Bull. Nov. Comp. Center, Comp. Science, 24 (2006), 129–148
© 2006 NCC Publisher

Iteration in discrete time stochastic Petri box calculus

I. V. Tarasyuk

Abstract. In the last decades, a number of stochastic enrichments of process algebras was constructed to specify stochastic processes within the well-developed framework of algebraic calculi. In [26], a continuous time stochastic extension of finite Petri box calculus (*PBC*) was proposed and called *sPBC*. The algebra *sPBC* has interleaving semantics due to the properties of continuous time distributions. The iteration operator was added to *sPBC* in [24] to specify infinite processes. Since *PBC* has step semantics, it could be more natural to propose its concurrent stochastic enrichment based on discrete time distributions. In [28], a discrete time stochastic extension *dtsPBC* of finite *PBC* was constructed. In this paper, we construct an enrichment of *dtsPBC* with iteration. A step operational semantics is defined in terms of labeled transition systems based on action and inaction rules. A denotational semantics is defined in terms of a subclass of labeled discrete time stochastic Petri nets (LDTSPNs) called discrete time stochastic Petri boxes (dts-boxes). Consistency of both semantics is demonstrated.

Keywords: stochastic Petri nets, stochastic process algebras, Petri box calculus, iteration, discrete time, transition systems, operational semantics, dts-boxes, denotational semantics.

1. Introduction

Stochastic Petri nets (SPNs) are a well-known model for quantitative analysis of discrete dynamic event systems proposed initially in [20]. A stochastic process corresponding to this formal model is a Markov chain generated and analyzed by well-developed algorithms and methods. Firing probabilities distributed along continuous or discrete time scale are associated with transitions of an SPN. Thus, there exist SPNs with continuous [20, 13] and discrete [21] time. Markov chains of the corresponding types are associated with the SPNs. As a rule, for SPNs with continuous time (CTSPNs), exponential or phase distributions of transition probabilities are used. For SPNs with discrete time (DTSPNs), geometric distributions or their combinations are usually used. Transitions of CTSPNs fire one by one at continuous time moments. Hence, the semantics of this model is an interleaving one, where parallel computations are modeled by all possible execution sequences of their components. Transitions of DTSPNs fire concurrently in steps at discrete time moments. Hence, this model has a step semantics, where parallel computations are modeled by sequences of concurrent occurrences (steps) of their components. In [10, 11], a labeling for transitions of CTSPNs with action names was proposed. The labeling allows SPNs to model processes with functionally similar components: the transitions corresponding to the similar components are labeled by the same action. Therefore, one can compare both functional and performance properties, and labeled SPNs turn into a formalism for quantitative and qualitative analysis.

Algebraic calculi occupy a special place among formal models for specification of concurrent systems and analysis of their behavioral properties. In such process algebras (PAs), a system or a process is specified by an algebraic formula. Verification of the properties is accomplished at a syntactic level by means of well-developed systems of equivalences, axioms and inference rules. One of the first PAs was CCS (Calculus of Communicating Systems) [19]. Process algebras have been acknowledged to be very suitable formalism to operate with real time and stochastic systems as well. In the last years, stochastic extensions of PAs called stochastic process algebras (SPAs) became very popular as a modeling framework. SPAs do not just specify actions that can happen (qualitative features) as usual process algebras, but they associate some quantitative parameters with actions (quantitative characteristics). The most popular SPAs proposed so far are $PEPA$ [14], $TIPP$ [15] and $EMPA$ [4]. The papers [5, 9, 12, 27] propose a variety of other SPAs.

Process algebras allow one to specify processes in a compositional way via an expressive formal syntax. On the other hand, Petri nets provide one with an ability for visual representation of a process structure and execution. Hence, the relationship between SPNs and SPAs is of particular interest, since it allows one to combine advantages of both models. For this, a semantics of algebraic formulas in terms of Petri nets is usually defined. In the stochastic case, the Markov chain of the stochastic process specified by an SPA formula is built based on the state transition graph of the corresponding SPN.

As a rule, stochastic process calculi proposed in the literature are based on interleaving. As a semantic domain, the interleaving formalism of transition systems is often used. Therefore, investigation of a stochastic extension for more expressive and powerful algebraic calculi is an important issue. At present, the development of step or "true concurrency" (such that parallelism is considered as a causal independence) SPAs is in the very beginning. At the same time, there does not yet exist an algebra of infinite concurrent stochastic processes.

Petri box calculus (PBC) is a flexible and expressive process algebra based on calculi CCS [19] and AFP_0 [18]. PBC was introduced more than 10 years ago [1], and it was well explored since that time [16, 2, 3]. Its goal was to propose a compositional semantics for high level constructs of concurrent programming languages in terms of elementary Petri nets. Formulas of PBC are combined not from single actions and variables, as in CCS, but from multisets of actions called multiactions which are the *basic formulas* of the calculus. In contrast to CCS, concurrency and synchronization are different operations, they form the *concurrent constructs.* The *sequential constructs* are sequence and choice. The *abstraction constructs* include restriction and relabeling operations. To specify infinite processes, the *hierarchical constructs* such as refinement, recursion and iteration were added. Thus, unlike CCS, the algebra PBC has an additional iteration construction to specify infiniteness in the cases when finite Petri nets can be used as the semantic interpretation. For PBC, a denotational semantics was proposed in terms of Petri boxes which are a subclass of Petri nets equipped with interface and considered up to isomorphism. The calculus PBC has a step operational semantics in terms of labeled transition systems based on the structural operational semantics (SOS) rules.

A stochastic extension of PBC called stochastic Petri box calculus ($sPBC$) was proposed in [26, 22]. In $sPBC$, multiactions have stochastic durations that follow negative exponential distribution. Each multiaction is instantaneous and equipped with a rate that is a parameter of the corresponding exponential distribution. The execution of a multiaction is possible only after the corresponding stochastic time delay. Only a finite part of PBC was used for the stochastic enrichment. This means that $sPBC$ has neither refinement nor recursion nor iteration operations. A denotational semantics was defined in terms of a subclass of labeled CTSPNs called stochastic Petri boxes (s-boxes). Calculus $sPBC$ has interleaving operational semantics in terms of labeled transition systems. Note that we have interleaving behaviour here because of the fact that a simultaneous firing of any two transitions has zero probability in accordance with the properties of continuous time distributions. Current research in this branch has an aim to extend the specification abilities of $sPBC$ and to define an appropriate congruence relation over algebraic formulas. The results on constructing the iteration for $sPBC$ were reported in [24]. In the paper [23], a number of new equivalence relations were proposed for regular terms of $sPBC$ to choose later a suitable candidate for a congruence. In [25], the special multiactions with zero time delay were added to $sPBC$. A denotational semantics of such an $sPBC$ extension was defined via a subclass of labeled generalized SPNs (GSPNs). The subclass is called generalized stochastic Petri boxes (gs-boxes).

Nevertheless, there were no stochastic extension of PBC with step semantics until recent times. It can be done with the use of labeled DTSPNs as a semantic domain, since discrete time models allow for concurrent action occurrences. The enrichment based of DTSPNs is natural because PBC has a step operational semantics.

We did some work on the development of concurrent discrete time SPNs and SPAs. In [6], labeled weighted DTSPNs (LWDTSPNs) were proposed that is a modification of DTSPNs by transition labeling and weights. In [8], labeled DTSPNs (LDTSPNs) were introduced. In [7, 8], a stochastic algebra of finite nondeterministic processes $StAFP_0$ was constructed with semantics in terms of a subclass of LWDTSPNs and LDTSPNs called stochastic acyclic nets (SANs). The calculus defined is a stochastic extension of the algebra AFP_0 introduced in [17]. $StAFP_0$ specifies concurrent stochastic processes and possesses a net semantics allowing one to preserve the level of parallelism. An axiomatization for the semantic equivalence of $StAFP_0$ was proposed. In [28], a discrete time stochastic extension $dtsPBC$ of finite PBC was constructed. A step operational and a net denotational semantics of $dtsPBC$ were defined, and their consistency was demonstrated. In addition, a variety of probabilistic equivalences were proposed to identify stochastic processes with similar behaviour which are differentiated by the semantic equivalence. The interrelations of all the introduced equivalences were studied.

In this paper, we construct an enrichment of $dtsPBC$ with the iteration operator to be able to specify infinite processes. First, we present the syntax of the extended $dtsPBC$. Each multiaction of the initial calculus PBC is associated with probability. Such a pair is called stochastic multiaction or activity. Second, we propose semantics of $dtsPBC$. A step operational semantics is constructed in terms of labeled transition systems based on action and inaction rules. A denotational semantics is defined in terms of a subclass of LDTSPNs called discrete time stochastic Petri boxes (dts-boxes). Consistency of operational and denotational semantics is proved.

The paper is organized as follows. In the next Section 2, the syntax of the extended calculus $dtsPBC$ is presented. Then, in Section 3, we construct operational semantics of the algebra in terms of labeled transition systems. In Section 4, we propose denotational semantics based on a subclass of LDTSPNs. The concluding Section 5 summarizes the results obtained and outlines research perspectives in this area.

2. Syntax

In this section, we propose the syntax of discrete time stochastic extension of finite PBC enriched with iteration called *discrete time stochastic Petri box calculus dtsPBC*.

First, we recall a definition of a multiset that is an extension of the set notion by allowing several identical elements.

Definition 1. Let X be a set. A finite *multiset (bag)* M over X is a mapping $M : X \to I\!N$ such that $|\{x \in X \mid M(x) > 0\}| < \infty$.

We denote the *set of all finite multisets* over X by $I\!N_f^X$. When $\forall x \in X\ M(x) \leq 1$, M is a proper set. The *cardinality* of a multiset M is defined as $|M| = \sum_{x \in X} M(x)$. We write $x \in M$ if $M(x) > 0$ and $M \subseteq M'$ if $\forall x \in X\ M(x) \leq M'(x)$. We define $(M + M')(x) = M(x) + M'(x)$ and $(M - M')(x) = \max\{0, M(x) - M'(x)\}$.

Let $Act = \{a, b, \ldots\}$ be the set of *elementary actions.* Then $\widehat{Act} = \{\hat{a}, \hat{b}, \ldots\}$ is the set of *conjunctive actions (conjugates)* such that $a \neq \hat{a}$ and $\hat{\hat{a}} = a$. Let $\mathcal{A} = Act \cup \widehat{Act}$ be the set of *all actions*, and $\mathcal{L} = I\!N_f^{\mathcal{A}}$ be the set of *all multiactions.* Note that $\emptyset \in \mathcal{L}$, this corresponds to an internal activity, i.e., the execution of a multiaction that contains no visible action names. The *alphabet* of $\alpha \in \mathcal{L}$ is defined as $\mathcal{A}(\alpha) = \{x \in \mathcal{A} \mid \alpha(x) > 0\}$.

An *activity (stochastic multiaction)* is a pair (α, ρ), where $\alpha \in \mathcal{L}$ and $\rho \in (0; 1)$ is the probability of the multiaction α. Let $\mathcal{SL}$ be the set of *all activities.* Let us 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 as $\mathcal{A}(\alpha, \rho) = \mathcal{A}(\alpha)$. For $(\alpha, \rho) \in \mathcal{SL}$, we define its *multiaction part* as $\mathcal{L}(\alpha, \rho) = \alpha$ and its *probability part* as $\Omega(\alpha, \rho) = \rho$.

Activities are combined into formulas by the following operations: *sequential execution* ;, *choice* [], *parallelism* $\|$, *relabeling* $[f]$, *synchronization sy*, *restriction rs* and *iteration* $[**]$.

Relabeling functions $f : \mathcal{A} \to \mathcal{A}$ are bijections preserving conjugates, i.e., $\forall x \in \mathcal{A}\ f(\hat{x}) = \widehat{f(x)}$. Let $\alpha, \beta \in \mathcal{L}$ be two multiactions such that for some action $a \in Act$ we have $a \in \alpha$ and $\hat{a} \in \beta$ or $\hat{a} \in \alpha$ and $a \in \beta$. Then synchronization of α and β by a is defined as $\alpha \oplus_a \beta = \gamma$, where

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

Static expressions specify the structure of a system. As we shall see, they correspond to unmarked SPNs.

Definition 2. Let $(\alpha, \rho) \in \mathcal{SL}$ and $a \in Act$. A *static expression* of *dtsPBC* is defined as

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

Let *StatExpr* denote the set of *all static expressions.*

To avoid inconsistency of the iteration operator, we should not allow any concurrency in the highest level of the second argument of iteration. This is not a severe restriction though, since we can always prefix parallel expressions by an activity with the empty multiaction and an appropriate probability.

Definition 3. Let $(\alpha, \rho) \in \mathcal{SL}$, $a \in Act$ and $E \in StatExpr$. A *regular static expression* of *dtsPBC* is defined as

$$\begin{aligned} D ::=\ & (\alpha,\rho) \mid D;E \mid D[]D \mid D[f] \mid D \ \mathsf{rs}\ a \mid D \ \mathsf{sy}\ a \mid [D*D*E], \\ E ::=\ & (\alpha,\rho) \mid E;E \mid E[]E \mid E\|E \mid E[f] \mid E \ \mathsf{rs}\ a \mid E \ \mathsf{sy}\ a \mid [E*D*E]. \end{aligned}$$

Let *RegStatExpr* denote the set of *all regular static expressions.*

Dynamic expressions specify the states of a system. As we shall see, they correspond to marked SPNs. Note that if an underlying static expression of a dynamic one is not regular, the corresponding marked SPN can be unsafe (though, it is 2-bounded in the worst case, see [2]).

Definition 4. Let $(\alpha,\rho) \in \mathcal{SL},\ a \in Act$ and $E \in RegStatExpr$. A *regular dynamic expression* of *dtsPBC* is defined as

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

Let *RegDynExpr* denote the set of *all regular dynamic expressions.*

We shall consider regular expressions only and omit the word "regular".

3. Operational semantics

In this section, we construct the step operational semantics in terms of labeled transition systems.

3.1. Inaction rules

First, we define inaction rules for overlined and underlined static expressions. Let $E, F, K \in RegStatExpr$ and $a \in Act$.

$$\begin{array}{lll}
\overline{E;F} \xrightarrow{\emptyset} \overline{E};F & \underline{E};F \xrightarrow{\emptyset} E;\overline{F} & E;\underline{F} \xrightarrow{\emptyset} \underline{E;F} \\
\overline{E[]F} \xrightarrow{\emptyset} \overline{E}[]F & \overline{E[]F} \xrightarrow{\emptyset} E[]\overline{F} & \underline{E}[]F \xrightarrow{\emptyset} \underline{E[]F} \\
E[]\underline{F} \xrightarrow{\emptyset} \underline{E[]F} & \overline{E\|F} \xrightarrow{\emptyset} \overline{E}\|\overline{F} & \underline{E}\|\underline{F} \xrightarrow{\emptyset} \underline{E\|F} \\
\overline{E[f]} \xrightarrow{\emptyset} \overline{E}[f] & \underline{E}[f] \xrightarrow{\emptyset} \underline{E[f]} & \overline{E \ \mathsf{rs}\ a} \xrightarrow{\emptyset} \overline{E} \ \mathsf{rs}\ a \\
\underline{E} \ \mathsf{rs}\ a \xrightarrow{\emptyset} \underline{E \ \mathsf{rs}\ a} & \overline{E \ \mathsf{sy}\ a} \xrightarrow{\emptyset} \overline{E} \ \mathsf{sy}\ a & \underline{E} \ \mathsf{sy}\ a \xrightarrow{\emptyset} \underline{E \ \mathsf{sy}\ a} \\
\overline{[E*F*K]} \xrightarrow{\emptyset} [\overline{E}*F*K] & [\underline{E}*F*K] \xrightarrow{\emptyset} [E*\overline{F}*K] & \\
[E*\underline{F}*K] \xrightarrow{\emptyset} [E*\overline{F}*K] & [E*\underline{F}*K] \xrightarrow{\emptyset} [E*F*\overline{K}] & \\
[E*F*\underline{K}] \xrightarrow{\emptyset} \underline{[E*F*K]} & &
\end{array}$$

Second, we propose inaction rules for arbitrary dynamic expressions. Let $E, F \in RegStatExpr,\ G, H, \widetilde{G}, \widetilde{H} \in RegDynExpr$ and $a \in Act$.

$$G \xrightarrow{\emptyset} G \qquad \frac{G\xrightarrow{\emptyset}\widetilde{G},\ \circ\in\{;,[]\}}{G\circ E\xrightarrow{\emptyset}\widetilde{G}\circ E} \qquad \frac{G\xrightarrow{\emptyset}\widetilde{G},\ \circ\in\{;,[]\}}{E\circ G\xrightarrow{\emptyset}E\circ\widetilde{G}} \qquad \frac{G\xrightarrow{\emptyset}\widetilde{G}}{G\|H\xrightarrow{\emptyset}\widetilde{G}\|H}$$

$$\frac{H\xrightarrow{\emptyset}\widetilde{H}}{G\|H\xrightarrow{\emptyset}G\|\widetilde{H}} \qquad \frac{G\xrightarrow{\emptyset}\widetilde{G}}{G[f]\xrightarrow{\emptyset}\widetilde{G}[f]} \qquad \frac{G\xrightarrow{\emptyset}\widetilde{G},\ \circ\in\{\mathsf{rs},\mathsf{sy}\}}{G\circ a\xrightarrow{\emptyset}\widetilde{G}\circ a} \qquad \frac{G\xrightarrow{\emptyset}\widetilde{G}}{[G{*}E{*}F]\xrightarrow{\emptyset}[\widetilde{G}{*}E{*}F]}$$

$$\frac{G\xrightarrow{\emptyset}\widetilde{G}}{[E{*}G{*}F]\xrightarrow{\emptyset}[E{*}\widetilde{G}{*}F]} \qquad \frac{G\xrightarrow{\emptyset}\widetilde{G}}{[E{*}F{*}G]\xrightarrow{\emptyset}[E{*}F{*}\widetilde{G}]}$$

Note that the rule $G \xrightarrow{\emptyset} G$ is intentionally included in the set of rules above. It reflects a non-zero probability to stay in a state at the next time moment that is an essential feature of discrete time stochastic processes.

A regular dynamic expression G is *operative* if no inaction rule can be applied to it, with the exception of $G \xrightarrow{\emptyset} G$. Note that any dynamic expression can be always transformed into a (not necessarily unique) operative one using inaction rules. Let *OpRegDynExpr* denote the set of *all operative regular dynamic expressions* of *dtsPBC*.

Definition 5. Let $\simeq\, = (\xrightarrow{\emptyset} \cup \xleftarrow{\emptyset})^*$ be dynamic expression isomorphism in *dtsPBC*. Two dynamic expressions G and G' are *isomorphic*, denoted by $G \simeq G'$, if they can be reached from each other by applying inaction rules.

3.2. Action rules

Now we propose action rules which describe expression transformations due to the execution of multisets of activities. Let $(\alpha,\rho),(\beta,\chi)\in\mathcal{SL}$, $E,F\in$ *RegStatExpr*, $G,H\in$ *OpRegDynExpr*, $\widetilde{G},\widetilde{H}\in$ *RegDynExpr* and $a\in$ *Act*. Moreover, let $\Gamma,\Delta\in I\!\!N_f^{\mathcal{SL}}$. The *alphabet* of $\Gamma\in I\!\!N_f^{\mathcal{SL}}$ is defined as $\mathcal{A}(\Gamma)=\cup_{(\alpha,\rho)\in\Gamma}\mathcal{A}(\alpha)$.

$$\overline{(\alpha,\rho)} \xrightarrow{\{(\alpha,\rho)\}} \underline{(\alpha,\rho)} \qquad \frac{G\xrightarrow{\Gamma}\widetilde{G}}{G;E\xrightarrow{\Gamma}\widetilde{G};E} \qquad \frac{G\xrightarrow{\Gamma}\widetilde{G}}{E;G\xrightarrow{\Gamma}E;\widetilde{G}} \qquad \frac{G\xrightarrow{\Gamma}\widetilde{G}}{G[]E\xrightarrow{\Gamma}\widetilde{G}[]E}$$

$$\frac{G\xrightarrow{\Gamma}\widetilde{G}}{E[]G\xrightarrow{\Gamma}E[]\widetilde{G}} \qquad \frac{G\xrightarrow{\Gamma}\widetilde{G}}{G\|H\xrightarrow{\Gamma}\widetilde{G}\|H} \qquad \frac{H\xrightarrow{\Gamma}\widetilde{H}}{G\|H\xrightarrow{\Gamma}G\|\widetilde{H}} \qquad \frac{G\xrightarrow{\Gamma}\widetilde{G},\ H\xrightarrow{\Delta}\widetilde{H}}{G\|H\xrightarrow{\Gamma+\Delta}\widetilde{G}\|\widetilde{H}}$$

$$\frac{G\xrightarrow{\Gamma}\widetilde{G}}{G[f]\xrightarrow{f(\Gamma)}\widetilde{G}[f]} \qquad \frac{G\xrightarrow{\Gamma}\widetilde{G},\ a,\hat{a}\not\in\mathcal{A}(\Gamma)}{G\ \mathsf{rs}\ a\xrightarrow{\Gamma}\widetilde{G}\ \mathsf{rs}\ a} \qquad \frac{G\xrightarrow{\Gamma}\widetilde{G}}{G\ \mathsf{sy}\ a\xrightarrow{\Gamma}\widetilde{G}\ \mathsf{sy}\ a} \qquad \frac{G\xrightarrow{\Gamma}\widetilde{G}}{[G{*}E{*}F]\xrightarrow{\Gamma}[\widetilde{G}{*}E{*}F]}$$

$$\frac{G\xrightarrow{\Gamma}\widetilde{G}}{[E{*}G{*}F]\xrightarrow{\Gamma}[E{*}\widetilde{G}{*}F]} \qquad \frac{G\xrightarrow{\Gamma}\widetilde{G}}{[E{*}F{*}G]\xrightarrow{\Gamma}[E{*}F{*}\widetilde{G}]}$$

$$\frac{G\ \mathsf{sy}\ a \xrightarrow{\Gamma+\{(\alpha,\rho)\}+\{(\beta,\chi)\}} \widetilde{G}\ \mathsf{sy}\ a,\ a\in\mathcal{A}(\alpha),\ \hat{a}\in\mathcal{A}(\beta)}{G\ \mathsf{sy}\ a \xrightarrow{\Gamma+\{(\alpha\oplus_a\beta,\rho\cdot\chi)\}} \widetilde{G}\ \mathsf{sy}\ a}$$

Note that in the last rule above we multiply the probabilities of synchronized multiactions since this corresponds to the probability of event intersection.

3.3. Transition systems

Now we define transition systems associated with dynamic expressions.

The expressions of *dtsPBC* can contain identical activities. To avoid technical difficulties, such as those with proper calculation of the state change probabilities for multiple transitions, we can always enumerate coinciding activities from left to right in the syntax of expressions. In the following, we suppose that all identical activities are enumerated. The new activities resulted from synchronization will be annotated with concatenation of the numbering of the activities they come from. Such new activities will be considered up to the permutation of their numbering resulting from the applications of the second rule for synchronization. After such an enumeration, the multisets of activities over arrows in the action rules will be the proper sets.

Definition 6. Let G be a dynamic expression. Then $[G]_{\simeq} = \{H \mid G \simeq H\}$ is the equivalence class of G with respect to isomorphism (isomorphism class). The *derivation set* of a dynamic expression G, denoted by $DR(G)$, is the minimal set such that

- $[G]_{\simeq} \in DR(G)$;
- if $[H]_{\simeq} \in DR(G)$ and $\exists \Gamma\ H \xrightarrow{\Gamma} \widetilde{H}$ then $[\widetilde{H}]_{\simeq} \in DR(G)$.

Let G be a dynamic expression and $[H]_{\simeq} \in DR(G)$.

The set of *all multisets of activities executable from* H is defined as $Exec(H) = \{\Gamma \mid \exists J \in [H]_{\simeq}\ \exists \widetilde{J}\ J \xrightarrow{\Gamma} \widetilde{J}\}$.

Let $\Gamma \in Exec(H)$. The probability that the activities from Γ *try to happen* in H is

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

When $Exec(H) = \emptyset$, we define $PF(\emptyset, H) = 1$, since we stay in H in this case. Thus, $PF(\Gamma, H)$ could be interpreted as a *joint* probability of independent events. Each such an event is interpreted as trying or not trying to occur of a particular activity from Γ.

The probability that the activities from Γ *happen* in H is

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

Thus, $PT(\Gamma, H)$ is the probability that the multiset Γ tries to happen *normalized* by the probability to occur for *any* multiset executable from H.

The probability that the execution of *any* activities changes H to $\widetilde{H}$ is

$$PM(H, \widetilde{H}) = \sum_{\{\Gamma \mid \exists J \in [H]_{\simeq}, \widetilde{J} \in [\widetilde{H}]_{\simeq} \ J \xrightarrow{\Gamma} \widetilde{J}\}} PT(\Gamma, J).$$

Since $PM(H, \widetilde{H})$ is the probability for *any* multiset of activities to change H to $\widetilde{H}$, we use summation in the definition.

Definition 7. Let G be a dynamic expression. The *(labeled 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 $L_G \subseteq I\!N_f^{\mathcal{SL}} \times (0;1]$;
- the set of *transitions* is $\mathcal{T}_G = \{([H]_{\simeq}, (\Gamma, PT(\Gamma, H)), [\widetilde{H}]_{\simeq}) \mid [H]_{\simeq} \in DR(G),\ H \xrightarrow{\Gamma} \widetilde{H}\}$;
- the *initial state* is $s_G = [G]_{\simeq}$.

Thus, the transition system $TS(G)$ associated with a dynamic expression G describes all steps that happen at discrete moments of time with some (one-step) probability and consist of multisets of activities. These steps change states, and the states are the isomorphism classes of dynamic expressions obtained by application of action rules starting from the expressions belonging to $[G]_{\simeq}$. A transition $(s, (\Gamma, \mathcal{P}), \tilde{s}) \in \mathcal{T}_G$ will be written as $s \xrightarrow{\Gamma}_{\mathcal{P}} \tilde{s}$. It is interpreted as follows: the probability to change the state s to $\tilde{s}$ as a result of executing Γ is $\mathcal{P}$. We write $s \xrightarrow{\Gamma} \tilde{s}$ if $\exists \mathcal{P}\ s \xrightarrow{\Gamma}_{\mathcal{P}} \tilde{s}$.

Note that Γ could be the empty set, and its execution does not change isomorphism classes. This corresponds to the application of inaction rules to the expressions from the isomorphism classes. We have to keep track of such executions called *empty loops*, because they have nonzero probabilities by the definition of $PF(\emptyset, H)$ and the fact that multiaction probabilities cannot be equal to 1.

Definition 8. Let G, G' be dynamic expressions and

$$TS(G) = (S_G, L_G, \mathcal{T}_G, s_G), TS(G') = (S_{G'}, L_{G'}, \mathcal{T}_{G'}, s_{G'})$$

be their transition systems. A mapping $\beta : S_G \to S_{G'}$ is an *isomorphism* between $TS(G)$ and $TS(G')$, denoted by $\beta : TS(G) \simeq TS(G')$, if β is a bijection such that $\beta(s_G) = \beta(s_{G'})$ and $\forall s, \tilde{s} \in S_G\ \forall \Gamma\ s \xrightarrow{\Gamma}_{\mathcal{P}} \tilde{s} \Leftrightarrow \beta(s) \xrightarrow{\Gamma}_{\mathcal{P}} \beta(\tilde{s})$. Two transition systems $TS(G)$ and $TS(G')$ are *isomorphic*, denoted by $TS(G) \simeq TS(G')$, if $\exists \beta : TS(G) \simeq TS(G')$.

Transition systems of static expressions can be defined as well. For $E \in \mathit{RegStatExpr}$ let $TS(E) = TS(\overline{E})$.

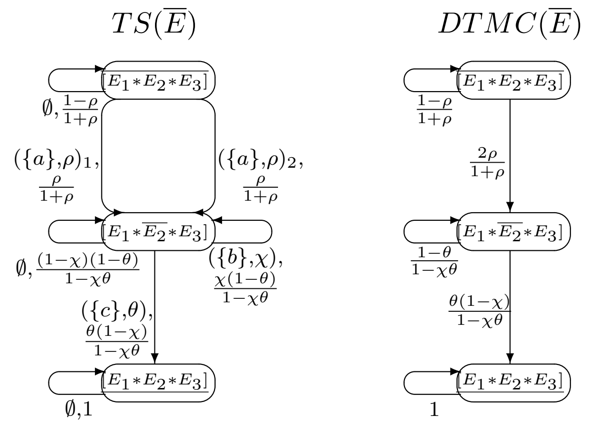
$TS(\overline{E})$
$DTMC(\overline{E})$
$[\overline{E_1 * E_2 * E_3}]$
$\emptyset, \frac{1-\rho}{1+\rho}$
$(\{a\},\rho)_1, \frac{\rho}{1+\rho}$
$(\{a\},\rho)_2, \frac{\rho}{1+\rho}$
$[E_1 * \overline{E_2} * E_3]$
$\emptyset, \frac{(1-\chi)(1-\theta)}{1-\chi\theta}$
$(\{b\},\chi), \frac{\chi(1-\theta)}{1-\chi\theta}$
$(\{c\},\theta), \frac{\theta(1-\chi)}{1-\chi\theta}$
$[E_1 * E_2 * \underline{E_3}]$
$\emptyset,1$
$\frac{1-\rho}{1+\rho}$
$\frac{2\rho}{1+\rho}$
$\frac{1-\theta}{1-\chi\theta}$
$\frac{\theta(1-\chi)}{1-\chi\theta}$
1

Figure 1. The transition system and the underlying DTMC of $\overline{E}$ for $E = [((\{a\},\rho)_1[](\{a\},\rho)_2) * (\{b\},\chi) * (\{c\},\theta)]$

Definition 9. Two dynamic expressions G and G' are *isomorphic with respect to transition systems*, denoted by $G =_{ts} G'$, if $TS(G) \simeq TS(G')$.

Definition 10. Let G be a dynamic expression. The *underlying discrete time Markov chain (DTMC)* of G, denoted by $DTMC(G)$, has the state space $DR(G)$ and transitions $[H]_\simeq \rightarrow_{PM(H,\widetilde{H})} [\widetilde{H}]_\simeq$, if $\exists\Gamma\ [H]_\simeq \xrightarrow{\Gamma} [\widetilde{H}]_\simeq$.

Underlying DTMCs of static expressions can be defined as well. For $E \in RegStatExpr$, let $DTMC(E) = DTMC(\overline{E})$.

Example 1. 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: $E = [((\{a\},\rho)_1[](\{a\},\rho)_2) * (\{b\},\chi) * (\{c\},\theta)]$. In Figure 1 the transition system $TS(\overline{E})$ and the underlying DTMC $DTMC(\overline{E})$ are presented. Note that, for simplicity of the graphical representation, states are depicted by expressions belonging to the corresponding isomorphism classes, and singleton multisets of activities are written without braces.

4. Denotational semantics

In this section, we construct the denotational semantics in terms of a subclass of labeled DTSPNs called discrete time stochastic Petri boxes (dts-boxes).

4.1. Labeled DTSPNs

Now we introduce a class of labeled discrete time stochastic Petri nets.

Definition 11. A *labeled 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 $P_N \cup T_N \neq \emptyset$ and $P_N \cap T_N = \emptyset$;
- $W_N : (P_N \times T_N) \cup (T_N \times P_N) \to I\!N$ is a function describing the *weights of arcs* between places and transitions;
- $\Omega_N : T_N \to (0; 1)$ is the *transition probability* function;
- $L_N : T_N \to Act_\tau$ is the *transition labeling* function assigning labels from a finite set of visible actions *Act* or an invisible action τ to transitions (i.e., $Act_\tau = Act \cup \{\tau\}$);
- $M_N \in I\!N_f^{P_N}$ is the *initial marking.*

A graphical representation of LDTSPNs is like that for standard labeled Petri nets but with probabilities written near the corresponding transitions. If the probabilities are not depicted, they are considered to be of no importance in the corresponding examples. The names of places and transitions are depicted near them when needed. If the names are omitted but used, it is supposed that the places and transitions are numbered from left to right and from top to down.

Let N be an LDTSPN and $t \in T_N$, $U \in I\!N_f^{T_N}$. The *precondition* ${}^\bullet t$ and the *postcondition* $t^\bullet$ of t are the multisets of places defined as $({}^\bullet t)(p) = W_N(p,t)$ and $(t^\bullet)(p) = W_N(t,p)$. The *precondition* ${}^\bullet U$ and the *postcondition* $U^\bullet$ of U are the multisets of places defined as ${}^\bullet U = \sum_{t \in U} {}^\bullet t$ and $U^\bullet = \sum_{t \in U} t^\bullet$.

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 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 ${}^\bullet U \subseteq M$. 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. Thus, we do not allow self-concurrency, i.e., firing of transitions concurrently to themselves. This restriction is introduced because we would like to avoid technical difficulties while calculating probabilities for multisets of transitions.

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 ${}^\bullet U \subseteq M$. The probability that the transitions from U *try to fire* in M is

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

In the case $U=\emptyset$ we define

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

Thus, $PF(U,M)$ could be interpreted as a *joint* probability of independent events. Each such an event is interpreted as trying or not trying to fire of a particular transition from U. When no transitions are enabled in M, we have $PF(\emptyset,M)=1$, since we stay in M in this case.

Let U be a transition set that is enabled in M. Concurrent firing of the transitions from U changes the marking M to $\widetilde{M} = M - {}^\bullet U + U^\bullet$, denoted by $M \xrightarrow{U}_{PT(U,M)} \widetilde{M}$, where the probability of this step is

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

In the case $U=\emptyset$ we have $M=\widetilde{M}$ and

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

Thus, $PT(U,M)$ is the probability that the set U tries to fire *normalized* by the probability to fire for *any* set enabled in M.

We write $M \xrightarrow{U} \widetilde{M}$ if $\exists\mathcal{P}\ M \xrightarrow{U}_{\mathcal{P}} \widetilde{M}$.

Definition 12. Let N be an LDTSPN.

- The *reachability set* of N, denoted by $RS(N)$, is the minimal set of markings such that
  - $M_N \in RS(N)$;
  - if $M\in RS(N)$ and $\exists U\ M \xrightarrow{U} \widetilde{M}$ then $\widetilde{M}\in RS(N)$.
- The *reachability graph* of N, denoted by $RG(N)$, is a directed labeled graph with the set of nodes $RS(N)$ and an arc labeled with $(U,\mathcal{P})$ between nodes M and $\widetilde{M}$ if $M \xrightarrow{U}_{\mathcal{P}} \widetilde{M}$.
- The *underlying discrete time Markov chain (DTMC)* of N, denoted by $DTMC(N)$, has the state space $RS(N)$ and transitions $M \rightarrow_{PM(M,\widetilde{M})} \widetilde{M}$, if $\exists U\ M \xrightarrow{U} \widetilde{M}$, where the transition probability is

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

Thus, $PM(M,\widetilde{M})$ is the probability for *any* transition set to change marking M to $\widetilde{M}$, hence we use summation in the definition.

4.2. Algebra of dts-boxes

Now we propose discrete time stochastic Petri boxes and associated algebraic operations to define a net representation of *dtsPBC* expressions.

Definition 13. A *plain discrete time stochastic Petri box (plain 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 $P_N \cup T_N \neq \emptyset$ and $P_N \cap T_N = \emptyset$;
- $W_N : (P_N \times T_N) \cup (T_N \times P_N) \to I\!\!N$ is a function describing the *weights of arcs* between places and transitions;
- Λ_N is the *place and transition labeling* function such that $\Lambda_N : P_N \to \{\mathsf{e}, \mathsf{i}, \mathsf{x}\}$ (it specifies *entry, internal* and *exit* places, respectively) and $\Lambda_N : T_N \to \mathcal{SL}$ (it associates activities with transitions).

Moreover, $\forall t \in T_N\ {}^\bullet t \neq \emptyset \neq t^\bullet,\ {}^\bullet t \cap t^\bullet = \emptyset$. In addition, if we define the set of *entry* places of N as ${}^\circ N = \{p \in P_N \mid \Lambda_N(p) = \mathsf{e}\}$, and the set of *exit* places of N as $N^\circ = \{p \in P_N \mid \Lambda_N(p) = \mathsf{x}\}$, then the following is required to hold: ${}^\circ N \neq \emptyset \neq N^\circ,\ {}^\bullet({}^\circ N) = \emptyset = (N^\circ)^\bullet$.

A *marked plain dts-box* is a pair (N, M_N), where N is a plain dts-box and $M_N \in I\!\!N_f^{P_N}$ is the *initial marking.* We shall use the following notation: $\overline{N} = (N, {}^\circ N)$ and $\underline{N} = (N, N^\circ)$. Note that 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 Ω_N and L_N are defined as follows: $\forall t \in T_N\ \Omega_N(t) = \Omega(\Lambda_N(t)),\ L_N(t) = \mathcal{L}(\Lambda_N(t))$. In this case, the label τ of silent transitions from the LDTSPN corresponds to the multiaction part $\emptyset$ of activities which label unobservable transitions of the corresponding dts-box. The behaviour of marked dts-boxes follows from the firing rule of LDTSPNs. A plain dts-box N is *safe*, if $\overline{N}$ is so, i.e., $\forall M \in RS(\overline{N})\ M \subseteq P_N$. A plain dts-box N is *clean* if $N^\circ \subseteq M \ \Rightarrow\ M = N^\circ$, i.e., if there are tokens in exit places, then all exit places and only they have tokens.

To define a semantic function that associates a plain dts-box with every static expression of *dtsPBC*, we need to propose the *enumeration* function $Enu : T_N \to I\!\!N^*$. It associates the numbers with transitions of a plain dts-box N in accordance with the enumeration of activities from left to right in the syntax of the underlying static expression. In the case of synchronization, the function associates concatenation of the numbering of the transitions it comes from with the resulting new transition. The transitions resulting from synchronization are considered up to the permutation of their numbering resulting from the applications of the second rule for synchronization to the corresponding expression.

The structure of the plain dts-box corresponding to a static expression is constructed as in PBC, see [2]. I.e., we use simultaneous refinement and relabeling meta-operator (net refinement) in addition to the *operator dts-boxes* corresponding to the algebraic operations of *dtsPBC* and featuring transformational transition relabelings. Thus, the resulting plain dts-boxes are safe and clean. In the definition of denotational semantics, we shall use standard constructions used for PBC. For convenience, we only use slightly different notation: ϱ, Θ and u stand for ρ (relabeling), Ω (operator box) and v (transition name) from PBC setting, respectively.

The relabeling relations $\varrho \subseteq I\!N_f^{\mathcal{SL}} \times \mathcal{SL}$ are defined as follows:

- $\varrho_{id} = \{(\{(\alpha, \rho)\}, (\alpha, \rho) \mid (\alpha, \rho) \in \mathcal{SL}\}$ is the *identity* relabeling keeping the interface as it is;
- $\varrho_{[f]} = \{(\{(\alpha, \rho)\}, (f(\alpha), \rho) \mid (\alpha, \rho) \in \mathcal{SL}\}$;
- $\varrho_{\mathsf{rs}\ a} = \{(\{(\alpha, \rho)\}, (\alpha, \rho) \mid (\alpha, \rho) \in \mathcal{SL},\ a, \hat{a} \notin \mathcal{A}(\alpha)\}$;
- $\varrho_{\mathsf{sy}\ a}$ is the least relabeling relation contained in ϱ_{id} such that if $(\Gamma, \{(\alpha + \{a\}, \rho)\} \in \varrho_{\mathsf{sy}\ a}$ and $(\Delta, \{(\beta + \{\hat{a}\}, \chi)\} \in \varrho_{\mathsf{sy}\ a}$ then $(\Gamma + \Delta, \{(\alpha + \beta, \rho \cdot \chi)\} \in \varrho_{\mathsf{sy}\ a}$.

The plain and operator dts-boxes are presented in Figure 2. The symbol i is usually omitted.

Now we define the enumeration function Enu for every operator of *dtsPBC*. Let $Box_{dts}(E) = (P_E, T_E, W_E, \Omega_E, L_E)$ be the plain dts-box corresponding to a static expression E, and Enu_E be the enumeration function for T_E.

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

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

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

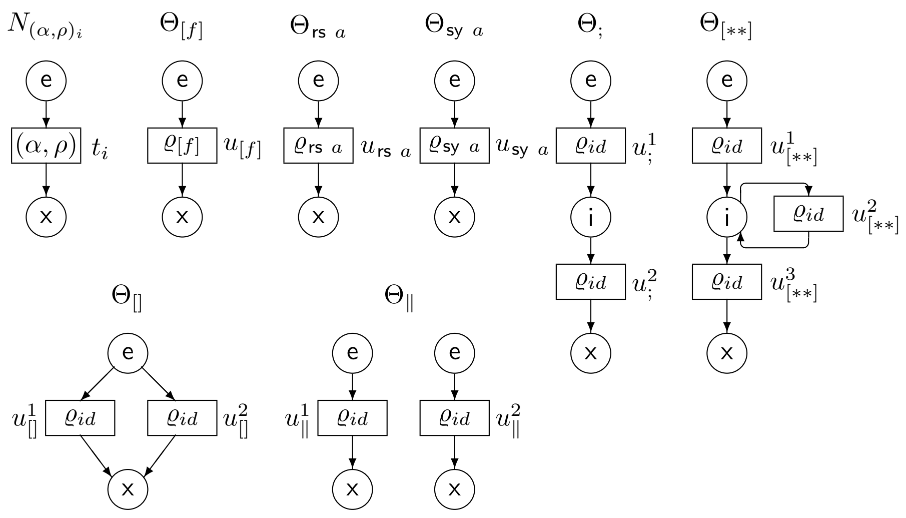

Figure 2. The plain and operator dts-boxes

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

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

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

- $Box_{dts}(E\ \mathsf{sy}\ a) = \Theta_{\mathsf{sy}\ a}(Box_{dts}(E))$. Note that $\forall v, w \in T_E$ such that $L_E(v) = \alpha + \{a\},\ L_E(w) = \beta + \{\hat{a}\}$, the new transition t resulting from synchronization of v and w has the label $L(t) = \alpha + \beta$, probability $\Omega(t) = \Omega_E(v) \cdot \Omega_E(w)$ and enumeration $Enu(t) = Enu_E(v) \cdot Enu_E(w)$. Thus, the enumeration is defined as

$$Enu(t) = \begin{cases} Enu_E(t), & t \in T_E; \\ Enu_E(v) \cdot Enu_E(w), & t \text{ results from synchronization} \\ & \text{of } v \text{ and } w. \end{cases}$$

To avoid introducing redundant transitions generated by synchronizing the same transition set in a different order, we only consider a single one of them in the plain dts-box.

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

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

Now we can formally define the denotational semantics as a homomorphism.

Definition 14. 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:

1. $Box_{dts}((\alpha,\rho)_i) = N_{(\alpha,\rho)_i}$;
2. $Box_{dts}(E \circ F) = \Theta_\circ(Box_{dts}(E), Box_{dts}(F)),\ \circ \in \{;, [], \|\}$;
3. $Box_{dts}(E[f]) = \Theta_{[f]}(Box_{dts}(E))$;
4. $Box_{dts}(E \circ a) = \Theta_{\circ a}(Box_{dts}(E)),\ \circ \in \{\mathsf{rs}, \mathsf{sy}\}$;
5. $Box_{dts}([E * F * K]) = \Theta_{[**]}(Box_{dts}(E), Box_{dts}(F), Box_{dts}(K))$.

The dts-boxes of dynamic expressions can be defined as well. For $E \in RegStatExpr$, let $Box_{dts}(\overline{E}) = \overline{Box_{dts}(E)}$ and $Box_{dts}(\underline{E}) = \underline{Box_{dts}(E)}$. Note that any dynamic expression can be decomposed into overlined or underlined static expressions or those without overlines and underlines, and the definition of dts-boxes is compositional.

Isomorphism is a coincidence of systems up to renaming of their components or states. Let $\simeq$ denote isomorphism between transition systems or DTMCs and reachability graphs. Note that in this case, the names of transitions of the dts-box corresponding to a static expression could be identified with the enumerated activities of the latter.

Theorem 1. *For any static expression E*

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

Proof. As for the qualitative (functional) behaviour, we have the same isomorphism as in *PBC*. The quantitative behaviour is the same by the following reasons. First, the activities of a static expression have probability parts coinciding with probabilities of the transitions belonging to the corresponding plain dts-box. Second, in both semantics, conflicts are resolved via the same probability functions.

Proposition 1. *For any static expression E*

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

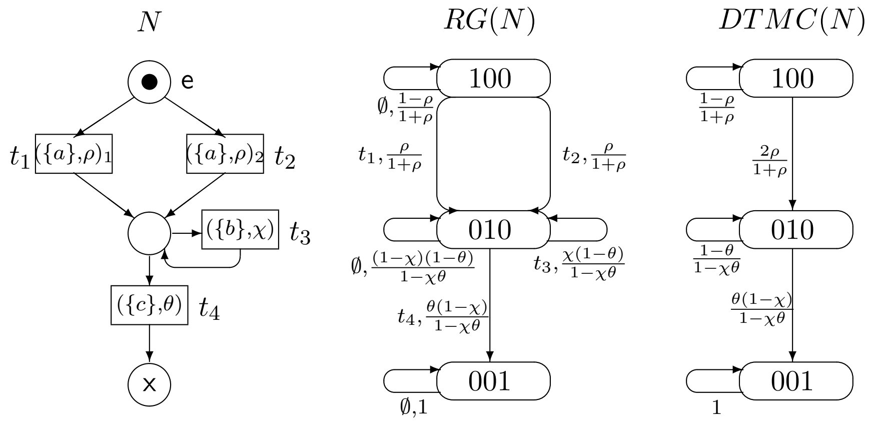
N
RG(N)
DTMC(N)
e
t_1 $(\{a\},\rho)_1$
$(\{a\},\rho)_2$ t_2
$(\{b\},\chi)$ t_3
$(\{c\},\theta)$ t_4
x
100
010
001
$\emptyset,\frac{1-\rho}{1+\rho}$
$t_1,\frac{\rho}{1+\rho}$
$t_2,\frac{\rho}{1+\rho}$
$\emptyset,\frac{(1-\chi)(1-\theta)}{1-\chi\theta}$
$t_3,\frac{\chi(1-\theta)}{1-\chi\theta}$
$t_4,\frac{\theta(1-\chi)}{1-\chi\theta}$
$\emptyset,1$
$\frac{1-\rho}{1+\rho}$
$\frac{2\rho}{1+\rho}$
$\frac{1-\theta}{1-\chi\theta}$
$\frac{\theta(1-\chi)}{1-\chi\theta}$
1

Figure 3. 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

Proof. By Theorem 1 and definitions of the underlying DTMCs for dynamic expressions and LDTSPNs, since transition probabilities of the associated DTMCs are the sums of those belonging to transition systems or reachability graphs.

Example 2. Let $E = [((\{a\},\rho)_1[](\{a\},\rho)_2)*(\{b\},\chi)*(\{c\},\theta)]$, i.e., it is from Example 1. In Figure 3 the marked dts-box $N = Box_{dts}(\overline{E})$, its reachability graph $RG(N)$ and the underlying DTMC $DTMC(N)$ are presented. It is easy to see that $TS(\overline{E})$ and $RG(N)$ are isomorphic, as well as $DTMC(\overline{E})$ and $DTMC(N)$.

5. Conclusion

In this paper, we have proposed a discrete time stochastic extension of a finite part of *PBC* enriched with iteration called *dtsPBC*. The new calculus has the concurrent step operational semantics based on transition systems and the denotational semantics in terms of a subclass of LDTSPNs. Consistency of operational and denotational semantics was established.

Future work consists in defining the algebraic equivalences of *dtsPBC* which abstract from the silent activities, i.e., those with empty multiaction part. As a result, we shall have the algebraic analogues of the net probabilistic equivalences from [7]. The equivalence relations will allow one to identify stochastic processes with similar behaviour which are differentiated by too strict notion of the semantic equivalence. Then, we can try to construct a congruence relation based on some algebraic equivalence we have proposed. Moreover, we plan to extend *dtsPBC* with recursion.

References

[1] Best E., Devillers R., Hall J.G. The box calculus: a new causal algebra with multi-label communication // Lect. Notes Comput. Sci. — 1992. — Vol. 609. — P. 21–69.

[2] Best E., Devillers R., Koutny M. Petri net algebra // EATCS Monographs on Theor. Comput. Sci. — 2001. Springer Verlag. — 378 p.

[3] Best E., Devillers R., Koutny M. The box algebra = Petri nets + process expressions // Information and Computation. — 2002. — Vol. 178. — P. 44–100.

[4] Bernardo M., Gorrieri R. A tutorial on EMPA: a theory of concurrent processes with nondeterminism, priorities, probabilities and time // Theor. Comput. Sci. — 1998. — Vol. 202. — P. 1–54.

[5] Buchholz P., Kemper P. Quantifying the dynamic behavior of process algebras // Lect. Notes Comput. Sci. — 2001. — Vol. 2165. — P. 184–199.

[6] Buchholz P., Tarasyuk I.V. A class of stochastic Petri nets with step semantics and related equivalence notions. — Technische Berichte. — Dresden, 2000. — 18 p. — (Fakultät Informatik. Technische Universität; Vol. TUD-FI00-12). — `ftp://ftp.inf.tu-dresden.de/pub/berichte/tud00-12.ps.gz`

[7] Buchholz P., Tarasyuk I.V. Net and algebraic approaches to probablistic modeling // Joint Bull. Novosibirsk Comp. Center and Institute of Informatics Systems. Ser. Computer Science. — Novosibirsk, 2001. — Iss. 15. — P. 31–64. — `http://www.iis.nsk.su/persons/itar/spnpancc.pdf`

[8] Buchholz P., Tarasyuk I.V. Equivalences for stochastic Petri nets and stochastic process algebras // Vestnik, Quartal J. of Novosibirsk State University. Ser.: Mathematics, Mechanics and Informatics. — 2006. — Vol. 6, N 1. — P. 14–42. — `http://www.iis.nsk.su/persons/itar/vestnik.pdf`

[9] Buchholz P. Markovian process algebra: composition and equivalence // Proc. of 2^{nd} Workshop on Process Algebras and Performance Modelling, Arbeitsberichte des IMMD / Ed. by U. Herzog, M. Rettelbach. — 1994. — Vol. 27. — P. 11–30.

[10] Buchholz P. A notion of equivalence for stochastic Petri nets // Lect. Notes Comput. Sci. — 1995. — Vol. 935. — P. 161–180.

[11] Buchholz P. Iterative decomposition and aggregation of labeled GSPNs // Lect. Notes Comput. Sci. — 1998. — Vol. 1420. — P. 226–245.

[12] Donatelli S., Ribaudo M., Hillston J. A comparison of perfomance evaluation process algebra and generalized stochastic Petri nets // Proc. of 6^{th} Internat. Workshop on Petri Nets and Performance Models, Durham, USA. — IEEE Computer Society Press, 1995. — P. 158–168.

[13] Florin G., Natkin S. Les reseaux de Petri stochastiques // Technique et Science Informatique. — 1985. — Vol. 4, N 1.

[14] Hillston J. A compositional approach to performance modelling. — Cambridge University Press, 1996.

[15] Hermanns H., Rettelbach M. Syntax, semantics, equivalences and axioms for MTIPP // Proc. of 2^{nd} Workshop on Process Algebras and Performance Modelling. — 1994. — Vol. 27. — P. 71–88.

[16] Koutny M., Best E. Operational and denotational semantics for the box algebra // Theor. Comput. Sci. — 1999. — Vol. 211, N 1–2. — P. 1–83. — `http://parsys.informatik.uni-oldenburg.de/~best/publications/tcs.ps.gz`

[17] Kotov V.E., Cherkasova L.A. On structural properties of generalized processes // Lect. Notes Comput. Sci. — 1985. — Vol. 188. — P. 288–306.

[18] Kotov V.E. An algebra for parallelism based on Petri nets // Lect. Notes Comput. Sci. — 1978. — Vol. 64. — P. 39–55.

[19] Milner R.A.J. Communication and concurrency. — NY: Prentice-Hall International, 1989.

[20] Molloy M. Performance analysis using stochastic Petri nets // IEEE Transactions on Software Engineering. — 1982. — Vol. 31, N 9. — P. 913–917.

[21] Molloy M. Discrete time stochastic Petri nets // IEEE Transactions on Software Engineering.— 1985. — Vol. 11, N 4. — P. 417–423.

[22] Macià H.S., Valero V.R., Cuartero F.G. A congruence relation in finite sPBC. — Albacete, 2002. — 34 p. — (Tech. Rep. / Department of Computer Science, University of Castilla-La Mancha; Vol. DIAB-02-01-31). — `http://www.info-ab.uclm.es/retics/publications/2002/tr020131.ps`

[23] Macià H.S., Valero V.R., Cuartero F.G. Defining equivalence relations in sPBC // Proceedings of 1^{st} International Conference on the Principles of Software Engineering - 04 (PriSE'04). — November 2004. — P. 195–205, Buenos Aires, Argentina. — `http://www.info-ab.uclm.es/retics/publications/2004/prise04.pdf`

[24] Macià H.S., Valero V.R., Cazorla D.L., Cuartero F.G. Introducing the iteration in sPBC. // Lect. Notes Comp. Sci. — 2004. — Vol. 3235. — P. 292–308. — `http://www.info-ab.uclm.es/retics/publications/2004/forte04.pdf`

[25] Macià H.S., Valero V.R., Cuartero F.G., Ruiz M.C.D. sPBC with immediate multiactions (Draft paper). — 27 p.

[26] Macià H.S., Valero V.R., de Frutos D.E. sPBC: a Markovian extension of finite Petri box calculus // Proc. of 9^{th} IEEE Internat. Workshop on Petri Nets and Performance Models - 01 (PNPM'01). — Aachen: IEEE Computer Society Press, 2001. — P. 207–216. — `http://www.info-ab.uclm.es/retics/publications/2001/pnpm01.ps`

[27] Ribaudo M. Stochastic Petri net semantics for stochastic process algebra // Proc. of 6th Internat. Workshop on Petri Nets and Performance Models. — Durham: IEEE Computer Society Press, 1995. — P. 148–157.

[28] Tarasyuk I.V. Discrete time stochastic Petri box calculus. — Carl von Ossietzky Universität Oldenburg, 2005. — 25 p. — (Berichte aus dem Department für Informatik; Vol. 3/05). —
`http://www.iis.nsk.su/persons/itar/dtspbcib.pdf`