}) = DTMC(\overline{E}).$$

Thus, the TPM and the steady-state PMF for $DTMC(\overline{F})$ and $DTMC(\overline{E})$ coincide.

Both the first performance index and the second group of indices are the same for the standard and abstract systems. The following performance index is based on the depersonalised viewpoint:

— The action of a philosopher beginning to eat, $(\{b\}, \frac{1}{4})$, is only possible from the states $s_2, s_3, s_6, s_7, s_{10}, s_{11}$. In each of these states, the probability of beginning to eat is the sum of the execution probabilities for all multisets of activities containing $(\{b\}, \frac{1}{4})$. Thus, the *steady-state probability of a philosopher beginning to eat* is

$$\begin{aligned}
&\psi_2 \sum_{\{\Gamma | (\{b\}, \frac{1}{4}) \in \Gamma\}} PT(\Gamma, s_2) + \psi_3 \sum_{\{\Gamma | (\{b\}, \frac{1}{4}) \in \Gamma\}} PT(\Gamma, s_3) \\
&+ \psi_6 \sum_{\{\Gamma | (\{b\}, \frac{1}{4}) \in \Gamma\}} PT(\Gamma, s_6) + \psi_7 \sum_{\{\Gamma | (\{b\}, \frac{1}{4}) \in \Gamma\}} PT(\Gamma, s_7) \\
&+ \psi_{10} \sum_{\{\Gamma | (\{b\}, \frac{1}{4}) \in \Gamma\}} PT(\Gamma, s_{10}) + \psi_{11} \sum_{\{\Gamma | (\{b\}, \frac{1}{4}) \in \Gamma\}} PT(\Gamma, s_{11}) \\
&= \tfrac{29}{209} \left(\tfrac{3}{29} + \tfrac{1}{29} + \tfrac{3}{29} + \tfrac{1}{29} + \tfrac{3}{29} + \tfrac{1}{29} + \tfrac{3}{29} + \tfrac{1}{29} + \tfrac{3}{29} + \tfrac{1}{29}\right) \\
&\quad + \tfrac{20}{209} \left(\tfrac{3}{20} + \tfrac{1}{20} + \tfrac{3}{20} + \tfrac{1}{20}\right) \\
&\quad + \tfrac{20}{209} \left(\tfrac{3}{20} + \tfrac{1}{20} + \tfrac{3}{20} + \tfrac{1}{20}\right) + \tfrac{20}{209} \left(\tfrac{3}{20} + \tfrac{1}{20} + \tfrac{3}{20} + \tfrac{1}{20}\right) \\
&\quad + \tfrac{20}{209} \left(\tfrac{3}{20} + \tfrac{1}{20} + \tfrac{3}{20} + \tfrac{1}{20}\right) + \tfrac{20}{209} \left(\tfrac{3}{20} + \tfrac{1}{20} + \tfrac{3}{20} + \tfrac{1}{20}\right) \\
&= \tfrac{60}{209}.
\end{aligned}$$

The marked dts-boxes corresponding to the dynamic expressions of the standard and abstract dining philosophers are similar, as are the marked dts-boxes corresponding to the dynamic expression of the standard and abstract dining philosophers systems.

We have

$$DR(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4\},$$

where

$\mathcal{K}_1 = \{s_1\}$ (the initial state)
$\mathcal{K}_2 = \{s_2\}$ (the system is activated and no philosophers dine)
$\mathcal{K}_3 = \{s_3, s_6, s_7, s_{10}, s_{11}\}$ (one philosopher dines)
$\mathcal{K}_4 = \{s_4, s_5, s_8, s_9, s_{12}\}$ (two philosophers dine).

Figure 14 shows the quotient transition system $TS_{\underline{\leftrightarrow}_{ss}}(\overline{F})$.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

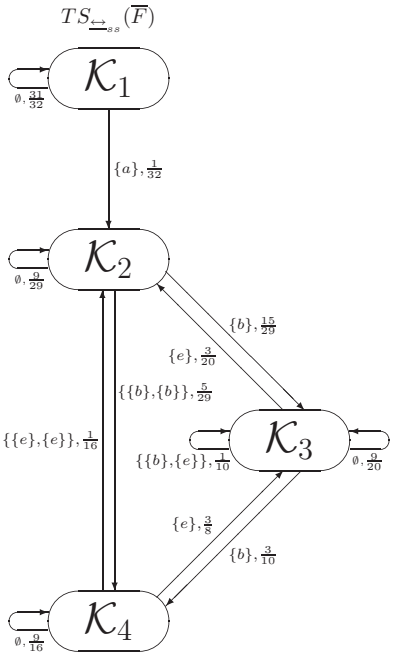



Fig. 14. The quotient transition system for the abstract dining philosophers system.

The quotient average sojourn time vector of $\overline{F}$ is

$$SJ' = \left(32, \tfrac{29}{20}, \tfrac{20}{9}, \tfrac{16}{7}\right).$$

The quotient sojourn time variance vector of $\overline{F}$ is

$$VAR' = \left(992, \tfrac{261}{400}, \tfrac{180}{121}, \tfrac{144}{49}\right).$$

The TPM for $DTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is

$$\mathbf{P}' = \begin{pmatrix} \frac{31}{32} & \frac{1}{32} & 0 & 0 \\ 0 & \frac{9}{29} & \frac{15}{29} & \frac{5}{29} \\ 0 & \frac{3}{20} & \frac{11}{20} & \frac{3}{10} \\ 0 & \frac{1}{16} & \frac{3}{8} & \frac{9}{16} \end{pmatrix}.$$

Table 5 shows the transient and steady-state probabilities $\psi'_i[k]$ $(1 \leqslant i \leqslant 4)$ for the quotient abstract dining philosophers system at the time moments $k \in \{0, 20, 40, \ldots, 200\}$ and $k = \infty$, and Figure 15 shows the alteration diagram (evolution in time) for the transient probabilities.

The steady-state PMF for $DTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is

$$\psi' = \left(0, \tfrac{29}{209}, \tfrac{100}{209}, \tfrac{80}{209}\right).$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Table 5. *Transient and steady-state probabilities for the quotient abstract dining philosophers system.*

| k | 0 | 20 | 40 | 60 | 80 | 100 | 120 | 140 | 160 | 180 | 200 | $\infty$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\psi_1'[k]$ | 1 | 0.5299 | 0.2808 | 0.1488 | 0.0789 | 0.0418 | 0.0222 | 0.0117 | 0.0062 | 0.0033 | 0.0017 | 0 |
| $\psi_2'[k]$ | 0 | 0.0842 | 0.1098 | 0.1234 | 0.1306 | 0.1345 | 0.1365 | 0.1375 | 0.1381 | 0.1384 | 0.1386 | 0.1388 |
| $\psi_3'[k]$ | 0 | 0.2183 | 0.3406 | 0.4054 | 0.4398 | 0.4580 | 0.4676 | 0.4727 | 0.4754 | 0.4769 | 0.4776 | 0.4785 |
| $\psi_4'[k]$ | 0 | 0.1675 | 0.2687 | 0.3223 | 0.3507 | 0.3658 | 0.3738 | 0.3780 | 0.3802 | 0.3814 | 0.3821 | 0.3828 |

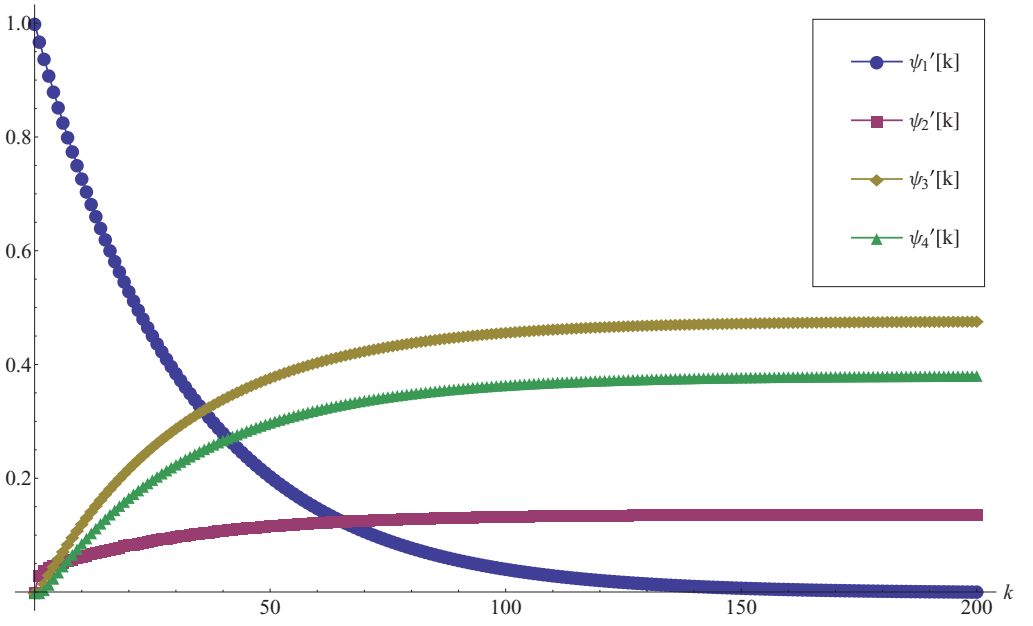



Fig. 15. (Colour online) Transient probabilities alteration diagram for the quotient abstract dining philosophers system.

We can now calculate the main performance indices as follows:

— The average recurrence time in the state $\mathcal{K}_2$, where all the forks are available, which is called the *average system run-through*, is

$$\frac{1}{\psi_2'} = \frac{209}{29} = 7\frac{6}{29}.$$

— Nobody eats in the state $\mathcal{K}_2$, so the *fraction of time when no philosophers dine* is

$$\psi_2' = \frac{29}{209}.$$

Only one philosopher eats in the state $\mathcal{K}_3$, so the *fraction of time when only one philosopher dines* is

$$\psi_3' = \frac{100}{209}.$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Two philosophers eat together in the state $\mathcal{K}_4$, so the *fraction of time when two philosophers dine* is

$$\psi'_4 = \tfrac{80}{209}.$$

The *relative fraction of time when two philosophers dine compared with when only one philosopher dines* is

$$\tfrac{80}{209} \cdot \tfrac{209}{100} = \tfrac{4}{5}.$$

— The action of a philosopher beginning to eat, $\{b\}$, is only possible from the states $\mathcal{K}_2, \mathcal{K}_3$. In each of these states, the probability of beginning to eat is the sum of the execution probabilities for all multisets of multiactions containing $\{b\}$. Thus, the *steady-state probability of a philosopher beginning to eat* is

$$\psi'_2 \sum_{\{A,\widetilde{\mathcal{K}} \mid \{b\} \in A,\ \mathcal{K}_2 \stackrel{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\mathcal{K}_2, \widetilde{\mathcal{K}}) \ + \ \psi'_3 \sum_{\{A,\widetilde{\mathcal{K}} \mid \{b\} \in A,\ \mathcal{K}_3 \stackrel{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\mathcal{K}_3, \widetilde{\mathcal{K}})$$

$$= \tfrac{29}{209}\left(\tfrac{15}{29} + \tfrac{5}{29}\right) + \tfrac{100}{209}\left(\tfrac{3}{10} + \tfrac{1}{10}\right)$$
$$= \tfrac{60}{209}.$$

Note that the performance indices are the same for the complete and quotient abstract dining philosophers systems. The coincidence of both the first performance index and the second group of indices clearly illustrates the result of Proposition 7.1. The coincidence of the third performance index is due to Theorem 7.3: just apply the theorem to the derived step traces $\{\{b\}\}$, $\{\{b\},\{b\}\}$, $\{\{b\},\{e\}\}$ of the expressions $\overline{F}$ and $\overline{F'}$, and then sum the left and right parts of the three resulting equalities.

### 8.3. *The generalised system*

We will now determine the effect of the multiaction probabilities from the specification of the dining philosophers system on its performance. We will assume that all these multiactions have the same probability $\rho$. The resulting specification $K$ of the generalised dining philosophers system is defined as follows.

The static expression for philosopher $i$ ($1 \leqslant i \leqslant 4$) is

$$K_i = [(\{x_i\},\rho) * ((((\{b_i,\widehat{y_i}\},\rho);(\{e_i,\widehat{z_i}\},\rho))[]((\{y_{i+1}\},\rho);(\{z_{i+1}\},\rho))) * \mathsf{Stop}].$$

And the static expression for philosopher 5 is

$$K_5 = [(\{a,\widehat{x_1},\widehat{x_2},\widehat{x_2},\widehat{x_4}\},\rho) * ((((\{b_5,\widehat{y_5}\},\rho);(\{e_5,\widehat{z_5}\},\rho))[]((\{y_1\},\rho);(\{z_1\},\rho))) * \mathsf{Stop}].$$

The static expression for the generalised dining philosophers system is

$$K = (K_1 \| K_2 \| K_3 \| K_4 \| K_5)\ \mathsf{sr}\ (x_1,x_2,x_3,x_4,y_1,y_2,y_3,y_4,y_5,z_1,z_2,z_3,z_4,z_5).$$

$DR(\overline{K})$ has 12 states, which are interpreted as follows:

$\tilde{s}_1$: the initial state
$\tilde{s}_2$: the system is activated and no philosophers are dining yet
$\tilde{s}_3$: philosopher 1 dines

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

$\tilde{s}_4$: philosophers 1 and 4 dine
$\tilde{s}_5$: philosophers 1 and 3 dine
$\tilde{s}_6$: philosopher 4 dines
$\tilde{s}_7$: philosopher 3 dines
$\tilde{s}_8$: philosophers 2 and 4 dine
$\tilde{s}_9$: philosophers 3 and 5 dine
$\tilde{s}_{10}$: philosopher 2 dines
$\tilde{s}_{11}$: philosopher 5 dines
$\tilde{s}_{12}$: philosophers 2 and 5 dine.

The average sojourn time vector of $\overline{K}$ is

$$\widetilde{SJ} = \left(\frac{1}{\rho^5}, \frac{1+3\rho^2+\rho^4}{5\rho^2}, \frac{1+\rho^2}{\rho^2(3-\rho^2)}, \frac{1}{\rho^2(2-\rho^2)}, \frac{1}{\rho^2(2-\rho^2)}, \frac{1+\rho^2}{\rho^2(3-\rho^2)}, \right.$$
$$\left. \frac{1+\rho^2}{\rho^2(3-\rho^2)}, \frac{1}{\rho^2(2-\rho^2)}, \frac{1}{\rho^2(2-\rho^2)}, \frac{1+\rho^2}{\rho^2(3-\rho^2)}, \frac{1+\rho^2}{\rho^2(3-\rho^2)}, \frac{1}{\rho^2(2-\rho^2)}\right).$$

The sojourn time variance vector of $\overline{K}$ is

$$\widetilde{VAR} = \left(\frac{1-\rho^5}{\rho^{10}}, \frac{(1-\rho^2)^2(1+3\rho^2+\rho^4)}{25\rho^4}, \frac{(1-\rho^2)^2(1+\rho^2)}{\rho^4(3-\rho^2)^2}, \frac{(1-\rho^2)^2}{\rho^4(2-\rho^2)^2}, \frac{(1-\rho^2)^2}{\rho^4(2-\rho^2)^2}, \frac{(1-\rho^2)^2(1+\rho^2)}{\rho^4(3-\rho^2)^2}, \right.$$
$$\left. \frac{(1-\rho^2)^2(1+\rho^2)}{\rho^4(3-\rho^2)^2}, \frac{(1-\rho^2)^2}{\rho^4(2-\rho^2)^2}, \frac{(1-\rho^2)^2}{\rho^4(2-\rho^2)^2}, \frac{(1-\rho^2)^2(1+\rho^2)}{\rho^4(3-\rho^2)^2}, \frac{(1-\rho^2)^2(1+\rho^2)}{\rho^4(3-\rho^2)^2}, \frac{(1-\rho^2)^2}{\rho^4(2-\rho^2)^2}\right).$$

With $\chi = 1-\rho^2$ and $\theta = 1+3\rho^2+\rho^4$, the TPM for $DTMC(\overline{K})$ is

$$\widetilde{\mathbf{P}} = \begin{pmatrix}
1-\rho^5 & \rho^5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{\chi^2}{\theta} & \frac{\rho^2\chi}{\theta} & \frac{\rho^4}{\theta} & \frac{\rho^4}{\theta} & \frac{\rho^2\chi}{\theta} & \frac{\rho^2\chi}{\theta} & \frac{\rho^4}{\theta} & \frac{\rho^4}{\theta} & \frac{\rho^2\chi}{\theta} & \frac{\rho^2\chi}{\theta} & \frac{\rho^4}{\theta} \\
0 & \frac{\rho^2\chi}{1+\rho^2} & \frac{\chi^2}{1+\rho^2} & \frac{\rho^2\chi}{1+\rho^2} & \frac{\rho^2\chi}{1+\rho^2} & \frac{\rho^4}{1+\rho^2} & \frac{\rho^4}{1+\rho^2} & 0 & 0 & 0 & 0 & 0 \\
0 & \rho^4 & \rho^2\chi & \chi^2 & 0 & \rho^2\chi & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \rho^4 & \rho^2\chi & 0 & \chi^2 & 0 & \rho^2\chi & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{\rho^2\chi}{1+\rho^2} & \frac{\rho^4}{1+\rho^2} & \frac{\rho^2\chi}{1+\rho^2} & 0 & \frac{\chi^2}{1+\rho^2} & 0 & \frac{\rho^2\chi}{1+\rho^2} & 0 & \frac{\rho^4}{1+\rho^2} & 0 & 0 \\
0 & \frac{\rho^2\chi}{1+\rho^2} & \frac{\rho^4}{1+\rho^2} & 0 & \frac{\rho^2\chi}{1+\rho^2} & 0 & \frac{\chi^2}{1+\rho^2} & 0 & \frac{\rho^2\chi}{1+\rho^2} & 0 & \frac{\rho^4}{1+\rho^2} & 0 \\
0 & \rho^4 & 0 & 0 & 0 & \rho^2\chi & 0 & \chi^2 & 0 & \rho^2\chi & 0 & 0 \\
0 & \rho^4 & 0 & 0 & 0 & 0 & \rho^2\chi & 0 & \chi^2 & 0 & \rho^2\chi & 0 \\
0 & \frac{\rho^2\chi}{1+\rho^2} & 0 & 0 & 0 & \frac{\rho^4}{1+\rho^2} & 0 & \frac{\rho^2\chi}{1+\rho^2} & 0 & \frac{\chi^2}{1+\rho^2} & \frac{\rho^4}{1+\rho^2} & \frac{\rho^2\chi}{1+\rho^2} \\
0 & \frac{\rho^2\chi}{1+\rho^2} & 0 & 0 & 0 & 0 & \frac{\rho^4}{1+\rho^2} & 0 & \frac{\rho^2\chi}{1+\rho^2} & \frac{\rho^4}{1+\rho^2} & \frac{\chi^2}{1+\rho^2} & \frac{\rho^2\chi}{1+\rho^2} \\
0 & \rho^4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \rho^2\chi & \rho^2\chi & \chi^2
\end{pmatrix}.$$

The steady-state PMF for $DTMC(\overline{K})$ is

$$\tilde{\psi} = \tfrac{1}{11+8\rho^2+\rho^4}(0, 1+3\rho^2+\rho^4, 1+\rho^2, 1, 1, 1+\rho^2, 1+\rho^2, 1, 1, 1+\rho^2, 1+\rho^2, 1).$$

We can now calculate the main performance indices as follows:

— The average recurrence time in the state $s_2$, where all the forks are available, which is called the *average system run-through*, is

$$\frac{1}{\tilde{\psi}_2} = \tfrac{11+8\rho^2+\rho^4}{1+3\rho^2+\rho^4}.$$

— Nobody eats in the state $s_2$, so the *fraction of time when no philosophers dine* is

$$\tilde{\psi}_2 = \tfrac{1+3\rho^2+\rho^4}{11+8\rho^2+\rho^4}.$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Only one philosopher eats in the states $s_3, s_6, s_7, s_{10}, s_{11}$, so the *fraction of time when only one philosopher dines* is

$$\begin{aligned}\tilde{\psi}_3 + \tilde{\psi}_6 + \tilde{\psi}_7 + \tilde{\psi}_{10} + \tilde{\psi}_{11} &= \tfrac{1+\rho^2}{11+8\rho^2+\rho^4} + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4} + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4} + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4} + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4} \\ &= \tfrac{5(1+\rho^2)}{11+8\rho^2+\rho^4}.\end{aligned}$$

Two philosophers eat together in the states $s_4, s_5, s_8, s_9, s_{12}$, so the *fraction of time when two philosophers dine* is

$$\begin{aligned}\tilde{\psi}_4 + \tilde{\psi}_5 + \tilde{\psi}_8 + \tilde{\psi}_9 + \tilde{\psi}_{12} &= \tfrac{1}{11+8\rho^2+\rho^4} + \tfrac{1}{11+8\rho^2+\rho^4} + \tfrac{1}{11+8\rho^2+\rho^4} + \tfrac{1}{11+8\rho^2+\rho^4} + \tfrac{1}{11+8\rho^2+\rho^4} \\ &= \tfrac{5}{11+8\rho^2+\rho^4}.\end{aligned}$$

The *relative fraction of time when two philosophers dine compared with when only one philosopher dines* is

$$\tfrac{5}{11+8\rho^2+\rho^4} \cdot \tfrac{11+8\rho^2+\rho^4}{5(1+\rho^2)} = \tfrac{1}{1+\rho^2}.$$

— The action of the first philosopher beginning to eat, $(\{b_1\}, \rho^2)$, is only possible from the states $s_2, s_6, s_7$. In each of these states, the probability of beginning to eat is the sum of the execution probabilities for all multisets of activities containing $(\{b_1\}, \rho^2)$. Thus, the *steady-state probability of the first philosopher beginning to eat* is

$$\begin{aligned}&\tilde{\psi}_2 \sum_{\{\Gamma|(\{b_1\},\rho^2)\in\Gamma\}} PT(\Gamma, s_2) + \tilde{\psi}_6 \sum_{\{\Gamma|(\{b_1\},\rho^2)\in\Gamma\}} PT(\Gamma, s_6) + \tilde{\psi}_7 \sum_{\{\Gamma|(\{b_1\},\rho^2)\in\Gamma\}} PT(\Gamma, s_7) \\ &= \tfrac{1+3\rho^2+\rho^4}{11+8\rho^2+\rho^4}\left(\tfrac{\rho^2(1-\rho^2)}{1+3\rho^2+\rho^4} + \tfrac{\rho^4}{1+3\rho^2+\rho^4} + \tfrac{\rho^4}{1+3\rho^2+\rho^4}\right) \\ &\quad + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4}\left(\tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2}\right) \\ &\quad + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4}\left(\tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2}\right) \\ &= \tfrac{\rho^2(3+\rho^2)}{11+8\rho^2+\rho^4}.\end{aligned}$$

## 8.4. *The abstract generalised system and its reduction*

In this section, we consider a modification of the generalised dining philosophers system with abstraction from personalities. We call this system the abstract generalised dining philosophers system.

The static expression for philosopher $i$ $(1 \leqslant i \leqslant 4)$ is

$$L_i = [(\{x_i\}, \rho) * ((((\{b, \widehat{y_i}\}, \rho); (\{e, \widehat{z_i}\}, \rho))[]\ (((\{y_{i+1}\}, \rho); (\{z_{i+1}\}, \rho))) * \mathsf{Stop}].$$

And the static expression for philosopher 5 is

$$L_5 = [(\{a, \widehat{x_1}, \widehat{x_2}, \widehat{x_2}, \widehat{x_4}\}, \rho) * ((((\{b, \widehat{y_5}\}, \rho); (\{e, \widehat{z_5}\}, \rho))[](((\{y_1\}, \rho); (\{z_1\}, \rho))) * \mathsf{Stop}].$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

The static expression for the abstract generalised dining philosophers system is

$$L = (L_1 \| L_2 \| L_3 \| L_4 \| L_5) \text{ sr } (x_1, x_2, x_3, x_4, y_1, y_2, y_3, y_4, y_5, z_1, z_2, z_3, z_4, z_5).$$

$DR(\overline{L})$ resembles $DR(\overline{K})$ and $TS(\overline{L})$ is similar to $TS(\overline{K})$, and

$$DTMC(\overline{L}) = DTMC(\overline{K}).$$

Thus, the TPM and the steady-state PMF for $DTMC(\overline{L})$ and $DTMC(\overline{K})$ coincide.

The first performance index and the second group of indices are the same for the standard and abstract generalised systems. The following performance index is based on the depersonalised viewpoint:

— The action of a philosopher beginning to eat, $(\{b\}, \rho^2)$, is only possible from the states $\tilde{s}_2, \tilde{s}_3, \tilde{s}_6, \tilde{s}_7, \tilde{s}_{10}, \tilde{s}_{11}$. In each of these states, the probability of beginning to eat is the sum of the execution probabilities for all multisets of activities containing $(\{b\}, \rho^2)$. Thus, the *steady-state probability of a philosopher beginning to eat* is

$$\begin{aligned}
&\tilde{\psi}_2 \sum_{\{\Gamma|(\{b\},\rho^2)\in\Gamma\}} PT(\Gamma, \tilde{s}_2) + \tilde{\psi}_3 \sum_{\{\Gamma|(\{b\},\rho^2)\in\Gamma\}} PT(\Gamma, \tilde{s}_3) \\
&+ \tilde{\psi}_6 \sum_{\{\Gamma|(\{b\},\rho^2)\in\Gamma\}} PT(\Gamma, \tilde{s}_6) + \tilde{\psi}_7 \sum_{\{\Gamma|(\{b\},\rho^2)\in\Gamma\}} PT(\Gamma, \tilde{s}_7) \\
&+ \tilde{\psi}_{10} \sum_{\{\Gamma|(\{b\},\rho^2)\in\Gamma\}} PT(\Gamma, \tilde{s}_{10}) + \tilde{\psi}_{11} \sum_{\{\Gamma|(\{b\},\rho^2)\in\Gamma\}} PT(\Gamma, \tilde{s}_{11}) \\
&= \tfrac{1+3\rho^2+\rho^4}{11+8\rho^2+\rho^4} \left( \tfrac{\rho^2(1-\rho^2)}{1+3\rho^2+\rho^4} + \tfrac{\rho^4}{1+3\rho^2+\rho^4} + \tfrac{\rho^2(1-\rho^2)}{1+3\rho^2+\rho^4} + \tfrac{\rho^4}{1+3\rho^2+\rho^4} \right. \\
&\qquad + \tfrac{\rho^2(1-\rho^2)}{1+3\rho^2+\rho^4} + \tfrac{\rho^4}{1+3\rho^2+\rho^4} + \tfrac{\rho^2(1-\rho^2)}{1+3\rho^2+\rho^4} + \tfrac{\rho^4}{1+3\rho^2+\rho^4} \\
&\qquad \left. + \tfrac{\rho^2(1-\rho^2)}{1+3\rho^2+\rho^4} + \tfrac{\rho^4}{1+3\rho^2+\rho^4} \right) \\
&\quad + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4} \left( \tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2} + \tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2} \right) \\
&\quad + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4} \left( \tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2} + \tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2} \right) \\
&\quad + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4} \left( \tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2} + \tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2} \right) \\
&\quad + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4} \left( \tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2} + \tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2} \right) \\
&\quad + \tfrac{1+\rho^2}{11+8\rho^2+\rho^4} \left( \tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2} + \tfrac{\rho^2(1-\rho^2)}{1+\rho^2} + \tfrac{\rho^4}{1+\rho^2} \right) \\
&= \tfrac{15\rho^2}{11+8\rho^2+\rho^4}.
\end{aligned}$$

We have

$$DR(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_1, \widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_4\},$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

where

$\widetilde{\mathcal{K}}_1 = \{\tilde{s}_1\}$ (the initial state)

$\widetilde{\mathcal{K}}_2 = \{\tilde{s}_2\}$ (the system is activated and no philosophers dine)

$\widetilde{\mathcal{K}}_3 = \{\tilde{s}_3, \tilde{s}_6, \tilde{s}_7, \tilde{s}_{10}, \tilde{s}_{11}\}$ (one philosopher dines)

$\widetilde{\mathcal{K}}_4 = \{\tilde{s}_4, \tilde{s}_5, \tilde{s}_8, \tilde{s}_9, \tilde{s}_{12}\}$ (two philosophers dine).

The quotient average sojourn time vector of $\overline{L}$ is

$$\widetilde{SJ}' = \left(\tfrac{1}{\rho^5}, \tfrac{1+3\rho^2+\rho^4}{5\rho^2}, \tfrac{1+\rho^2}{\rho^2(3-\rho^2)}, \tfrac{1}{\rho^2(2-\rho^2)}\right).$$

The quotient sojourn time variance vector of $\overline{L}$ is

$$\widetilde{VAR}' = \left(\tfrac{1-\rho^5}{\rho^{10}}, \tfrac{(1-\rho^2)^2(1+3\rho^2+\rho^4)}{25\rho^4}, \tfrac{(1-\rho^2)^2(1+\rho^2)}{\rho^4(3-\rho^2)^2}, \tfrac{(1-\rho^2)^2}{\rho^4(2-\rho^2)^2}\right).$$

The TPM for $DTMC_{\underleftrightarrow{}_{ss}}(\overline{L})$ is

$$\widetilde{\mathbf{P}}' = \begin{pmatrix} 1-\rho^5 & \rho^5 & 0 & 0 \\ 0 & \frac{(1-\rho^2)^2}{1+3\rho^2+\rho^4} & \frac{5\rho^2(1-\rho^2)}{1+3\rho^2+\rho^4} & \frac{5\rho^4}{1+3\rho^2+\rho^4} \\ 0 & \frac{\rho^2(1-\rho^2)}{1+\rho^2} & \frac{1-2\rho^2+3\rho^4}{1+\rho^2} & \frac{2\rho^2(1-\rho^2)}{1+\rho^2} \\ 0 & \rho^4 & 2\rho^2(1-\rho^2) & (1-\rho^2)^2 \end{pmatrix}.$$

The steady-state PMF for $DTMC_{\underleftrightarrow{}_{ss}}(\overline{L})$ is

$$\tilde{\psi}' = \left(0, \tfrac{1+3\rho^2+\rho^4}{11+8\rho^2+\rho^4}, \tfrac{5(1+\rho^2)}{11+8\rho^2+\rho^4}, \tfrac{5}{11+8\rho^2+\rho^4}\right).$$

We can now calculate the main performance indices as follows:

— The average recurrence time in the state $\widetilde{\mathcal{K}}_2$, where all the forks are available, which is called the *average system run-through*, is

$$\frac{1}{\tilde{\psi}'_2} = \tfrac{11+8\rho^2+\rho^4}{1+3\rho^2+\rho^4}.$$

— Nobody eats in the state $\widetilde{\mathcal{K}}_2$, so the *fraction of time when no philosophers dine* is

$$\tilde{\psi}'_2 = \tfrac{1+3\rho^2+\rho^4}{11+8\rho^2+\rho^4}.$$

Only one philosopher eats in the state $\widetilde{\mathcal{K}}_3$, so the *fraction of time when only one philosopher dines* is

$$\tilde{\psi}'_3 = \tfrac{5(1+\rho^2)}{11+8\rho^2+\rho^4}.$$

Two philosophers eat together in the state $\widetilde{\mathcal{K}}_4$, so the *fraction of time when two philosophers dine* is

$$\tilde{\psi}'_4 = \tfrac{5}{11+8\rho^2+\rho^4}.$$

The *relative fraction of time when two philosophers dine compared with when only one philosopher dines* is

$$\tfrac{5}{11+8\rho^2+\rho^4} \cdot \tfrac{11+8\rho^2+\rho^4}{5(1+\rho^2)} = \tfrac{1}{1+\rho^2}.$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

— The action of a philosopher beginning to eat, $\{b\}$, is only possible from the states $\widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_3$. In each of these states, the probability of beginning to eat is the sum of the execution probabilities for all multisets of multiactions containing $\{b\}$. Thus, the *steady-state probability of a philosopher beginning to eat* is

$$\tilde{\psi}'_2 \sum_{\{A,\widetilde{\mathcal{K}}|\{b\}\in A,\ \widetilde{\mathcal{K}}_2 \xrightarrow{A} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}) + \tilde{\psi}'_3 \sum_{\{A,\widetilde{\mathcal{K}}|\{b\}\in A,\ \widetilde{\mathcal{K}}_3 \xrightarrow{A} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}})$$

$$= \frac{1+3\rho^2+\rho^4}{11+8\rho^2+\rho^4}\left(\frac{5\rho^2(1-\rho^2)}{1+3\rho^2+\rho^4} + \frac{5\rho^4}{1+3\rho^2+\rho^4}\right) + \frac{5(1+\rho^2)}{11+8\rho^2+\rho^4}\left(\frac{2\rho^2(1-\rho^2)}{1+\rho^2} + \frac{2\rho^4}{1+\rho^2}\right) = \frac{15\rho^4}{11+8\rho^2+\rho^4}.$$

Note again that the performance indices are the same for the complete and quotient abstract generalised dining philosophers systems. The explanation of this fact is the same as for the complete and quotient abstract dining philosophers systems presented earlier.

We will now consider how changing the value of the parameter $\rho$ will affect the performance of the quotient abstract generalised dining philosophers system in its steady state. Remember that $\rho \in (0;1)$ is the probability for every multiaction of the system. The closer $\rho$ is to 0, the smaller is the probability of executing some activity at each discrete time step, hence, the system will most probably *stand idle*. The closer $\rho$ is to 1, the greater the probability of executing some activity at each discrete time step, hence, the system will most probably *operate*.

Since $\tilde{\psi}'_1 = 0$, only

$$\tilde{\psi}'_2 = \frac{1+3\rho^2+\rho^4}{11+8\rho^2+\rho^4}$$
$$\tilde{\psi}'_3 = \frac{5(1+\rho^2)}{11+8\rho^2+\rho^4}$$
$$\tilde{\psi}'_4 = \frac{5}{11+8\rho^2+\rho^4}$$

depend on $\rho$. Figure 16 shows graphs of $\tilde{\psi}'_2$, $\tilde{\psi}'_3$, $\tilde{\psi}'_4$ as functions of $\rho$. The graphs for $\tilde{\psi}'_2$ and $\tilde{\psi}'_4$ are symmetric around the constant probability $\frac{1}{4}$. It can be seen that the larger $\rho$ is, the smaller is the difference between $\tilde{\psi}'_2$ and $\tilde{\psi}'_4$ and the greater the difference between $\tilde{\psi}'_3$ and $\tilde{\psi}'_4$. Notice, however, that we do not allow $\rho = 0$ or $\rho = 1$.

Figure 17 shows the graph of the average system run-through, calculated as $\frac{1}{\tilde{\psi}'_2}$, as a function of $\rho$. It can be seen that the run-through tends to 11 when $\rho$ approaches 0, and tends to 4 when $\rho$ approaches 1. So, to speed up the operation of the system, we should take the parameter $\rho$ closer to 1.

The fraction of time when no philosophers dine, calculated as $\tilde{\psi}'_2$, tends to $\frac{1}{11}$ when $\rho$ approaches 0, and tends to $\frac{1}{4}$ when $\rho$ approaches 1. The fraction of time when only one philosopher dines, calculated as $\tilde{\psi}'_3$, tends to $\frac{5}{11}$ when $\rho$ approaches 0, and tends to $\frac{1}{2}$ when $\rho$ approaches 1. The fraction of time when two philosophers dine, calculated as $\tilde{\psi}'_4$, tends to $\frac{5}{11}$ when $\rho$ approaches 0, and tends to $\frac{1}{4}$ when $\rho$ approaches 1.

The first graph in Figure 18 represents the relative fraction of time when two philosophers dine compared with when only one philosopher dines, calculated as $\tilde{\psi}'_4/\tilde{\psi}'_3$,

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

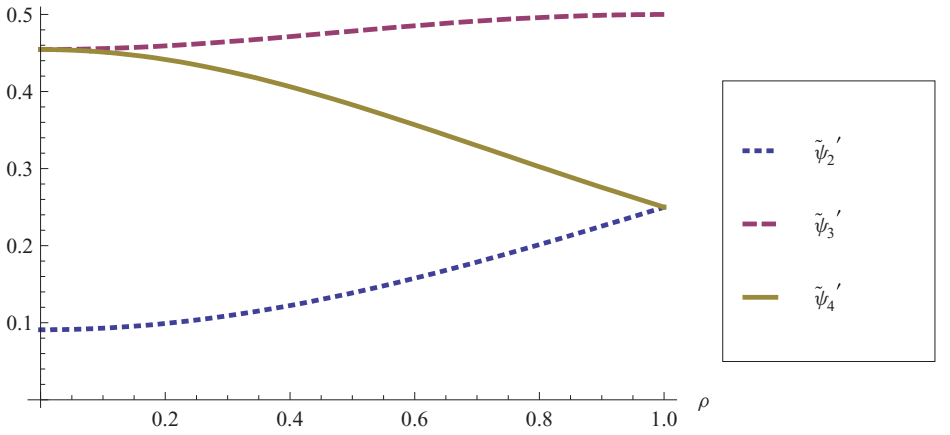



Fig. 16. (Colour online) Steady-state probabilities $\tilde{\psi}'_2$, $\tilde{\psi}'_3$, $\tilde{\psi}'_4$ as functions of the parameter $\rho$.

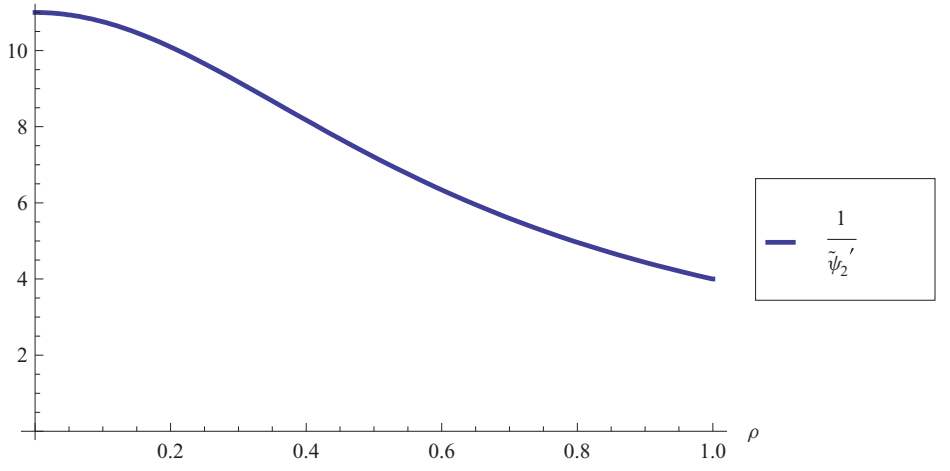



Fig. 17. (Colour online) Average system run-through $\frac{1}{\tilde{\psi}'_2}$ as a function of the parameter $\rho$.

as a function of $\rho$. It can be seen that the relative fraction tends to 1 when $\rho$ approaches 0, and tends to $\frac{1}{2}$ when $\rho$ approaches 1. So, to increase this relative fraction, we should take the parameter $\rho$ closer to 0.

The second graph in Figure 18 represents the steady-state probability of a philosopher beginning to eat, calculated as

$$\tilde{\psi}'_2\widetilde{\Sigma}'_2 + \tilde{\psi}'_3\widetilde{\Sigma}'_3,$$

where

$$\widetilde{\Sigma}'_i = \sum_{\{A,\widetilde{\mathcal{K}}|\{b\}\in A,\ \widetilde{\mathcal{K}}_i \overset{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_i, \widetilde{\mathcal{K}}),\ i \in \{2,3\},$$

as a function of $\rho$. It can be seen that the probability tends to 0 when $\rho$ approaches 0 and tends to $\frac{3}{4}$ when $\rho$ approaches 1. To increase this probability, we should take the parameter $\rho$ closer to 1.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

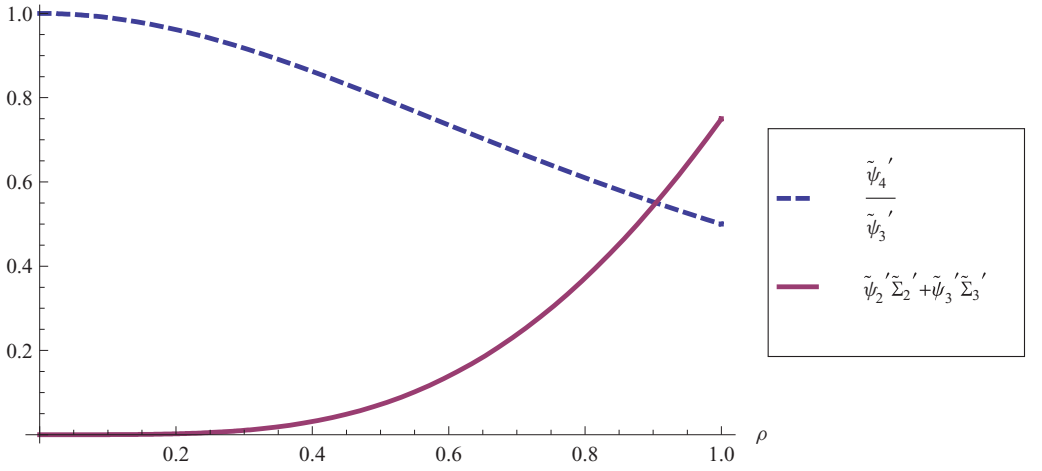



Fig. 18. (Colour online) Some performance indices as functions of the parameter $\rho$.

Since

$$\tilde{\psi}_4' - \tilde{\psi}_2' = \frac{4-3\rho^2-\rho^4}{11+8\rho^2+\rho^4},$$

the difference tends to $\frac{4}{11}$ when $\rho$ approaches 0, and tends to 0 when $\rho$ approaches 1. This difference can be treated as the difference between the fractions of time when two and when no philosophers dine. Since

$$\tilde{\psi}_3' - \tilde{\psi}_4' = \frac{5\rho^2}{11+8\rho^2+\rho^4},$$

the difference tends to 0 when $\rho$ approaches 0, and tends to $\frac{1}{4}$ when $\rho$ approaches 1. This difference can be treated as the difference between the fractions of time when one and when two philosophers dine.

From the performance viewpoint, the expression

$$\tilde{\psi}_3' + \tilde{\psi}_4' - \tilde{\psi}_2' = \frac{9+2\rho^2-\rho^4}{11+8\rho^2+\rho^4}$$

is more interesting. Figure 19 shows the graph of $\tilde{\psi}_3'+\tilde{\psi}_4'-\tilde{\psi}_2'$ as a function of $\rho$. The value of the expression tends to $\frac{9}{11}$ when $\rho$ approaches 0, and tends to $\frac{1}{2}$ when $\rho$ approaches 1. This value can be interpreted as the difference between the fractions of time when some (one or two) and when no philosophers dine.

Thus, when $\rho$ is closer to 0, more time is spent eating and less time remains for thinking, that is, *dining is preferred*. In this case, the dining time fractions for one and two philosophers approach the same value $\frac{5}{11}$ (the relative time fraction tends to 1). When $\rho$ is closer to 1, the situation is symmetric, that is, *thinking is preferred*. In this case, the dining time fraction for one philosopher approaches its maximum $\frac{1}{2}$, whereas the dining time fraction for two philosophers approaches its minimum $\frac{1}{4}$ (the relative time fraction tends to $\frac{1}{2}$).

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

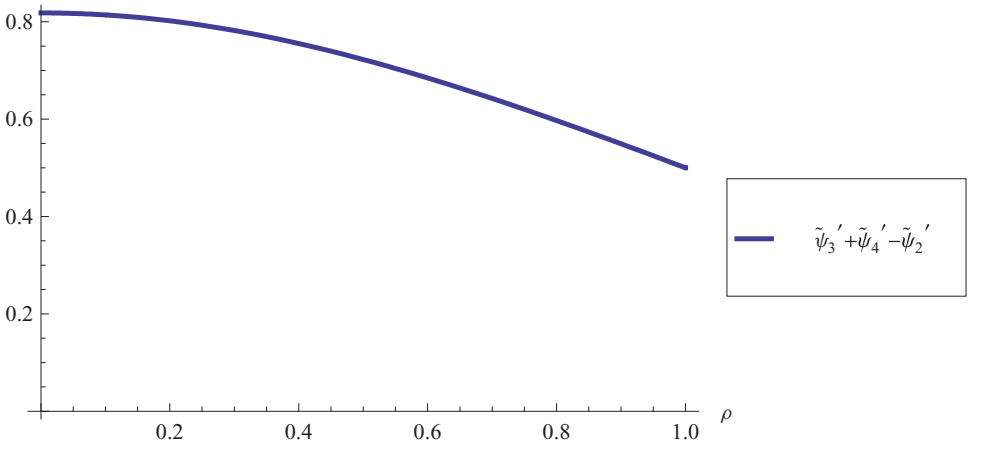



Fig. 19. (Colour online) Expression $\tilde{\psi}_3' + \tilde{\psi}_4' - \tilde{\psi}_2'$ as a function of the parameter $\rho$.

## 9. Related work

In this section, we give detailed consideration to the differences and similarities between dtsPBC and other well-known or similar SPAs so that we can subsequently determine the specific advantages of dtsPBC.

### 9.1. *Continuous time and interleaving semantics*

We will begin by comparing dtsPBC with classical SPAs, specifcally, Markovian Timed Processes for Performance Evaluation (MTIPP) (Hermanns and Rettelbach 1994), Performance Evaluation Process Algebra (PEPA) (Hillston 1996) and Extended Markovian Process Algebra (EMPA) (Bernardo and Gorrieri 1998).

In MTIPP, every activity is a pair consisting of the action name (including the symbol $\tau$ for the *internal*, invisible action) and a parameter specifying the exponential distribution of the activity duration (the *rate*). The operations are *prefix*, *choice*, *parallel* composition, including *synchronisation* on the specified action set, and *recursion*. It is also possible to specify processes using recursive equations. The interleaving semantics is defined on the basis of Markovian (that is, they are extended by the specification of rates) labelled transition systems. Note that we have interleaving behaviour here because the exponential probability distribution function is continuous, and according to the properties of continuous distributions, a simultaneous firing of any two activities has zero probability. Continuous time Markov chains (CTMCs) can be derived from the transition systems to analyse the performance.

In PEPA, activities are pairs consisting of action types (including the *unknown*, unimportant type $\tau$) and activity rates. The rate is either a parameter specifying the exponential distribution of the activity duration or it is *unspecified*, and denoted by $\top$. An activity with an unspecified rate is *passive* by its action type. The set of operations includes *prefix*, *choice*, *cooperation*, *hiding* and constants whose meaning is given by the defining equations, including *recursive* ones. Cooperation is accomplished on the set of action

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

types (the cooperation set) on which the components must *synchronise*, or cooperate. If the cooperation set is empty, the cooperation operator becomes the *parallel* combinator. The semantics is interleaving, and is defined using an extension of labelled transition systems that include the possibility of specifying activity rates. Continuous time Markov processes (CTMPs) are generated from the transition systems, and can be used for performance evaluation with the help of the embedded continuous time Markov chains (ECTMCs).

In EMPA, each action is a pair consisting of its type and rate. Actions can be *external* or *internal* (denoted by $\tau$) according to types. There are three kinds of actions according to rates: *timed* actions, with exponentially distributed durations (these are essentially the same as the MTIPP and PEPA actions); *immediate* actions, with priorities and weights (these are analogous to the immediate transitions of generalised SPNs, GSPNs); and *passive* actions (these are similar to PEPA's passive actions). Timed actions specify activities that are relevant for performance analysis. Immediate actions model logical events and activities that are either irrelevant from the performance viewpoint or much faster than the others. Passive actions model activities waiting for synchronisation with timed or immediate actions, and express non-deterministic choice. The operators are *prefix*, functional *abstraction*, functional *relabelling*, *alternative* composition and *parallel* composition. Parallel composition includes *synchronisation* on the set of action types, as in TCSP. The syntax also includes *recursive* definitions given by means of constants. The semantics is interleaving and based on labelled transition systems enriched with information on action rates. For the exponentially timed kernel of the algebra (the sublanguage restricted to exponentially timed and passive actions), it is possible to construct CTMCs from the transition systems of the process terms to analyse the performance.

In dtsPBC, every activity is a pair consisting of a multiaction (not just an action, as in classical SPAs) and its probability (not the rate, as in classical SPAs) to be executed independently. dtsPBC has a sequence operation in contrast to the prefix operation in classical SPAs. Arbitrary expressions can be combined using the sequence operator, so it is more flexible than the prefix operator, whose first argument is only a single activity.

The choice operation in dtsPBC is similar to that in MTIPP and PEPA, and to the alternative composition in EMPA, in the sense that the choice is probabilistic. However, a discrete probability function is used in dtsPBC, unlike the continuous ones in classical calculi.

Concurrency and synchronisation in dtsPBC are different operations (this feature is inherited from PBC), unlike in classical SPAs, where parallel composition (combinator) has a synchronisation capability. Relabelling in dtsPBC is analogous to that in EMPA, but it is extended to conjugated actions. The restriction operation in dtsPBC differs from hiding in PEPA and functional abstraction in EMPA, where the hidden actions are labelled with the 'silent' action symbol $\tau$. In dtsPBC, restriction by an action means that, for a given expression, any process behaviour containing the action or its conjugate is forbidden. Synchronisation on an elementary action in dtsPBC collects all the pairs consisting of this elementary action and its conjugate that are contained in the multiactions from the synchronised activities. The operation produces new activities such that the first element

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

of every resulting activity is the union of the multiactions from which all these pairs of conjugated actions are removed, and the second element is the product of the probabilities of the activities involved in the synchronisation. This differs from the way synchronisation is carried out in classical SPAs, where it is accomplished over identical action names, and every resulting activity consists of the same action name and the rate calculated using some expression (including sums, minimums and products) on the rates of the initial activities, such as the apparent rate in PEPA.

dtsPBC has no recursion operation or recursive definitions, but it includes the iteration operation to specify infinite looping behaviour with an explicit start and termination. dtsPBC has a discrete time semantics, and time delays in the states are geometrically distributed, unlike classical SPAs with their continuous time semantics and exponentially distributed activity delays. As a consequence, the semantics of dtsPBC is a step semantics, in contrast to the interleaving semantics of classical SPAs.

The performance can be investigated using the discrete time Markov chain (DTMC) extracted from the labelled probabilistic transition system associated with each dtsPBC expression. In classical SPAs, continuous time Markov chains (CTMCs) are used for performance evaluation. Bernardo *et al.* (1998) defined a denotational semantics for EMPA based on GSPNs, from which one can also extract the underlying SMCs and CTMCs (when both immediate and timed transitions are present) or discrete time Markov chains (DTMCs) (when there are only immediate transitions). dtsPBC has a denotational semantics in terms of LDTSPNs, from which the corresponding DTMCs can also be derived.

### 9.2. *Continuous time and non-interleaving semantics*

Amongst the various non-Markovian SPAs proposed (Katoen and D'Argenio 2001), only a few are non-interleaving. The semantics of all Markovian calculi is interleaving, and their action delays all have an exponential distribution, which is the only continuous probability distribution with the memoryless (Markovian) property. Brinksma *et al.* (1995) introduced Generalised Stochastic Process Algebra (GSPA), which has a true-concurrent denotational semantics in terms of generalised stochastic event structures (GSESs) with non-Markovian stochastic delays of events. However, they did not present an operational semantics or performance evaluation methods for GSPA. Later, Katoen *et al.* (1996) extracted generalised semi-Markov processes (GSMPs) from GSESs to analyse performance. Priami (1996) defined Stochastic $\pi$-calculus (S$\pi$) with general continuous distributions for activity delays. This has a proved operational semantics with transitions labelled by encodings of their deduction trees, but no well-established underlying performance model has been described for this version of S$\pi$. Bravetti *et al.* (1998) developed Generalised Semi-Markovian Process Algebra (GSMPA) with an ST-operational semantics and non-Markovian action delays, and for which performance analysis is accomplished using GSMPs.

Again, the first fundamental difference we find between dtsPBC and the GSPA, S$\pi$ and GSMPA calculi is that dtsPBC is based on PBC, whereas GSPA is an extension of Process Algebra (PA) (Brinksma *et al.* 1995), S$\pi$ extends $\pi$-calculus (Milner *et al.* 1992)

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

and GSMPA is an enrichment of EMPA. Thus, both GSPA and GSMPA have prefixing, choice (alternative composition), parallel composition, renaming (relabelling) and hiding (abstraction) operations, but only GSMPA permits constants. Unlike dtsPBC, GSPA has neither iteration nor recursion, and GSMPA allows only recursive definitions. On the other hand, S$\pi$ has additional operations to specify mobility.

The second significant difference is that geometrically distributed delays are associated with process states in dtsPBC, unlike the generally distributed delays assigned to events in GSPA or to activities in S$\pi$ and GSMPA. As a consequence, dtsPBC has a discrete time operational semantics allowing the concurrent execution of activities in steps. GSPA has no operational semantics, while S$\pi$ and GSMPA have continuous time semantics. In continuous time semantics, concurrency is simulated by interleaving, since the properties of continuous probability distributions mean that the simultaneous occurrence of two events has zero probability. Therefore, interleaving transitions are often annotated with additional information holding concurrency data. The transition labels in the operational semantics of S$\pi$ encode the action causality information and allow the derivation of the enabling relations and firing distributions of the concurrent transitions from the transition sequences. At the same time, abstracting from stochastic delays leads to the classical early interleaving semantics of $\pi$-calculus. The ST-operational semantics of GSMPA is based on decorated transition systems governed by transition rules with rather complex preconditions. There are two types of transitions: choice (action beginning) and termination (action ending) transitions. The choice transitions are labelled by weights of single actions chosen for execution while the termination transitions have no labels. Only single actions can begin, but several actions can end in parallel. Thus, the choice transitions are interleaving transitions while the termination transitions are step transitions. As a result, we obtain decorated interleaving/step transition systems. dtsPBC has an SPNs-based denotational semantics. Compared with event structures, PNs are more expressive and form a visually tractable formalism supporting finite specifications of infinite behaviour. Recursion in GSPA produces infinite GSESs, while dtsPBC has the iteration operation with a finite SPN semantics. The identification of infinite GSESs that can be finitely represented in GSPA is left for a future research.

### 9.3. *Discrete time*

van der Aalst *et al.* (2000) proposed a class of compositional DTSPNs, called dts-nets, that have generally distributed discrete time transition delays. The denotational semantics of a stochastic extension of (a subset of) Algebra of Communicating Processes (ACP) (Bergstra and Klop 1985) can be constructed using dts-nets. There are two types of transitions in dts-nets: immediate (timeless) transitions with zero delays and time transitions, whose delays are random values having general discrete time distributions. The top-down synthesis of dts-nets involves the substitution of their transitions by blocks (dts-subnets) corresponding to the sequence, choice, parallelism and iteration operators. van der Aalst *et al.* (2000) explained how to calculate the throughput time of dts-nets using the service time (defined as holding time or delay) of their transitions. To do this, they defined the notions of service distribution for the transitions and throughput distribution

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

for the building blocks. Since the throughput time of the parallelism block was calculated as the maximal service time for its two constituting transitions, the analogue of the step semantics approach was implemented.

Markovski and de Vink (2008) and Markovski and de Vink (2009) introduced an SPA called the Theory of Communicating Processes that has discrete stochastic time ($TCP^{dts}$). Its actions have (deterministic) discrete real-time delays (including zero time delays) or stochastic time delays. The algebra generalises real-time processes to discrete stochastic time ones by applying real-time properties to stochastic time and imposing a race condition on the real-time semantics. $TCP^{dts}$ has an interleaving operational semantics in terms of stochastic transition systems. The performance is analysed using discrete time probabilistic reward graphs, which are, essentially, the reward transition systems, with probabilistic states having a finite number of outgoing probabilistic transitions and timed states having a single outgoing timed transition. These graphs can be transformed by unfolding or geometrisation into discrete time Markov reward chains (DTMRCs), which are appropriate for transient or long-run (stationary) analysis.

The first difference between dtsPBC and the stochastic ACP and $TCP^{dts}$ algebras is that dtsPBC is based on PBC, while stochastic ACP and $TCP^{dts}$ are extensions of ACP. Stochastic ACP has only taken sequence, choice, parallelism and iteration operations from ACP, but dtsPBC also has relabelling, restriction and synchronisation, which are inherited from PBC. In $TCP^{dts}$, in addition to standard action prefixing, alternative, parallel composition, encapsulation (similar to restriction) and recursive variables, there are also timed delay prefixing, dependent delays scope and maximal time progress operators, which are new for both ACP and dtsPBC.

The second difference is that geometrically distributed delays are associated with process states in dtsPBC, unlike the zero or generally distributed discrete time delays of actions in stochastic ACP and the deterministic or generally distributed stochastic delays of actions in $TCP^{dts}$.

No formal syntax or operational semantics is defined for stochastic ACP, and there is no explanation of how to derive Markov chains from the algebraic expressions or the corresponding dts-nets to analyse performance. Neither is there an explicit statement of which type of semantics (interleaving or step) is accommodated in stochastic ACP. In spite of the discrete time approach, the operational semantics of $TCP^{dts}$ is still interleaving, unlike that of dtsPBC. In addition, no denotational semantics has been defined for $TCP^{dts}$.

Table 6 summarises the comparison of SPAs given in this section and in Section 1 by classifying the SPAs according to the concept of time they use and the type of operational semantics. The names of the SPAs with a denotational semantics based on SPNs are printed in bold, and the underlying stochastic process (if defined) is shown in parentheses near the name of the corresponding SPA.

## 10. Discussion

In this section we will discuss what advantages dtsPBC has over the SPAs described in Section 9.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Table 6. *Classification of stochastic process algebras.*

| Time | Interleaving semantics | Non-interleaving semantics |
|---|---|---|
| Continuous | MTIPP (CTMC), PEPA (CTMP), **EMPA** (SMC, CTMC), **sPBC** (CTMC), **gsPBC** (SMC) | GSPA (GSMP), S$\pi$, GSMPA (GSMP) |
| Discrete | $TCP^{dts}$ (DTMRC) | **stochastic ACP**, **dtsPBC** (DTMC), **dtsiPBC** (SMC, DTMC) |

An important aspect is the analytical tractability of the underlying stochastic process, which is used for performance analysis within SPAs. The underlying CTMCs in MTIPP and PEPA, as well as the SMCs in EMPA, are treated analytically, but these continuous time SPAs just have an interleaving semantics. GSPA, S$\pi$ and GSMPA are continuous time models, for which a non-interleaving semantics is constructed, but only simulation and numerical methods have been applied for the underlying GSMPs in GSPA and GSMPA, and no performance model has been defined for S$\pi$. Stochastic ACP and $TCP^{dts}$ are discrete time models with associated analytical methods for the throughput calculation in stochastic ACP and for performance evaluation based on the underlying DTMRCs in $TCP^{dts}$. However, both models only have an interleaving semantics. dtsPBC is a discrete time model with a non-interleaving semantics, in which analytical methods are applied to the underlying DTMCs. Hence, if an interleaving model is appropriate as a framework for the analytical solution of performance evaluation, there is a choice between the continuous time SPAs MTIPP, PEPA, EMPA and the discrete time stochastic ACP and $TCP^{dts}$. Otherwise, if a non-interleaving model is required, with the associated analytical methods for performance evaluation, and the discrete time approach is feasible, then dtsPBC is the right choice.

From the point of view of applications, it is important to consider what kind of system is more appropriate for modelling and analysing within each SPA. First, for interleaving continuous time systems, MTIPP and PEPA are well suited when the activity rates or the average sojourn time in the states are known in advance, and when an exponential distribution is a good approximation for the activity delay distributions, while EMPA can be used when the activity delays have a different duration order and for extended systems in which purely probabilistic choice or urgent activities must be implemented. GSPA and GSMPA are a good fit for modelling continuous time systems with the ability to hold the activity causality information, and with known activity delay distributions that cannot be approximated accurately by an exponential distribution, while S$\pi$ can additionally model mobility in such systems. $TCP^{dts}$ is a good choice for interleaving discrete time systems with deterministic (fixed) and generalised stochastic delays, while stochastic ACP can also be used to model non-interleaving systems, though it does not offer enough performance analysis methods. dtsPBC is suitable for step discrete time systems where the independent execution probabilities of activities are known and a geometrical distribution is a good approximation for the state residence time distributions.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Generally speaking, the stochastic process calculi proposed in the literature are based on interleaving, and parallelism is simulated by synchronous or asynchronous execution. As their semantic domain, the interleaving formalism of transition systems is often used. Thus, an investigation of stochastic extensions offering more expressive and powerful algebraic calculi is an important issue. The development of step or 'true concurrency' (such that parallelism is considered as a causal independence) SPAs is an interesting and non-trivial problem, which has attracted special attention in recent years. Nevertheless, only a few formal stochastic models have been defined with DTMCs acting as the underlying stochastic processes. As mentioned in Fourneau (2010), such models are more difficult to analyse since a lot of events can occur simultaneously in discrete time systems (the models have a step semantics), and the probability of a set of events cannot be easily related to the probability of the single events. As observed in Horváth *et al.* (2012), even for stochastic models with generally distributed time delays, some restrictions on the degree of concurrency are required to simplify the analysis techniques. In particular, the enabling restriction requires that no two generally distributed transitions are enabled in any reachable marking. This means that their activity periods cannot intersect, and that no two such transitions can fire simultaneously, which results in an interleaving semantics for the model.

Stochastic models with discrete time and step semantics have an important advantage over those that just have an interleaving semantics. This is because the underlying Markov chains of parallel stochastic processes have additional transitions corresponding to the simultaneous execution of concurrent (that is, non-synchronised) activities. This kind of transition allows us to bypass a lot of intermediate states, which should otherwise be visited when an interleaving semantics is used. When step semantics is used, the intermediate states may also be visited with some probability (this is an advantage since some alternative system's behaviour may start from these states), but this probability is not greater than the corresponding one for an interleaving semantics. While in an interleaving semantics only the empty or singleton (multi)sets of activities can be executed, generally speaking, in step semantics, the (multi)sets of activities with more than one element can also be executed. Hence, in step semantics, there are more execution variants from each state than in the interleaving case, and the execution probabilities, whose sum should be equal to 1, are distributed among more possibilities. Therefore, systems with parallel stochastic processes usually have smaller average run-through. When the underlying Markov chains of the processes are ergodic, they will take fewer discrete time units to stabilise the behaviour since their TPMs will be denser because of additional non-zero elements outside the main diagonal. Hence, both the first passage-time performance indices based on the transient probabilities and the steady-state performance indices based on the stationary probabilities can be computed more quickly, resulting in faster quantitative analysis of the systems. On the other hand, step semantics, induced by the simultaneous firing of several transitions at each step, is natural for Petri nets and allows us to exploit the full power of the model.

Thus, the main advantages of dtsPBC are the flexible multiaction labels and the set of powerful operations, as well as a step operational and a Petri net denotational semantics allowing the concurrent execution of activities (transitions), together with an ability to carry out analytical performance evaluation.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

## 11. Conclusions

In this paper, working in the context of dtsPBC with iteration, we have defined stochastic algebraic equivalences having natural net analogues on LDTSPNs. We have constructed a diagram of interrelations for the algebraic equivalences, and explained how we can reduce the transition systems and DTMCs, as well as the expressions and dts-boxes modulo the stochastic equivalences. We have investigated which of the proposed equivalences guarantee identity of the stationary behaviour, and proved that the weakest of the relations we have considered, *viz.* step stochastic bisimulation equivalence, has this property. We have also presented a case study based on the dining philosophers system as an example of modelling, performance evaluation and performance preserving reduction in the framework of the calculus.

Our framework has two advantages. First, it can be used to specify concurrent composition and synchronisation of (multi)actions, which is impossible with classical Markov chains. Second, the algebraic formulas represent processes in a more compact way than PNs and allow the application of syntactic transformations and comparisons. Process algebras are compositional by definition, and their operations naturally correspond to operators of programming languages. Hence, it is much easier to construct a complex model in the algebraic setting than in PNs. The complexity of the PNs generated for practical models in the literature demonstrates that constructing such PNs directly from the system specifications is not straightforward. dtsPBC is well suited to discrete time applications, such as business processes, neural and transportation networks, computer and communication systems, where discrete states change with a global time tick, as well as for applications in which the distributed architecture or the concurrency level should be preserved during modelling and analysis (remember that there are additional transitions in step semantics due to concurrent executions).

In the future, we plan to provide a logical characterisation for the stochastic equivalences using probabilistic modal logics. Our next research direction is to abstract from the silent activities in the definitions of the equivalences, that is, from the activities with an empty multiaction part. The main point here is that we should collect probabilities during such abstractions from an internal activity. Moreover, we plan to extend dtsPBC with recursion to enhance the specification power of the calculus.

## Appendix A. Proofs

### A.1. *Proof of Proposition 5.2*

**Proposition 5.2.** Let $G$ and $G'$ be dynamic expressions and $G \underline{\leftrightarrow}_{ss} G'$. Then $\mathcal{R}_{ss}(G, G')$ is the largest step stochastic bisimulation between $G$ and $G'$.

*Proof.* Using a similar approach to the proof of strong equivalence in Hillston (1996, Proposition 8.2.1), we shall prove the following fact about step stochastic bisimulation: if for all $j \in \mathcal{J}$, we have $\mathcal{R}_j : G \underline{\leftrightarrow}_{ss} G'$ for some index set $\mathcal{J}$, then the transitive closure of the union of all relations $\mathcal{R} = (\cup_{j \in \mathcal{J}} \mathcal{R}_j)^*$ is also an equivalence, and $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Since $\mathcal{R}_j$ is an equivalence for all $j \in \mathcal{J}$, it follows from the definition of $\mathcal{R}$ that $\mathcal{R}$ is also an equivalence.

Let $j \in \mathcal{J}$. Then, by the definition of $\mathcal{R}$, $(s_1, s_2) \in \mathcal{R}_j$ implies $(s_1, s_2) \in \mathcal{R}$. Hence, for all

$$\mathcal{H}_{jk} \in (DR(G) \cup DR(G'))/_{\mathcal{R}_j},$$

there exists

$$\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$$

such that $\mathcal{H}_{jk} \subseteq \mathcal{H}$. Moreover, there exists $\mathcal{J}'$ such that $\mathcal{H} = \cup_{k \in \mathcal{J}'} \mathcal{H}_{jk}$.

Using the denotation $\mathcal{R}(n) = (\cup_{j \in \mathcal{J}} \mathcal{R}_j)^n$ and letting $(s_1, s_2) \in \mathcal{R}$, by the definition of $\mathcal{R}$, there exists $n > 0$ such that $(s_1, s_2) \in \mathcal{R}(n)$. We shall now prove that $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$ by induction on $n$.

It is clear that for all $j \in \mathcal{J}$, the fact that $\mathcal{R}_j : G \underline{\leftrightarrow}_{ss} G'$ implies that for all $j \in \mathcal{J}$, we have $([G]_{\approx}, [G']_{\approx}) \in \mathcal{R}_j$, and we also have $([G]_{\approx}, [G']_{\approx}) \in \mathcal{R}$ by the definition of $\mathcal{R}$.

We still need to prove that $(s_1, s_2) \in \mathcal{R}$ implies that for all

$$\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$$

and all $A \in I\!N_f^{\mathcal{L}}$, we have

$$PM_A(s_1, \mathcal{H}) = PM_A(s_2, \mathcal{H}).$$

Proceeding with the induction, we have:

— $n = 1$:

In this case, $(s_1, s_2) \in \mathcal{R}$ implies that there exists $j \in \mathcal{J}$ such that $(s_1, s_2) \in \mathcal{R}_j$. Since $\mathcal{R}_j : G \underline{\leftrightarrow}_{ss} G'$, we get for all

$$\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$$

and all $A \in I\!N_f^{\mathcal{L}}$, that

$$\begin{aligned} PM_A(s_1, \mathcal{H}) &= \sum_{k \in \mathcal{J}'} PM_A(s_1, \mathcal{H}_{jk}) \\ &= \sum_{k \in \mathcal{J}'} PM_A(s_2, \mathcal{H}_{jk}) \\ &= PM_A(s_2, \mathcal{H}). \end{aligned}$$

— $n \to n + 1$:

Suppose that for all $m \leqslant n$ the fact that $(s_1, s_2) \in \mathcal{R}(m)$ implies that for all

$$\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$$

and all $A \in I\!N_f^{\mathcal{L}}$, we have

$$PM_A(s_1, \mathcal{H}) = PM_A(s_2, \mathcal{H}).$$

$(s_1, s_2) \in \mathcal{R}(n + 1)$ then implies that there exists $j \in \mathcal{J}$ such that $(s_1, s_2) \in \mathcal{R}_j \circ \mathcal{R}(n)$, that is, there exists

$$s_3 \in (DR(G) \cup DR(G'))$$

such that $(s_1, s_3) \in \mathcal{R}_j$ and $(s_3, s_2) \in \mathcal{R}(n)$.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Then, as in the $n = 1$ case, we get

$$PM_A(s_1, \mathcal{H}) = PM_A(s_3, \mathcal{H}).$$

By the induction hypothesis,

$$PM_A(s_1, \mathcal{H}) = PM_A(s_3, \mathcal{H}),$$

so for all

$$\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$$

and all $A \in \mathbb{N}_f^{\mathcal{L}}$, we have

$$PM_A(s_1, \mathcal{H}) = PM_A(s_3, \mathcal{H}) = PM_A(s_2, \mathcal{H}).$$

By definition, $\mathcal{R}_{ss}(G, G')$ is at least as large as the largest step stochastic bisimulation between $G$ and $G'$. It follows from the proof above that $\mathcal{R}_{ss}(G, G')$ is an equivalence and $\mathcal{R}_{ss}(G, G') : G \underline{\leftrightarrow}_{ss} G'$, and thus it is the largest step stochastic bisimulation between $G$ and $G'$. □

A.2. *Proof of Proposition 7.1*

**Proposition 7.1.** Let $G, G'$ be dynamic expressions with $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$. Let $\psi$ be the steady-state PMF for $DTMC(G)$ and $\psi'$ be the steady-state PMF for $DTMC(G')$. Then for all $\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$, we have

$$\sum_{s \in \mathcal{H} \cap DR(G)} \psi(s) = \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'(s').$$

*Proof.* It is sufficient to prove the statement of the proposition for transient PMFs only since

$$\psi = \lim_{k \to \infty} \psi[k]$$
$$\psi' = \lim_{k \to \infty} \psi'[k].$$

We proceed by induction on $k$:

— $k = 0$:
Note that the only non-zero values of the initial PMFs of $DTMC(G)$ and $DTMC(G')$ are $\psi[0]([G]_{\approx})$ and $\psi[0]([G']_{\approx})$. Let $\mathcal{H}_0$ be the equivalence class containing $[G]_{\approx}$ and $[G']_{\approx}$. Then

$$\begin{aligned} \sum_{s \in \mathcal{H}_0 \cap DR(G)} \psi[0](s) &= \psi[0]([G]_{\approx}) \\ &= 1 \\ &= \psi'[0]([G']_{\approx}) \\ &= \sum_{s' \in \mathcal{H}_0 \cap DR(G')} \psi'[0](s'). \end{aligned}$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

As for other equivalence classes, for all

$$\mathcal{H} \in ((DR(G) \cup DR(G'))/_{\mathcal{R}}) \setminus \mathcal{H}_0,$$

we have

$$\sum_{s \in \mathcal{H} \cap DR(G)} \psi[0](s) \;=\; 0 \;=\; \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'[0](s').$$

— $k \to k+1$:
Let

$$\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$$

and $s_1, s_2 \in \mathcal{H}$. Then for all

$$\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$$

and all $A \in I\!N_f^{\mathcal{L}}$, we have that

$$s_1 \stackrel{A}{\to}_{\mathcal{P}} \widetilde{\mathcal{H}} \quad \text{iff} \quad s_2 \stackrel{A}{\to}_{\mathcal{P}} \widetilde{\mathcal{H}}.$$

Therefore,

$$\begin{aligned}
PM(s_1, \widetilde{\mathcal{H}}) &= \sum_{\{\Gamma \mid \exists \mathfrak{s}_1 \in \widetilde{\mathcal{H}} \;\; s_1 \stackrel{\Gamma}{\to} \mathfrak{s}_1\}} PT(\Gamma, s_1) \\
&= \sum_{A \in I\!N_f^{\mathcal{L}}} \sum_{\{\Gamma \mid \exists \mathfrak{s}_1 \in \widetilde{\mathcal{H}} \;\; s_1 \stackrel{\Gamma}{\to} \mathfrak{s}_1, \;\; \mathcal{L}(\Gamma) = A\}} PT(\Gamma, s_1) \\
&= \sum_{A \in I\!N_f^{\mathcal{L}}} PM_A(s_1, \widetilde{\mathcal{H}}) \\
&= \sum_{A \in I\!N_f^{\mathcal{L}}} PM_A(s_2, \widetilde{\mathcal{H}}) \\
&= \sum_{A \in I\!N_f^{\mathcal{L}}} \sum_{\{\Gamma \mid \exists \mathfrak{s}_2 \in \widetilde{\mathcal{H}} \;\; s_2 \stackrel{\Gamma}{\to} \mathfrak{s}_2, \;\; \mathcal{L}(\Gamma) = A\}} PT(\Gamma, s_2) \\
&= \sum_{\{\Gamma \mid \exists \mathfrak{s}_2 \in \widetilde{\mathcal{H}} \;\; s_2 \stackrel{\Gamma}{\to} \mathfrak{s}_2\}} PT(\Gamma, s_2) \\
&= PM(s_2, \widetilde{\mathcal{H}}).
\end{aligned}$$

Since we have the previous equality for all $s_1, s_2 \in \mathcal{H}$, we can use the denotation

$$PM(\mathcal{H}, \widetilde{\mathcal{H}}) = PM(s_1, \widetilde{\mathcal{H}}) = PM(s_2, \widetilde{\mathcal{H}}).$$

Note that transitions from the states of $DR(G)$ always lead to states from the same set, so

$$\forall s \in DR(G) \;\; PM(s, \widetilde{\mathcal{H}}) = PM(s, \widetilde{\mathcal{H}} \cap DR(G)).$$

The same is true for $DR(G')$.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

By the induction hypothesis,

$$\sum_{s\in\mathcal{H}\cap DR(G)}\psi[k](s)=\sum_{s'\in\mathcal{H}\cap DR(G')}\psi'[k](s').$$

Furthermore,

$$\begin{aligned}
\sum_{\tilde{s}\in\widetilde{\mathcal{H}}\cap DR(G)}\psi[k+1](\tilde{s}) &= \sum_{\tilde{s}\in\widetilde{\mathcal{H}}\cap DR(G)}\sum_{s\in DR(G)}\psi[k](s)PM(s,\tilde{s})\\
&= \sum_{s\in DR(G)}\sum_{\tilde{s}\in\widetilde{\mathcal{H}}\cap DR(G)}\psi[k](s)PM(s,\tilde{s})\\
&= \sum_{s\in DR(G)}\psi[k](s)\sum_{\tilde{s}\in\widetilde{\mathcal{H}}\cap DR(G)}PM(s,\tilde{s})\\
&= \sum_{\mathcal{H}}\sum_{s\in\mathcal{H}\cap DR(G)}\psi[k](s)\sum_{\tilde{s}\in\widetilde{\mathcal{H}}\cap DR(G)}PM(s,\tilde{s})\\
&= \sum_{\mathcal{H}}\sum_{s\in\mathcal{H}\cap DR(G)}\psi[k](s)\sum_{\tilde{s}\in\widetilde{\mathcal{H}}\cap DR(G)}\sum_{\{\Gamma|s\xrightarrow{\Gamma}\tilde{s}\}}PT(\Gamma,s)\\
&= \sum_{\mathcal{H}}\sum_{s\in\mathcal{H}\cap DR(G)}\psi[k](s)\sum_{\{\Gamma|\exists\tilde{s}\in\widetilde{\mathcal{H}}\cap DR(G)\ s\xrightarrow{\Gamma}\tilde{s}\}}PT(\Gamma,s)\\
&= \sum_{\mathcal{H}}\sum_{s\in\mathcal{H}\cap DR(G)}\psi[k](s)PM(s,\widetilde{\mathcal{H}})\\
&= \sum_{\mathcal{H}}\sum_{s\in\mathcal{H}\cap DR(G)}\psi[k](s)PM(\mathcal{H},\widetilde{\mathcal{H}})\\
&= \sum_{\mathcal{H}}PM(\mathcal{H},\widetilde{\mathcal{H}})\sum_{s\in\mathcal{H}\cap DR(G)}\psi[k](s)\\
&= \sum_{\mathcal{H}}PM(\mathcal{H},\widetilde{\mathcal{H}})\sum_{s'\in\mathcal{H}\cap DR(G')}\psi'[k](s')\\
&= \sum_{\mathcal{H}}\sum_{s'\in\mathcal{H}\cap DR(G')}\psi'[k](s')PM(\mathcal{H},\widetilde{\mathcal{H}})\\
&= \sum_{\mathcal{H}}\sum_{s'\in\mathcal{H}'\cap DR(G')}\psi'[k](s')PM(s',\widetilde{\mathcal{H}})\\
&= \sum_{\mathcal{H}}\sum_{s'\in\mathcal{H}\cap DR(G')}\psi'[k](s')\sum_{\{\Gamma|\exists\tilde{s}'\in\widetilde{\mathcal{H}}\cap DR(G')\ s'\xrightarrow{\Gamma}\tilde{s}'\}}PT(\Gamma,s')\\
&= \sum_{\mathcal{H}}\sum_{s'\in\mathcal{H}\cap DR(G')}\psi'[k](s')\sum_{\tilde{s}'\in\widetilde{\mathcal{H}}\cap DR(G')}\sum_{\{\Gamma|\exists\tilde{s}'\ s'\xrightarrow{\Gamma}\tilde{s}'\}}PT(\Gamma,s')\\
&= \sum_{\mathcal{H}}\sum_{s'\in\mathcal{H}\cap DR(G')}\psi'[k](s')\sum_{\tilde{s}'\in\widetilde{\mathcal{H}}\cap DR(G')}PM(s',\tilde{s}')
\end{aligned}$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

$$
\begin{aligned}
&= \sum_{s' \in DR(G')} \psi'[k](s') \sum_{\widetilde{s}' \in \widetilde{\mathcal{H}} \cap DR(G')} PM(s', \widetilde{s}') \\
&= \sum_{s' \in DR(G')} \sum_{\widetilde{s}' \in \widetilde{\mathcal{H}} \cap DR(G')} \psi'[k](s') PM(s', \widetilde{s}') \\
&= \sum_{\widetilde{s}' \in \widetilde{\mathcal{H}} \cap DR(G')} \sum_{s' \in DR(G')} \psi'[k](s') PM(s', \widetilde{s}') \\
&= \sum_{\widetilde{s}' \in \widetilde{\mathcal{H}} \cap DR(G')} \psi'[k+1](\widetilde{s}').
\end{aligned}
$$

□

A.3. *Proof of Theorem 7.3*

**Theorem 7.3.** Let $G, G'$ be dynamic expressions with $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$. Let $\psi$ be the steady-state PMF for $DTMC(G)$ and $\psi'$ be the steady-state PMF for $DTMC(G')$. Let $\Sigma$ be a derived step trace of $G$ and $G'$. Then for all $\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$, we have

$$\sum_{s \in \mathcal{H} \cap DR(G)} \psi(s) PT(\Sigma, s) = \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'(s') PT(\Sigma, s').$$

*Proof.* Let

$$\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$$

and $s, \overline{s} \in \mathcal{H}$. For all

$$\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$$

and all $A \in \mathbb{N}_f^{\mathcal{L}}$, we have that

$$s \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}} \quad \text{iff} \quad \overline{s} \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}}.$$

This is valid for all $s, \overline{s} \in \mathcal{H}$, so we can rewrite it as $\mathcal{H} \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}}$ and use the denotation

$$PM_A(\mathcal{H}, \widetilde{\mathcal{H}}) = PM_A(s, \widetilde{\mathcal{H}}) = PM_A(\overline{s}, \widetilde{\mathcal{H}}).$$

Note that transitions from the states of $DR(G)$ always lead to those from the same set, so for all $s \in DR(G)$, we have

$$PM_A(s, \widetilde{\mathcal{H}}) = PM_A(s, \widetilde{\mathcal{H}} \cap DR(G)).$$

The same is true for $DR(G')$.

Let $\Sigma = A_1 \cdots A_n$ be a derived step trace of $G$ and $G'$. Then there exist

$$\mathcal{H}_0, \ldots, \mathcal{H}_n \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$$

such that

$$\mathcal{H}_0 \xrightarrow{A_1}_{\mathcal{P}_1} \mathcal{H}_1 \xrightarrow{A_2}_{\mathcal{P}_2} \cdots \xrightarrow{A_n}_{\mathcal{P}_n} \mathcal{H}_n.$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

We will now prove that the sum of probabilities of all the paths starting in every $s_0 \in \mathcal{H}_0$ and going through the states from $\mathcal{H}_1, \ldots, \mathcal{H}_n$ is equal to the product of $\mathcal{P}_1, \ldots, \mathcal{P}_n$:

$$\sum_{\{\Gamma_1,\ldots,\Gamma_n \mid s_0 \xrightarrow{\Gamma_1} \cdots \xrightarrow{\Gamma_n} s_n,\ \mathcal{L}(\Gamma_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \leqslant i \leqslant n)\}} \prod_{i=1}^{n} PT(\Gamma_i, s_{i-1}) = \prod_{i=1}^{n} PM_{A_i}(\mathcal{H}_{i-1}, \mathcal{H}_i).$$

We will prove this equality by induction on the derived step trace length $n$:

— $n = 1$:
We have

$$\sum_{\{\Gamma_1 \mid s_0 \xrightarrow{\Gamma_1} s_1,\ \mathcal{L}(\Gamma_1)=A_1,\ s_1 \in \mathcal{H}_1\}} PT(\Gamma_1, s_0) = PM_{A_1}(s_0, \mathcal{H}_1)$$
$$= PM_{A_1}(\mathcal{H}_0, \mathcal{H}_1).$$

— $n \to n+1$:
We have

$$\sum_{\{\Gamma_1,\ldots,\Gamma_n,\Gamma_{n+1} \mid s_0 \xrightarrow{\Gamma_1} \cdots \xrightarrow{\Gamma_n} s_n \xrightarrow{\Gamma_{n+1}} s_{n+1},\ \mathcal{L}(\Gamma_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \leqslant i \leqslant n+1)\}} \prod_{i=1}^{n+1} PT(\Gamma_i, s_{i-1})$$
$$= \sum_{\{\Gamma_{n+1} \mid s_n \xrightarrow{\Gamma_{n+1}} s_{n+1},\ \mathcal{L}(\Gamma_{n+1})=A_{n+1},\ s_n \in \mathcal{H}_n,\ s_{n+1} \in \mathcal{H}_{n+1}\}}$$
$$\sum_{\{\Gamma_1,\ldots,\Gamma_n \mid s_0 \xrightarrow{\Gamma_1} \cdots \xrightarrow{\Gamma_n} s_n,\ \mathcal{L}(\Gamma_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \leqslant i \leqslant n)\}} \prod_{i=1}^{n} PT(\Gamma_i, s_{i-1}) PT(\Gamma_{n+1}, s_n)$$
$$= \sum_{\{\Gamma_1,\ldots,\Gamma_n \mid s_0 \xrightarrow{\Gamma_1} \cdots \xrightarrow{\Gamma_n} s_n,\ \mathcal{L}(\Gamma_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \leqslant i \leqslant n)\}}$$
$$\left[ \prod_{i=1}^{n} PT(\Gamma_i, s_{i-1}) \sum_{\{\Gamma_{n+1} \mid s_n \xrightarrow{\Gamma_{n+1}} s_{n+1},\ \mathcal{L}(\Gamma_{n+1})=A_{n+1},\ s_n \in \mathcal{H}_n,\ s_{n+1} \in \mathcal{H}_{n+1}\}} PT(\Gamma_{n+1}, s_n) \right]$$
$$= \sum_{\{\Gamma_1,\ldots,\Gamma_n \mid s_0 \xrightarrow{\Gamma_1} \cdots \xrightarrow{\Gamma_n} s_n,\ \mathcal{L}(\Gamma_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \leqslant i \leqslant n)\}} \prod_{i=1}^{n} PT(\Gamma_i, s_{i-1}) PM_{A_{n+1}}(s_n, \mathcal{H}_{n+1})$$
$$= \sum_{\{\Gamma_1,\ldots,\Gamma_n \mid s_0 \xrightarrow{\Gamma_1} \cdots \xrightarrow{\Gamma_n} s_n,\ \mathcal{L}(\Gamma_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \leqslant i \leqslant n)\}} \prod_{i=1}^{n} PT(\Gamma_i, s_{i-1}) PM_{A_{n+1}}(\mathcal{H}_n, \mathcal{H}_{n+1})$$
$$= PM_{A_{n+1}}(\mathcal{H}_n, \mathcal{H}_{n+1}) \sum_{\{\Gamma_1,\ldots,\Gamma_n \mid s_0 \xrightarrow{\Gamma_1} \cdots \xrightarrow{\Gamma_n} s_n,\ \mathcal{L}(\Gamma_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \leqslant i \leqslant n)\}} \prod_{i=1}^{n} PT(\Gamma_i, s_{i-1})$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

$$= PM_{A_{n+1}}(\mathcal{H}_n, \mathcal{H}_{n+1}) \prod_{i=1}^{n} PM_{A_i}(\mathcal{H}_{i-1}, \mathcal{H}_i)$$
$$= \prod_{i=1}^{n+1} PM_{A_i}(\mathcal{H}_{i-1}, \mathcal{H}_i).$$

We now let $s_0, \overline{s}_0 \in \mathcal{H}_0$. We have

$$\begin{aligned}
&PT(A_1 \cdots A_n, s_0) \\
&\quad = \sum_{\{\Gamma_1,\ldots,\Gamma_n \mid s_0 \xrightarrow{\Gamma_1} \cdots \xrightarrow{\Gamma_n} s_n,\ \mathcal{L}(\Gamma_i)=A_i,\ (1\leqslant i\leqslant n)\}} \prod_{i=1}^{n} PT(\Gamma_i, s_{i-1}) \\
&\quad = \sum_{\mathcal{H}_1,\ldots,\mathcal{H}_n} \sum_{\{\Gamma_1,\ldots,\Gamma_n \mid s_0 \xrightarrow{\Gamma_1} \cdots \xrightarrow{\Gamma_n} s_n,\ \mathcal{L}(\Gamma_i)=A_i,\ s_i\in\mathcal{H}_i\ (1\leqslant i\leqslant n)\}} \prod_{i=1}^{n} PT(\Gamma_i, s_{i-1}) \\
&\quad = \sum_{\mathcal{H}_1,\ldots,\mathcal{H}_n} \prod_{i=1}^{n} PM_{A_i}(\mathcal{H}_{i-1}, \mathcal{H}_i) \\
&\quad = \sum_{\mathcal{H}_1,\ldots,\mathcal{H}_n} \sum_{\{\overline{\Gamma}_1,\ldots,\overline{\Gamma}_n \mid \overline{s}_0 \xrightarrow{\overline{\Gamma}_1} \cdots \xrightarrow{\overline{\Gamma}_n} \overline{s}_n,\ \mathcal{L}(\overline{\Gamma}_i)=A_i,\ \overline{s}_i\in\mathcal{H}_i\ (1\leqslant i\leqslant n)\}} \prod_{i=1}^{n} PT(\overline{\Gamma}_i, \overline{s}_{i-1}) \\
&\quad = \sum_{\{\overline{\Gamma}_1,\ldots,\overline{\Gamma}_n \mid \overline{s}_0 \xrightarrow{\overline{\Gamma}_1} \cdots \xrightarrow{\overline{\Gamma}_n} \overline{s}_n,\ \mathcal{L}(\overline{\Gamma}_i)=A_i,\ (1\leqslant i\leqslant n)\}} \prod_{i=1}^{n} PT(\overline{\Gamma}_i, \overline{s}_{i-1}) \\
&\quad = PT(A_1 \cdots A_n, \overline{s}_0).
\end{aligned}$$

Since we have the previous equality for all $s_0, \overline{s}_0 \in \mathcal{H}_0$, we can use the denotation

$$PT(A_1 \cdots A_n, \mathcal{H}_0) = PT(A_1 \cdots A_n, s_0) = PT(A_1 \cdots A_n, \overline{s}_0).$$

By Proposition 7.1,

$$\sum_{s\in\mathcal{H}\cap DR(G)} \psi(s) = \sum_{s'\in\mathcal{H}\cap DR(G')} \psi'(s'),$$

and we can now complete the proof as follows:

$$\begin{aligned}
\sum_{s\in\mathcal{H}\cap DR(G)} \psi(s) PT(\Sigma, s) &= \sum_{s\in\mathcal{H}\cap DR(G)} \psi(s) PT(\Sigma, \mathcal{H}) \\
&= PT(\Sigma, \mathcal{H}) \sum_{s\in\mathcal{H}\cap DR(G)} \psi(s)
\end{aligned}$$

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

$$= PT(\Sigma, \mathcal{H}) \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'(s')$$
$$= \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'(s') PT(\Sigma, \mathcal{H})$$
$$= \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'(s') PT(\Sigma, s').$$

□

**Acknowledgements**

I would like to thank Eike Best for encouraging discussions and valuable advice related to the subject of this paper. I am grateful to Peter Buchholz and Falko Bause for interest in my work and their informed consideration. Many thanks are also due to the anonymous referees for their helpful suggestions for improvements to the paper.

**References**

van der Aalst, W. M. P., van Hee, K. M. and Reijers, H. A. (2000) Analysis of discrete-time stochastic Petri nets. *Statistica Neerlandica* **54** (2) 237–255.

Autant, C. and Schnoebelen, Ph. (1992) Place bisimulations in Petri nets. In: Proceedings of 13th ICATPN. *Springer-Verlag Lecture Notes in Computer Science* **616** 45–61.

Baarir, S., Beccuti, M., Dutheillet, C., Franceschinis, G. and Haddad, S. (2011) Lumping partially symmetric stochastic models. *Performance Evaluation* **68** 21–44.

Bergstra, J. A. and Klop, J. W. (1985) Algebra of communicating processes with abstraction. *Theoretical Computer Science* **37** 77–121.

Bernardo, M. (2007) A survey of Markovian behavioral equivalences. In: Proceedings of 7th SFM. *Springer-Verlag Lecture Notes in Computer Science* **4486** 180–219.

Bernardo, M., Donatiello, L. and Gorrieri, R. (1998) A formal approach to the integration of performance aspects in the modeling and analysis of concurrent systems. *Information and Computation* **144** (2) 83–154.

Bernardo, M. and Gorrieri, R. (1998) A tutorial on EMPA: a theory of concurrent processes with nondeterminism, priorities, probabilities and time. *Theoretical Computer Science* **202** 1–54.

Best, E. and Koutny, M. (1995) A refined view of the box algebra. In: Proceedings of 16th ICATPN. *Springer-Verlag Lecture Notes in Computer Science* **935** 1–20.

Best, E., Devillers, R. and Hall, J. G. (1992) The box calculus: a new causal algebra with multi-label communication. In: Advances in Petri Nets 1992. *Springer-Verlag Lecture Notes in Computer Science* **609** 21–69.

Best, E., Devillers, R. and Koutny, M. (2001) *Petri net algebra*, EATCS Monographs on Theoretical Computer Science, Springer-Verlag.

Bravetti, M., Bernardo, M. and Gorrieri, R. (2008) Towards performance evaluation with general distributions in process algebras. In: Proceedings of 9th CONCUR. *Springer-Verlag Lecture Notes in Computer Science* **1466** 405–422.

Brinksma, E. and Hermanns, H. (2001) Process algebra and Markov chains. In: Proceedings of 1st EEF/Euro Summer School of Trends in Computer Science. *Springer-Verlag Lecture Notes in Computer Science* **2090** 183–231.

Brinksma, E., Katoen, J.-P., Langerak, R. and Latella, D. (1995) A stochastic causality-based process algebra. *The Computer Journal* **38** (7) 552–565.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Buchholz, P. (1995) A notion of equivalence for stochastic Petri nets. In: Proceedings of 16th ICATPN. *Springer-Verlag Lecture Notes in Computer Science* **935** 161–180.

Buchholz, P. (1998) Iterative decomposition and aggregation of labeled GSPNs. In: Proceedings of 19th ICATPN. *Springer-Verlag Lecture Notes in Computer Science* **1420** 226–245.

Buchholz, P. and Tarasyuk, I. V. (2001) Net and algebraic approaches to probablistic modeling. *Joint Novosibirsk Computing Center and Institute of Informatics Systems Bulletin, Series Computer Science* **15** 31–64.

Derisavi, S., Hermanns, H. and Sanders, W. H. (2003) Optimal state-space lumping of Markov chains. *Information Processing Letters* **87** (6) 309–315.

Fourneau, J. M. (2010) Collaboration of discrete-time Markov chains: Tensor and product form. *Performance Evaluation* **67** 779–796.

van Glabbeek, R. J., Smolka, S. A. and Steffen, B. (1995) Reactive, generative, and stratified models of probabilistic processes. *Information and Computation* **121** (1) 59–80.

Hermanns, H. and Rettelbach, M. (1994) Syntax, semantics, equivalences and axioms for MTIPP. In: Herzog U. and Rettelbach M. (eds.) Proceedings of 2nd International Workshop on Process Algebras and Performance Modelling (PAPM). *Arbeitsberichte des IMMD* **27**, University of Erlangen 71–88.

Hillston, J. (1994) The nature of synchronisation. Proceedings of 2nd International Workshop on Process Algebras and Performance Modelling (PAPM). *Arbeitsberichte des IMMD* **27**, University of Erlangen 51–70.

Hillston, J. (1996) *A compositional approach to performance modelling*, Cambridge University Press.

Horváth, A., Paolieri, M., Ridi, L. and Vicario, E. (2012) Transient analysis of non-Markovian models using stochastic state classes. *Performance Evaluation* **69** (7–8) 315–335.

Jou, C.-C. and Smolka, S. A. (1990) Equivalences, congruences and complete axiomatizations for probabilistic processes. In: Proceedings of 1st CONCUR. *Springer-Verlag Lecture Notes in Computer Science* **458** 367–383.

Katoen, J.-P., Brinksma, E., Latella, D. and Langerak, R. (1996) Stochastic simulation of event structures. In: Ribaudo, M. (ed.) *Proceedings of 4th International Workshop on Process Algebra and Performance Modelling (PAPM)*, CLUT Press, Torino, Italy 21–40.

Katoen, J.-P. and D'Argenio, P. R. (2001) General distributions in process algebra. In: Proceedings of 1st EEF/Euro Summer School of Trends in Computer Science. *Springer-Verlag Lecture Notes in Computer Science* **2090** 375–429.

Katoen, J.-P., Zapreev, I. S., Hahn, E. M., Hermanns, H. and Jansen, D. N. (2011) The ins and outs of the probabilistic model checker MRMC. *Performance Evaluation* **68** 90–104.

Koutny, M. (2000) A compositional model of time Petri nets. In: Proceedings of 21st ICATPN. *Springer-Verlag Lecture Notes in Computer Science* **1825** 303–322.

Kulkarni, V. G. (2009) Modeling and analysis of stochastic systems. *Texts in Statistical Science*, Chapman and Hall/CRC Press.

Kwiatkowska, M. Z., Norman, G. J. and Parker, D. (2006) Symmetry reduction for probabilistic model checking. In: Proceedings of ICIC 2006. *Springer-Verlag Lecture Notes in Computer Science* **4114** 234–248.

Larsen, K. G. and Skou, A. (1991) Bisimulation through probabilistic testing. *Information and Computation* **94** (1) 1–28.

Macià, H., Valero, V., Cazorla, D. and Cuartero, F. (2004) Introducing the iteration in sPBC. In: Proceedings of 24th FORTE. *Springer-Verlag Lecture Notes in Computer Science* **3235** 292–308.

Macià, H., Valero, V., Cuartero, F. and Ruiz, M. C. (2008) sPBC: a Markovian extension of Petri box calculus with immediate multiactions. *Fundamenta Informaticae* **87** (3-4) 367–406.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press

Macià, H., Valero, V. and de-Frutos, D. (2001) sPBC: a Markovian extension of finite Petri box calculus. *Proceedings of 9th IEEE International Workshop on Petri Nets and Performance Models (PNPM)*, IEEE Computer Society Press 207–216.

Markovski, J. and de Vink, E. P. (2008) Extending timed process algebra with discrete stochastic time. In: Proceedings of 12th AMAST. *Springer-Verlag Lecture Notes in Computer Science* **5140** 268–283.

Markovski, J. and de Vink, E. P. (2009) Performance evaluation of distributed systems based on a discrete real- and stochastic-time process algebra. *Fundamenta Informaticae* **95** (1) 157–186.

Marroquín, O. and de-Frutos, D. (2001) Extending the Petri box calculus with time. In: Proceedings of 22nd ICATPN. *Springer-Verlag Lecture Notes in Computer Science* **2075** 303–322.

Marsan, M. A. (1990) Stochastic Petri nets: an elementary introduction. In: Advances in Petri Nets 1989. *Springer-Verlag Lecture Notes in Computer Science* **424** 1–29.

Merlin, P. and Farber, D. J. (1976) Recoverability of communication protocols: implications of a theoretical study. *IEEE Transactions on Communications* **24** (9) 1036–1043.

Milner, R. A. J. (1989) *Communication and concurrency*, Prentice-Hall.

Milner, R. A. J., Parrow, J. and Walker, D. (1992) A calculus of mobile processes (I and II). *Information and Computation* **100** (1) 1–77.

Molloy, M. K. (1981) *On the integration of the throughput and delay measures in distributed processing models*, Ph.D. thesis, University of California, Los Angeles.

Molloy, M. K. (1985) Discrete time stochastic Petri nets. *IEEE Transactions on Software Engineering* **11** (4) 417–423.

Niaouris, A. (2005) An algebra of Petri nets with arc-based time restrictions. In: Proceedings of 1st ICTAC. *Springer-Verlag Lecture Notes in Computer Science* **3407** 447–462.

Paige, R. and Tarjan, R. E. (1987) Three partition refinement algorithms. *SIAM Journal on Computing* **16** (6) 973–989.

Peterson, J. L. (1981) *Petri net theory and modeling of systems*, Prentice-Hall.

Priami, C. (1996) Stochastic $\pi$-calculus with general distributions. In: Ribaudo, M. (ed.) *Proceedings of 4th International Workshop on Process Algebra and Performance Modelling (PAPM)*, CLUT Press, Torino, Italy , 41–57.

Ramchandani, C. (1973) *Perfomance evaluation of asynchronous concurrent systems by timed Petri nets*, Ph.D. thesis, Massachusetts Institute of Technology.

Tarasyuk, I. V. (2005) Discrete time stochastic Petri box calculus. *Berichte aus dem Department für Informatik* **3/05**, Carl von Ossietzky Universität Oldenburg, Germany.

Tarasyuk, I. V. (2006) Iteration in discrete time stochastic Petri box calculus. *Bulletin of the Novosibirsk Computing Center, Series Computer Science, IIS Special Issue* **24** 129–148.

Tarasyuk, I. V. (2007) Stochastic Petri box calculus with discrete time. *Fundamenta Informaticae* **76** (1–2) 189–218.

Tarasyuk, I. V. (2008) Investigating equivalence relations in dtsPBC. *Berichte aus dem Department für Informatik* **5/08**, Carl von Ossietzky Universität Oldenburg, Germany.

Tarasyuk, I. V., Macià, H. and Valero, V. (2010) Discrete time stochastic Petri box calculus with immediate multiactions. Technical Report **DIAB-10-03-1**, Department of Computer Systems, High School of Computer Science Engineering, University of Castilla-La Mancha, Albacete, Spain (available at `http://itar.iis.nsk.su/files/itar/pages/dtsipbc.pdf`).

Wimmer, R., Derisavi, S. and Hermanns, H. (2010) Symbolic partition refinement with automatic balancing of time and space. *Performance Evaluation* **67** 816–836.

https://doi.org/10.1017/S0960129513000029 Published online by Cambridge University Press