Bull. Nov. Comp. Center, Comp. Science, 32 (2011), 127–155

Performance evaluation of the generalized shared memory system in dtsPBC*

I. V. Tarasyuk

Abstract. The algebra dtsPBC is a discrete time stochastic extension of finite Petri box calculus (PBC) enriched with iteration. In this paper, a method of modeling and performance evaluation of concurrent systems in dtsPBC is outlined based on the stationary behaviour analysis. The method is then applied to the generalized shared memory system with a variable probability of activities.
Keywords: *stochastic process algebra, Petri box calculus, discrete time, iteration, stationary behaviour, performance evaluation, shared memory system, variable probability.*

1. Introduction

Algebraic process calculi are a recognized formal model for specification of computing systems and analysis of their behaviour. In such process algebras (PAs), systems and processes are specified by formulas, and verification of their properties is accomplished at a syntactic level via equivalences, axioms and inference rules. In the last decades, stochastic extensions of PAs were proposed. Stochastic process algebras (SPAs) do not just specify actions which can happen as usual process algebras (qualitative features), but they associate some quantitative parameters with actions (quantitative characteristics). The well-known SPAs are MTIPP [6], PEPA [5] and EMPA [4].

Petri box calculus (PBC) [2] is a flexible and expressive process algebra developed as a tool for specification of Petri nets structure and their interrelations. Its goal was also to propose a compositional semantics for high level constructs of concurrent programming languages in terms of elementary Petri nets. PBC has a step operational semantics in terms of labeled transition systems. Its denotational semantics was proposed in terms of a subclass of Petri nets (PNs) equipped with interface and considered up to isomorphism called Petri boxes.

A stochastic extension of PBC, stochastic Petri box calculus (sPBC), was proposed in [11]. Only a finite part of PBC was used for the stochastic enrichment, i.e., sPBC has neither refinement nor recursion nor iteration operations. The calculus has an interleaving operational semantics in terms of labeled transition systems. Its denotational semantics was defined in

*Partially supported by the Deutsche Forschungsgemeinschaft Grant 436 RUS 113/1002/01, and the Russian Foundation for Basic Research Grant 09-01-91334.

terms of a subclass of labeled continuous time stochastic PNs (LCTSPNs) called s-boxes. In [7], a computing system with n parallel processes and a critical section, as well as the producer/consumer system with a producer, a consumer and a buffer of capacity 1 or n, moreover, the alternating bit protocol with an emitter, a receptor and 2 channels, were described within sPBC. In [8, 9], iteration was added to sPBC and the producer/consumer system with a buffer of capacity n was specified within the calculus. In [10], special multiactions with zero time delay were added to sPBC and a manufacturing system with 3 machines and an assembler, as well as the AUY-protocol with a sender, receiver and 2 channels, were modeled. The mentioned example systems had an interleaving semantics, and no performance indices were calculated for them.

In [12, 14], a discrete time stochastic extension dtsPBC of finite PBC was presented. A step operational semantics of dtsPBC was constructed via labeled probabilistic transition systems. Its denotational semantics was defined with dts-boxes, a subclass of labeled discrete time stochastic PNs (LDTSPNs). The probabilistic equivalences and their interrelations were studied. In [13, 15], the iteration operator was added to dtsPBC.

Since dtsPBC has a discrete time semantics and geometrically distributed delays in the process states, unlike sPBC with continuous time semantics and exponentially distributed delays, the calculi apply two different approaches to the stochastic extension of PBC, in spite of some similarity of their syntax and semantics inherited from PBC. The main advantage of dtsPBC is that concurrency is treated naturally, like in PBC, whereas in sPBC parallelism is simulated by interleaving, obliging one to collect the information on causal independence of activities before constructing the semantics. Thus, parallelism is preserved in the semantics of all example systems considered as the case studies within dtsPBC.

In this paper, dtsPBC with iteration is a basic model. First, we present syntax of the calculus. Second, we describe its operational semantics in terms of labeled transition systems and denotational semantics based on LDTSPNs. We improve the operational semantics from [13, 15] by moving the empty loop rule (corresponding to one time unit stay in the current process state) from the inaction rules, consuming no time, to the action rules, requiring one time unit to apply. Then, we propose step stochastic bisimulation equivalence used to reduce transition systems of expressions and their underlying DTMCs. Further, the stationary behaviour of infinite stochastic processes of dtsPBC is described and several performance indices are defined based on the steady-state probabilities. We prove that the mentioned equivalence preserves the stationary behaviour. Finally, we present a case study of the generalized shared memory system explaining how to model and analyze performance of concurrent systems within dtsPBC, as well as in which way to reduce the systems while preserving their perfor-

mance indices, thus, making simpler their performance evaluation. We call the shared memory system generalized, since it is specified by an expression with a variable probability of activities. It is interpreted as a parameter of the performance indices of the system. The resulting performance index functions are then analyzed with a goal of performance optimization.

The paper is organized as follows. The syntax of dtsPBC extended with iteration is presented in Section 2. Section 3 describes its operational semantics and Section 4 presents its denotational semantics. In Section 5, step stochastic bisimulation equivalence is defined and applied to reduce transition systems and Markov chains. The method of performance evaluation of dtsPBC processes is outlined in Section 6. In Section 7, performance of the generalized shared memory system is analyzed. The concluding Section 8 summarizes the results obtained and outlines research perspectives.

2. Syntax

In this section, we propose the syntax of discrete time stochastic PBC.

We denote a *set of all finite multisets* over X by $\mathbb{N}_f^X$. Let $Act = \{a, b, \ldots\}$ be a set of *elementary actions*. Then $\widehat{Act} = \{\hat{a}, \hat{b}, \ldots\}$ is a set of *conjugated actions (conjugates)* such that $a \neq \hat{a}$ and $\hat{\hat{a}} = a$. Let $\mathcal{A} = Act \cup \widehat{Act}$ be a set of *all actions*, and $\mathcal{L} = \mathbb{N}_f^{\mathcal{A}}$ be a set of *all multiactions*. Note that $\emptyset \in \mathcal{L}$, this corresponds to 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. The multiaction probabilities are used to calculate probabilities of state changes (steps) at discrete time moments. Let $\mathcal{SL}$ be a 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 $\mathcal{A}(\alpha, \rho) = \mathcal{A}(\alpha)$. The *alphabet* of $\Gamma \in \mathbb{N}_f^{\mathcal{SL}}$ is defined as $\mathcal{A}(\Gamma) = \cup_{(\alpha,\rho)\in\Gamma}\mathcal{A}(\alpha)$. For $(\alpha, \rho) \in \mathcal{SL}$, we define its *multiaction part* as $\mathcal{L}(\alpha, \rho) = \alpha$ and its *probability part* as $\Omega(\alpha, \rho) = \rho$. The *multiaction part* of $\Gamma \in \mathbb{N}_f^{\mathcal{SL}}$ is defined as $\mathcal{L}(\Gamma) = \sum_{(\alpha,\rho)\in\Gamma} \alpha$.

Activities are combined into formulas by the next operations: *sequential execution* ;, *choice* [], *parallelism* $\|$, *relabeling* $[f]$ of actions, *restriction* rs over a single action, *synchronization* sy on an action and its conjugate and *iteration* $[**]$ with three arguments: initialization, body and termination.

Sequential execution and choice have a standard interpretation, like in other process algebras, but parallelism does not include synchronization, unlike the corresponding operation in the standard process calculi.

Relabeling functions $f : \mathcal{A} \to \mathcal{A}$ are bijections preserving conjugates, i.e., $\forall x \in \mathcal{A}\ f(\hat{x}) = \widehat{f(x)}$. Relabeling is extended to multiactions in a usual way:

for $\alpha \in \mathcal{L}$ we define $f(\alpha) = \sum_{x \in \alpha} f(x)$. Relabeling is also extended to the multisets of activities: for $\Gamma \in I\!N_f^{\mathcal{SL}}$ we define $f(\Gamma) = \sum_{(\alpha,\rho)\in\Gamma}(f(\alpha), \rho)$.

Restriction over an action a means that, for a given expression, any process behaviour containing a or its conjugate $\hat{a}$ is not allowed.

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

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

Static expressions specify the structure of processes. The expressions correspond to unmarked LDTSPNs (which are marked by definition).

Definition 1. Let $(\alpha, \rho) \in \mathcal{SL}$, $a \in Act$. A *static expression* of dtsPBC is

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

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

To avoid inconsistency of the iteration operator, we do not allow any concurrency in the highest level of the second argument of iteration. This is not a severe restriction, since we can prefix parallel expressions by an activity with the empty multiaction. The mentioned inconsistency can result in non-safe nets [3].

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

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

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

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

Definition 3. Let $E \in StatExpr$, $a \in Act$. A *dynamic expression* of dtsPBC is

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

DynExpr denotes the set of *all dynamic expressions* of dtsPBC.
A dynamic expression is *regular* if its underlying static expression is regular. *RegDynExpr* denotes the set of *all regular dynamic expressions* of dtsPBC.

3. Operational semantics

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

3.1. Inaction rules

Inaction rules for dynamic expressions describe their structural transformations which do not change the states of the specified processes. As we will see, the application of an inaction rule to a dynamic expression does not lead to any discrete time step in the corresponding LDTSPN, hence, no transitions are fired and its current marking remains unchanged. Thus, an application of every inaction rule does not require any discrete time delay, i.e., the dynamic expression transformation described by the rule is accomplished instantly.

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

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

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

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

A regular dynamic expression G is *operative* if no inaction rule can be applied to it. *OpRegDynExpr* denotes the set of *all operative regular dynamic expressions* of dtsPBC. Any regular dynamic expression can be transformed into a (not necessarily unique) operative one by using the inaction

Table 2. Inaction rules for arbitrary regular dynamic expressions

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

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

Definition 4. Let $\approx\ =(\Rightarrow\cup\Leftarrow)^*$ be a structural equivalence of 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

Action rules describe dynamic expression transformations due to 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 are fired and the current marking is changed, unless there is a self-loop produced by the iterative execution of a non-empty multiset (which, additionally, should be one-element, i.e., a 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 execution of the empty multiset of activities at a discrete time step. The rule reflects a non-zero probability to stay 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 are fired and the current marking is not changed. This is a new rule that has no prototype among inaction rules of PBC, since it represents a time delay. The PBC rule $G\stackrel{\emptyset}{\rightarrow}G$ from [3] in our setting would correspond to the rule $G\Rightarrow G$ describing the stay in the current state when no time elapses, but it is not needed to transform dynamic expressions into operative ones.

Thus, an application of every action rule or the empty loop rule requires one discrete time unit delay, i.e., 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.

In Table 3, we define the action and empty loop rules. In this table, $(\alpha,\rho),\ (\beta,\chi) \in \mathcal{SL},\ E,F \in RegStatExpr,\ G,H \in OpRegDynExpr,\ \widetilde{G},\widetilde{H} \in RegDynExpr$ and $a \in Act$. Moreover, $\Gamma,\Delta \in I\!N_f^{\mathcal{SL}} \setminus \{\emptyset\}$ and $\Gamma' \in I\!N_f^{\mathcal{SL}}$.

Table 3. Action and empty loop rules

El $G \xrightarrow{\emptyset} G$	**B** $\dfrac{}{\underline{(\alpha,\rho)} \xrightarrow{\{(\alpha,\rho)\}} (\alpha,\rho)}$	**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** $\dfrac{G \xrightarrow{\Gamma} \widetilde{G},\ a,\hat{a}\notin\mathcal{A}(\Gamma)}{G\ \mathsf{rs}\ a \xrightarrow{\Gamma} \widetilde{G}\ \mathsf{rs}\ a}$
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 $\dfrac{G \xrightarrow{\Gamma} \widetilde{G}}{G\ \mathsf{sy}\ a \xrightarrow{\Gamma} \widetilde{G}\ \mathsf{sy}\ a}$	**Sy2** $\dfrac{G\ \mathsf{sy}\ a \xrightarrow{\Gamma'+\{(\alpha,\rho)\}+\{(\beta,\chi)\}} \widetilde{G}\ \mathsf{sy}\ a,\ a\in\alpha,\ \hat{a}\in\beta}{G\ \mathsf{sy}\ a \xrightarrow{\Gamma'+\{(\alpha\oplus_a\beta,\rho\cdot\chi)\}} \widetilde{G}\ \mathsf{sy}\ a}$	

In the rule **Sy2**, we multiply the probabilities of synchronized multiactions, this corresponds somehow to the probability of the event intersection. We do not allow self-synchronization, i.e., synchronization of an activity with itself. The purpose is to avoid many technical difficulties, see [3].

3.3. Transition systems

Now we define labeled probabilistic transition systems associated with dynamic expressions and used to define their operational semantics.

Note that 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 synchronization will be annotated with concatenation of numberings of the activities they come from, hence, the numbering should have a tree structure to reflect the effect of multiple synchronizations. The numbering below encodes a binary tree with the leaves labeled by natural numbers.

Definition 5. Let $\iota \in I\!N$. The *numbering* of expressions is $\iota ::= \ \iota \mid (\iota)(\iota)$.

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

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

$Cont(\iota) = \begin{cases} \{\iota\}, & \iota \in I\!N; \\ Cont(\iota_1) \cup Cont(\iota_2), & \iota = (\iota_1)(\iota_2). \end{cases}$

After we apply the enumeration, the multisets of activities from the expressions become the proper sets. In the following, we suppose that the identical activities are enumerated when it is necessary to avoid ambiguity. This enumeration is considered to be implicit.

Definition 6. Let G be a dynamic expression. Then $[G]_{\approx} = \{H \mid G \approx H\}$ is the equivalence class of G w.r.t. the structural equivalence. The *derivation set* $DR(G)$ of a dynamic expression G is the minimal set such that $[G]_{\approx} \in DR(G)$ or, if $[H]_{\approx} \in DR(G)$ and $\exists \Gamma\ H \stackrel{\Gamma}{\to} \widetilde{H}$, then $[\widetilde{H}]_{\approx} \in DR(G)$.

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

The set of *all multisets of activities executable in* s is defined as $Exec(s) = \{\Gamma \mid \exists H \in s\ \exists \widetilde{H}\ H \stackrel{\Gamma}{\to} \widetilde{H}\}$.

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

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

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

The definition of $PF(\Gamma, s)$ (and those of other probability functions) is based on the enumeration of activities which is considered implicit.

The *probability to execute the multiset of activities* $\Gamma \in Exec(s)$ *in* s is

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

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

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

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 $\mathcal{T}_G = \{(s, (\Gamma, PT(\Gamma, s)), \tilde{s}) \mid s \in DR(G), \exists H \in s\ \exists \widetilde{H} \in \tilde{s}\ H \stackrel{\Gamma}{\to} \widetilde{H}\}$;
- the *initial state* is $s_G = [G]_{\approx}$.

The transition system $TS(G)$ associated with a dynamic expression G describes all steps that happen at discrete time moments with some (one-step) probability and consist of multisets of activities. Every step happens instantly after one discrete time unit delay. The step can change the current state to another one. The states are the structural equivalence classes of dynamic expressions obtained by application of action rules starting from the expressions belonging to $[G]_{\approx}$. A transition $(s, (\Gamma, \mathcal{P}), \tilde{s}) \in \mathcal{T}_G$ will be written as $s \xrightarrow{\Gamma}_{\mathcal{P}} \tilde{s}$. The interpretation is: the probability to change the state s to $\tilde{s}$ in the result of executing Γ is $\mathcal{P}$.

Note that Γ can be the empty multiset, and its execution does not change the current state (i.e., the equivalence class), since we have a loop transition $s \xrightarrow{\emptyset}_{\mathcal{P}} s$ from a state s to itself in the result of executing the empty multiset. This corresponds to 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 nonzero probabilities. This follows from the definition of $PF(\emptyset, s)$ and the fact that multiaction probabilities cannot be equal to 1 as they belong to the interval $(0; 1)$. The step probabilities belong to the interval $(0; 1]$. The step probability is 1 in the case when we cannot leave a state s, hence, 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 $\exists \mathcal{P}\ s \xrightarrow{\Gamma}_{\mathcal{P}} \tilde{s}$ and $s \to \tilde{s}$ if $\exists \Gamma\ s \xrightarrow{\Gamma} \tilde{s}$.

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

Definition 8. 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 the transitions $s \to_{\mathcal{P}} \tilde{s}$, if $s \to \tilde{s}$ and $\mathcal{P} = PM(s, \tilde{s})$.

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. One can interpret staying in a state in the next discrete time moment as a failure and leaving it as a success of some trial series. It is easy to see that the random variables are geometrically distributed, since the probability to stay in the state $s \in DR(G)$ for $k-1$ time moments and leave it at the moment $k \geq 1$ 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 the elements $SJ(s),\ s \in DR(G)$. The *sojourn time variance in the state* s is $VAR(s) = \frac{1}{(1-PM(s,s))^2}$. The *sojourn time variance vector* of G, denoted by VAR, has the elements $VAR(s),\ s \in DR(G)$.

4. Denotational semantics

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

Definition 9. 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*, such that $P_N \cup T_N \neq \emptyset$ and $P_N \cap T_N = \emptyset$;
- $W_N : (P_N \times T_N) \cup (T_N \times P_N) \to \mathbb{N}$ is a function providing the *weights of arcs* between places and transitions;
- Λ_N is the *place and transition labeling* function such that
  - $\Lambda_N|_{P_N} : P_N \to \{\mathsf{e}, \mathsf{i}, \mathsf{x}\}$ (specifies the *entry, internal, exit* places);
  - $\Lambda_N|_{T_N} : T_N \to \{\varrho \mid \varrho \subseteq \mathbb{N}_f^{\mathcal{SL}} \times \mathcal{SL}\}$ (it associates transitions with the *relabeling relations* on activities).

Let $t \in T_N$, $U \in \mathbb{N}_f^{T_N}$. The *precondition* ${}^\bullet t$ and the *postcondition* $t^\bullet$ of t are the multisets of places defined 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 $U^\bullet = \sum_{t \in U} t^\bullet$. We require that $\forall t \in T_N \ {}^\bullet t \neq \emptyset \neq t^\bullet$. In addition, for the set of *entry* places of N defined as ${}^\circ N = \{p \in P_N \mid \Lambda_N(p) = \mathsf{e}\}$ and the set of *exit* places of N defined as $N^\circ = \{p \in P_N \mid \Lambda_N(p) = \mathsf{x}\}$, it holds: ${}^\circ N \neq \emptyset \neq N^\circ$, ${}^\bullet({}^\circ N) = \emptyset = (N^\circ)^\bullet$.

A dts-box is *plain* if $\forall t \in T_N \ \Lambda_N(t) \in \mathcal{SL}$, i.e., $\Lambda_N(t)$ is a constant relabeling that will be defined later. In case of the constant relabeling, the shorthand notation (by an activity) for $\Lambda_N(t)$ will be used. 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 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: $\forall t \in T_N \ \Omega_N(t) = \Omega(\Lambda_N(t))$ and $L_N(t) = \mathcal{L}(\Lambda_N(t))$. Behaviour of the marked dts-boxes follows from the firing rule of LDTSPNs.

To define a semantic function that associates a plain dts-box with every static expression of dtsPBC, we shall propose the *enumeration* function $Enu : T_N \to Num$ which associates numberings with transitions of the plain dts-box N according to those of activities. In the case of synchronization, the function associates concatenation of the parenthesized numberings of the synchronized transitions with a resulting new transition.

The structure of the plain dts-box corresponding to a static expression is constructed like in PBC, see [3], i.e., we use a simultaneous refinement and

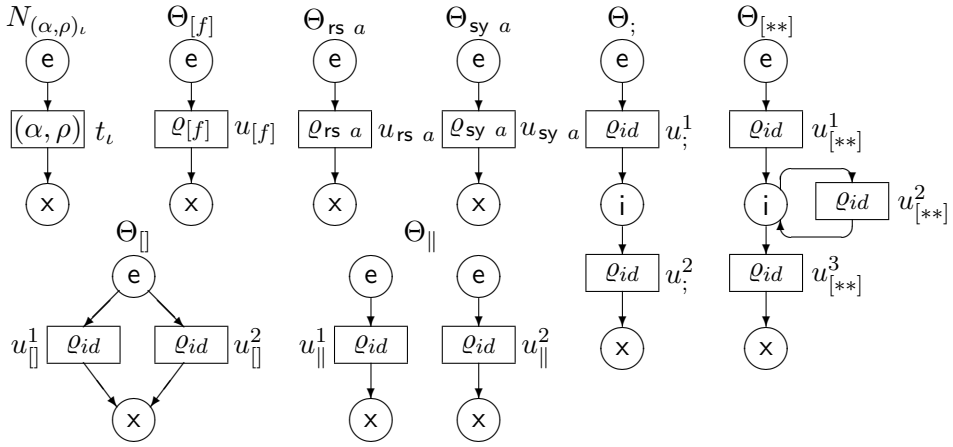

Figure 1. The plain and operator dts-boxes

relabeling meta-operator (net refinement) in addition to the *operator dts-boxes* corresponding to the algebraic operations of dtsPBC and featuring transformational transition relabelings. In the definition of the denotational semantics, we shall apply standard constructions used for PBC. Let Θ denote an *operator box* and u denote a *transition name* from PBC setting.

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;
- $\varrho_{(\alpha,\rho)} = \{(\emptyset,(\alpha,\rho))\}$ is the *constant relabeling* identical to $(\alpha,\rho) \in \mathcal{SL}$;
- $\varrho_{[f]} = \{(\{(\alpha,\rho)\},(f(\alpha),\rho)) \mid (\alpha,\rho) \in \mathcal{SL}\}$;
- $\varrho_{\mathsf{rs}\ a} = \{(\{(\alpha,\rho)\},(\alpha,\rho)) \mid (\alpha,\rho) \in \mathcal{SL},\ a,\hat{a} \notin \alpha\}$;
- $\varrho_{\mathsf{sy}\ a}$ is the least relabeling relation contained in ϱ_{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 presented in Figure 1. Note that the symbol i is usually omitted.

Now we define the enumeration function *Enu* for every operator of dtsPBC. Let $Box_{dts}(E) = (P_E, T_E, W_E, \Lambda_E)$ be the plain dts-box corresponding to a static expression E, and Enu_E be the enumeration function for T_E. We shall use the analogous notation for static expressions F and K.

- $Box_{dts}(E \circ F) = \Theta_\circ(Box_{dts}(E), Box_{dts}(F)),\ \circ \in \{;,[],\|\}$. Since we do not introduce any new transitions, we preserve the initial numbering: $Enu(t) = \begin{cases} Enu_E(t), & t \in T_E; \\ Enu_F(t), & t \in T_F. \end{cases}$

- $Box_{dts}(E[f]) = \Theta_{[f]}(Box_{dts}(E))$. Since we only 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 transitions labeled with multiactions containing a or $\hat{a}$, this does not change the numbering of the remaining transitions: $Enu(t) = Enu_E(t),\ t \in T_E$, $a, \hat{a} \notin \mathcal{L}(\Lambda_E(t))$.
- $Box_{dts}(E\ \mathsf{sy}\ a) = \Theta_{\mathsf{sy}\ a}(Box_{dts}(E))$. Note that $\forall v, w \in T_E$ such that $\Lambda_E(v) = (\alpha, \rho),\ \Lambda_E(w) = (\beta, \chi)$ and $a \in \alpha,\ \hat{a} \in \beta$, the new transition t resulting from synchronization of v and w has the label $\Lambda(t) = (\alpha \oplus_a \beta, \rho \cdot \chi)$ and the numbering $Enu(t) = (Enu_E(v))(Enu_E(w))$. Thus, the enumeration function is
$$Enu(t) = \begin{cases} Enu_E(t), & t \in T_E; \\ (Enu_E(v))(Enu_E(w)), & t \text{ results from synchronization} \\ & \text{of } v \text{ and } w. \end{cases}$$
When we synchronize the same set of transitions in different orders, we obtain several resulting transitions with the same label and probability, but with different numberings having the same content. In this case, we shall consider only a single transition from the resulting ones in the plain dts-box to avoid introducing redundant transitions. For example, if the transitions t and u are generated by synchronizing v and w in different orders, we have $\Lambda(t) = (\alpha \oplus_a \beta, \rho \cdot \chi) = \Lambda(u)$, but $Enu(t) = (Enu_E(v))(Enu_E(w)) \neq (Enu_E(w))(Enu_E(v)) = Enu(u)$, whereas $Cont(Enu(t)) = Cont(Enu(v)) \cup Cont(Enu(w)) = Cont(Enu(u))$. Then only one transition t (or, symmetrically, u) will appear in $Box_{dts}(E\ \mathsf{sy}\ a)$.
- $Box_{dts}([E * F * K]) = \Theta_{[**]}(Box_{dts}(E), Box_{dts}(F), Box_{dts}(K))$. Since we do not introduce any new transitions, we preserve the initial numbering: $Enu(t) = \begin{cases} Enu_E(t), & t \in T_E; \\ Enu_F(t), & t \in T_F; \\ Enu_K(t), & t \in T_K. \end{cases}$

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

Definition 10. 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. $Box_{dts}((\alpha, \rho)_\iota) = N_{(\alpha,\rho)_\iota}$;
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 static 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 this definition is compositional in the sense that, for any arbitrary dynamic expression, we may decompose it in some inner dynamic and static expressions, for which we may apply the definition, thus obtaining the corresponding plain dts-boxes, which can be joined according to the term structure (by 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 transitions of the dts-box corresponding to an 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 1. *For any static expression* E, $TS(\overline{E}) \simeq RG(Box_{dts}(\overline{E}))$.

Proof. For 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 via the same probability functions. □

Proposition 1. *For any static expression* E,

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

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

5. Step stochastic bisimulation equivalence

Bisimulation equivalences respect the particular points of choice in the behavior 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 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 $[G']_{\approx}$, are to 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 the definition below, we consider $\mathcal{L}(\Gamma) \in I\!N_f^{\mathcal{L}}$ for $\Gamma \in I\!N_f^{\mathcal{SL}}$, i.e., the (possibly empty) multisets of multiactions. The multiactions can be empty, then $\mathcal{L}(\Gamma)$ contains the elements $\emptyset$, and it is not empty itself.

Let G be a dynamic expression and $\mathcal{H} \subseteq DR(G)$. Then, for any $s \in DR(G)$ and $A \in I\!N_f^{\mathcal{L}}$, we write $s \stackrel{A}{\rightarrow}_{\mathcal{P}} \mathcal{H}$, where $\mathcal{P} = PM_A(s, \mathcal{H})$ is the *overall probability to move from s into the set of states $\mathcal{H}$ via steps with the multiaction part A* defined as

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

We write $s \stackrel{A}{\rightarrow} \mathcal{H}$ if $\exists \mathcal{P}\ s \stackrel{A}{\rightarrow}_{\mathcal{P}} \mathcal{H}$. Further, we write $s \rightarrow_{\mathcal{P}} \mathcal{H}$ if $\exists A\ s \stackrel{A}{\rightarrow} \mathcal{H}$, where $\mathcal{P} = PM(s, \mathcal{H})$ is the *overall probability to move from s into the set of states $\mathcal{H}$ via any steps* defined as

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

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 to come from any two equivalent states into the same "composite" equivalence class (w.r.t. 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 11. 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. $(s_1, s_2) \in \mathcal{R} \ \Rightarrow\ \forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}\ \forall A \in I\!N_f^{\mathcal{L}}$
   $s_1 \stackrel{A}{\rightarrow}_{\mathcal{P}} \mathcal{H} \ \Leftrightarrow\ s_2 \stackrel{A}{\rightarrow}_{\mathcal{P}} \mathcal{H}$.

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

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 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. □

The equivalences which we proposed can be used to reduce transition systems and DTMCs of expressions (reachability graphs and DTMCs of dts-boxes). Reductions of graph-based models, like transition systems, reachability graphs and DTMCs, result in those with less states (the graph nodes). The goal of the reduction is to decrease the number of states in the semantic representation of the modeled system while preserving its important qualitative and quantitative properties. Thus, the reduction allows one to simplify the behaviour and performance analysis of systems.

An *autobisimulation* is a bisimulation between an expression and itself. For a dynamic expression G and a step stochastic autobisimulation on it $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G$, let $\mathcal{K} \in DR(G)/_{\mathcal{R}}$ and $s_1, s_2 \in \mathcal{K}$. We have $\forall \widetilde{\mathcal{K}} \in DR(G)/_{\mathcal{R}}$ $\forall A \in \mathbb{N}_f^{\mathcal{L}}$ $s_1 \stackrel{A}{\rightarrow}_{\mathcal{P}} \widetilde{\mathcal{K}} \Leftrightarrow s_2 \stackrel{A}{\rightarrow}_{\mathcal{P}} \widetilde{\mathcal{K}}$. The previous statement is valid for all $s_1, s_2 \in \mathcal{K}$, hence, 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 $\exists \mathcal{P}$ $\mathcal{K} \stackrel{A}{\rightarrow}_{\mathcal{P}} \widetilde{\mathcal{K}}$ and $\mathcal{K} \rightarrow \widetilde{\mathcal{K}}$ if $\exists A$ $\mathcal{K} \stackrel{A}{\rightarrow} \widetilde{\mathcal{K}}$. The 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 (w.r.t. $\mathcal{R}$) 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 (w.r.t. $\mathcal{R}$) of states* of G, denoted by $SJ_{\mathcal{R}}$, has the elements $SJ_{\mathcal{R}}(\mathcal{K})$, $\mathcal{K} \in DR(G)/_{\mathcal{R}}$. The *sojourn time variance in the equivalence class (w.r.t. $\mathcal{R}$) of states $\mathcal{K}$* is $VAR_{\mathcal{R}}(\mathcal{K}) = \frac{1}{(1-PM(\mathcal{K},\mathcal{K}))^2}$. The *sojourn time variance vector for the equivalence classes (w.r.t. $\mathcal{R}$) of states* of G, denoted by $VAR_{\mathcal{R}}$, has the elements $VAR_{\mathcal{R}}(\mathcal{K})$, $\mathcal{K} \in DR(G)/_{\mathcal{R}}$.

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 2, $\mathcal{R}_{ss}(G)$ is the largest step stochastic autobisimulation on G. Based on the equivalence classes w.r.t. $\mathcal{R}_{ss}(G)$, the quotient (by $\underline{\leftrightarrow}_{ss}$) transition systems and the quotient (by $\underline{\leftrightarrow}_{ss}$) underlying DTMCs of expressions can be defined. The mentioned equivalence classes 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 one.

Definition 12. Let G be a dynamic expression. The *quotient (by $\underline{\leftrightarrow}_{ss}$) (labeled probabilistic) transition system* of G is a quadruple $TS_{\underline{\leftrightarrow}_{ss}}(G) = (S_{\underline{\leftrightarrow}_{ss}}, L_{\underline{\leftrightarrow}_{ss}}, \mathcal{T}_{\underline{\leftrightarrow}_{ss}}, s_{\underline{\leftrightarrow}_{ss}})$, where

- $S_{\underline{\leftrightarrow}_{ss}} = DR(G)/_{\mathcal{R}_{ss}(G)}$;

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

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

Definition 13. Let G be a dynamic expression. The *quotient (by $\underline{\leftrightarrow}_{ss}$) underlying DTMC* of G, denoted by $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$, has the state space $DR(G)/_{\mathcal{R}_{ss}(G)}$ and the transitions $\mathcal{K} \to_{\mathcal{P}} \widetilde{\mathcal{K}}$, 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 transition systems and underlying DTMCs are the minimal reductions of the mentioned objects modulo $\underline{\leftrightarrow}_{ss}$. The quotients can be used to simplify analysis of system properties which are preserved by $\underline{\leftrightarrow}_{ss}$, since less states should be examined for it. The comprehensive reduction example will be presented in Section 7.

6. Performance evaluation

Stationary distribution is used for performance evaluation. Performance indices are calculated based on the steady-state probabilities. Let us describe the stationary behaviour of infinite stochastic processes specified by dynamic expressions whose underlined DTMCs contain one ergodic subset of states.

Let G be a dynamic expression. The elements $\mathcal{P}_{ij}$ $(1 \le i, j \le n = |DR(G)|)$ of the (one-step) transition probability matrix (TPM) $\mathbf{P}$ for $DTMC(G)$ are defined as $\mathcal{P}_{ij} = \begin{cases} PM(s_i, s_j), & s_i \to s_j; \\ 0, & \text{otherwise.} \end{cases}$

The transient (k-step, $k \in I\!N$) probability mass function (PMF) $\psi[k] = (\psi_1[k], \ldots, \psi_n[k])$ for $DTMC(G)$ is a solution of the equation system $\psi[k] = \psi[0]\mathbf{P}^k$, s.t. $\psi[0] = (\psi_1[0], \ldots, \psi_n[0])$ is the initial PMF $\psi_i[0] = \begin{cases} 1, & s_i = [G]_{\approx}; \\ 0, & \text{otherwise.} \end{cases}$ Note also that $\psi[k+1] = \psi[k]\mathbf{P}$ $(k \in I\!N)$.

The steady-state PMF $\psi = (\psi_1, \ldots, \psi_n)$ for $DTMC(G)$ is a solution of the equation system $\begin{cases} \psi(\mathbf{P} - \mathbf{E}) = \mathbf{0} \\ \psi \mathbf{1}^T = 1 \end{cases}$, where $\mathbf{E}$ is a unitary matrix of dimension n and $\mathbf{0}$ is a vector with n values 0, $\mathbf{1}$ is that with n values 1.

When $DTMC(G)$ has a single steady state, we have $\psi = \lim_{k\to\infty} \psi[k]$. Let $s, \tilde{s} \in DR(G)$, $S, \widetilde{S} \subseteq DR(G)$ for a dynamic expression G. The following *performance indices* are based on the steady-state PMF for $DTMC(G)$.

- The *average recurrence (return) time in the state s* 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* S *w.r.t. 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 to perform a step with an activity* (α,ρ) 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 *on the states* is $\sum_{s\in DR(G)}\psi(s)r(s)$.

The following proposition demonstrates that, for two dynamic expressions related by $\underline{\leftrightarrow}_{ss}$, the steady-state probabilities to come in an equivalence class coincide. One can also interpret the result stating that the mean recurrence time for an equivalence class is the same for both expressions.

Proposition 3. *Let* G, G' *be dynamic expressions with* $\mathcal{R} : G\underline{\leftrightarrow}_{ss}G'$. *Then* $\forall\mathcal{H}\in(DR(G)\cup DR(G'))/_{\mathcal{R}}$

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

Proof. Analogous to the proof of Proposition 3 from [15], but with the use of the probability functions respecting empty loops. □

By Proposition 3, $\underline{\leftrightarrow}_{ss}$ preserves the quantitative properties of the stationary behaviour. Now we shall demonstrate that the qualitative properties of the stationary behaviour based on the multiaction labels are preserved as well.

Definition 14. A *derived step trace* of a dynamic expression G is $\Sigma = A_1\cdots A_n\in(I\!N_f^{\mathcal{L}})^*$ s.t. $\exists s\in DR(G)\ s\xrightarrow{\Gamma_1}s_1\xrightarrow{\Gamma_2}\cdots\xrightarrow{\Gamma_n}s_n,\ \mathcal{L}(\Gamma_i)=A_i$ $(1\le i\le n)$. The *probability to execute the derived step trace* Σ *in* s is

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

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

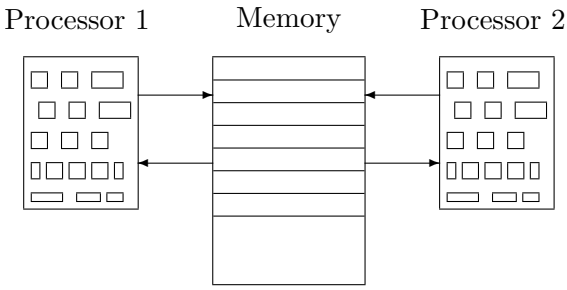

Figure 2. The diagram of the shared memory system

Theorem 2. *Let G, G' be dynamic expressions with $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$ and Σ be a derived step trace of G and G'. Then $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$*

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

Proof. Analogous to the proof of Theorem 4 from [15], but with the use of the probability functions respecting empty loops. □

7. The generalized shared memory system

Consider a model of two processors accessing a common shared memory described in [1] in the continuous time setting on GSPNs. We shall analyze this shared memory system in the discrete time stochastic setting of dtsPBC, where concurrent execution of activities is possible. The model performs as follows. After activation of the system (turning the computer on), two processors are active, and the common memory is available. Each processor can request an access to the memory. When a processor starts acquisition of the memory, another processor should wait until the former one ends its memory operations, and the system returns to the state with both active processors and the available common memory. The diagram of the system is depicted in Figure 2.

7.1. The concrete system

Let us explain the meaning of actions from the syntax of dtsPBC expressions which will specify the system modules. The action a corresponds to the system activation. The actions r_i $(1 \leq i \leq 2)$ represent the common memory request of processor i. The actions b_i and e_i correspond to the beginning and the end, respectively, of the common memory access of processor i. The other actions are used for communication purposes only via synchronization, and we abstract from them later using restriction.

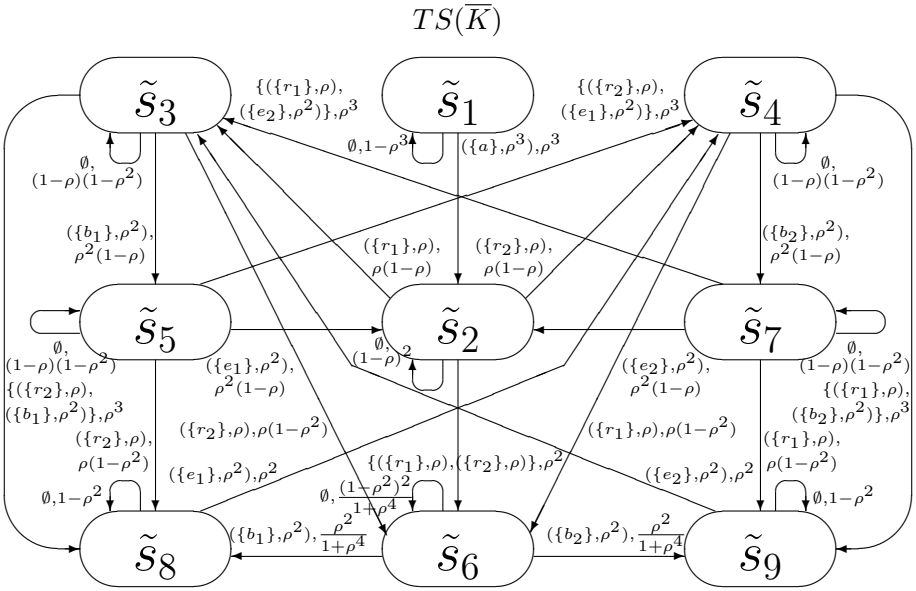

Figure 3. The transition system of the generalized shared memory system

Let us determine which is the influence of the multiaction probabilities from specification of the shared memory system on its performance. Suppose that all the multiactions have the same generalized probability ρ. The resulting specification K is defined as follows.

The static expression of the first processor is

$K_1 = [(\{x_1\},\rho) * ((\{r_1\},\rho);(\{b_1,y_1\},\rho);(\{e_1,z_1\},\rho)) * \mathsf{Stop}]$.

The static expression of the second processor is

$K_2 = [(\{x_2\},\rho) * ((\{r_2\},\rho);(\{b_2,y_2\},\rho);(\{e_2,z_2\},\rho)) * \mathsf{Stop}]$.

The static expression of the shared memory is

$K_3 = [(\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{\widehat{y_1}\},\rho);(\{\widehat{z_1}\},\rho))[]((\{\widehat{y_2}\},\rho);(\{\widehat{z_2}\},\rho))) * \mathsf{Stop}]$.

The static expression of the generalized shared memory system with two processors is $K = (K_1 \| K_2 \| K_3)$ sy x_1 sy x_2 sy y_1 sy y_2 sy z_1 sy z_2 rs x_1 rs x_2 rs y_1 rs y_2 rs z_1 rs z_2.

$DR(\overline{K})$ consists of 9 equivalence classes: $\tilde{s}_1$ is the initial state, $\tilde{s}_2$: the system is activated and the memory is not requested, $\tilde{s}_3$: the memory is requested by the first processor, $\tilde{s}_4$: the memory is requested by the second processor, $\tilde{s}_5$: the memory is allocated to the first processor, $\tilde{s}_6$: the memory is requested by two processors, $\tilde{s}_7$: the memory is allocated to the second processor, $\tilde{s}_8$: the memory is allocated to the first processor and the memory is requested by the second processor, $\tilde{s}_9$: the memory is allocated to the second processor and the memory is requested by the first processor.

Figure 3 presents the transition system $TS(\overline{K})$.

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

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

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

$$\widetilde{VAR} = \left(\tfrac{1}{\rho^6}, \tfrac{1}{\rho^2(2-\rho)^2}, \tfrac{1}{\rho 2(1+\rho-\rho^2)^2}, \tfrac{1}{\rho^2(1+\rho-\rho^2)^2}, \tfrac{1}{\rho^2(1+\rho-\rho^2)^2}, \tfrac{(1+\rho^4)^2}{2\rho^4}, \tfrac{1}{\rho^2(1+\rho-\rho^2)^2}, \tfrac{1}{\rho^4}, \tfrac{1}{\rho^4}\right).$$

Let us denote $\chi = 1-\rho$ and $\theta = 1-\rho^2$. The TPM for $DTMC(\overline{K})$ is

$$\widetilde{\mathbf{P}} = \begin{bmatrix} 1-\rho^3 & \rho^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \chi^2 & \rho\chi & \rho\chi & 0 & \rho^2 & 0 & 0 & 0 \\ 0 & 0 & \chi\theta & 0 & \rho^2\chi & \rho\theta & 0 & \rho^3 & 0 \\ 0 & 0 & 0 & \chi\theta & 0 & \rho^2\chi & \rho\theta & 0 & \rho^3 \\ 0 & \rho^2\chi & 0 & \rho^3 & \chi\theta & 0 & 0 & \rho\theta & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{\theta^2}{1+\rho^4} & 0 & \frac{\rho^2}{1+\rho^4} & \frac{\rho^2}{1+\rho^4} \\ 0 & \rho^2\chi & \rho^3 & 0 & 0 & 0 & \chi\theta & 0 & \rho\theta \\ 0 & 0 & 0 & \rho^2 & 0 & 0 & 0 & \theta & 0 \\ 0 & 0 & \rho^2 & 0 & 0 & 0 & 0 & 0 & \theta \end{bmatrix}.$$

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

$$\tilde{\psi} = \tfrac{1}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(0, 2\rho^3(1-\rho)^2, \rho(2-\rho)(1+\rho-\rho^2), \rho(2-\rho)(1+\rho-\rho^2), \rho^2(2-3\rho+\rho^2), (1+\rho^4)(2+\rho-5\rho^2+\rho^3+\rho^4), \rho^2(2-3\rho+\rho^2), 2+3\rho-6\rho^2+\rho^3+\rho^4, 2+3\rho-6\rho^2+\rho^3+\rho^4).$$

We can now calculate the main performance indices.

- The average recurrence time in the state $\tilde{s}_2$, where no processor requests the memory, called the *average system run-through*, is $\frac{1}{\tilde{\psi}_2} = \frac{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}{2\rho^3(1-\rho)^2}$.
- The common memory is available only in the states $\tilde{s}_2, \tilde{s}_3, \tilde{s}_4, \tilde{s}_6$. The steady-state probability that the memory is available is $\tilde{\psi}_2 + \tilde{\psi}_3 + \tilde{\psi}_4 + \tilde{\psi}_6 = \frac{2\rho^3(1-\rho)^2}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8} + \frac{\rho(2-\rho)(1+\rho-\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8} + \frac{\rho(2-\rho)(1+\rho-\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8} + \frac{(1+\rho^4)(2+\rho-5\rho^2+\rho^3+\rho^4)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8} = \frac{2+5\rho-3\rho^2-3\rho^3+\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$. Then the steady-state probability that the memory is used (i.e., not available), called the *shared memory utilization*, is $1 - \frac{2+5\rho-3\rho^2-3\rho^3+\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8} = \frac{4+6\rho-8\rho^2-4\rho^3+4\rho^4}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$.

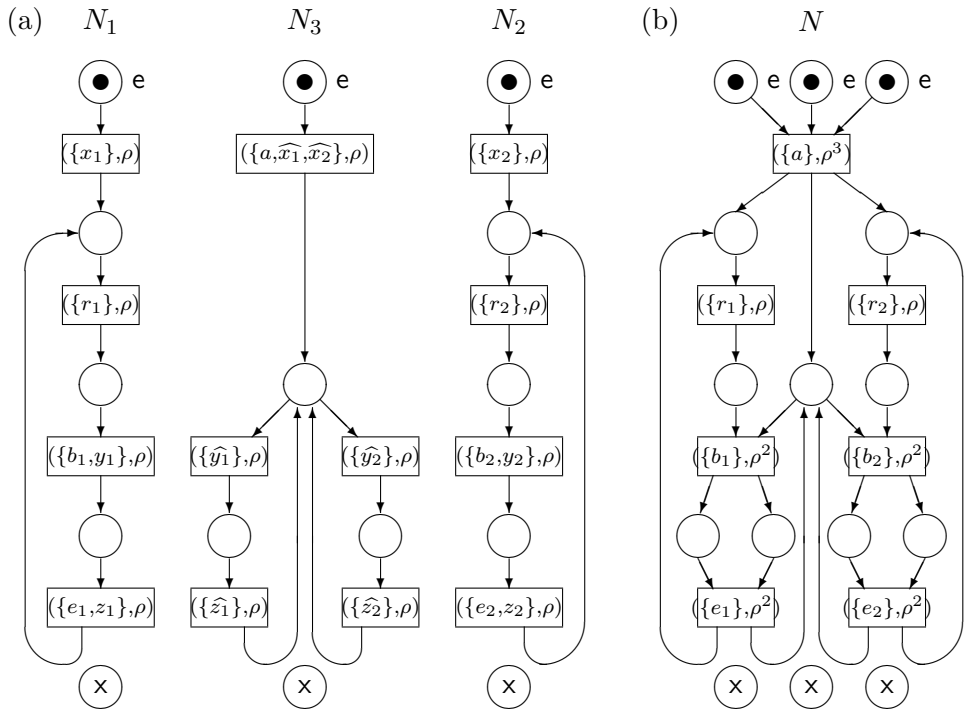

Figure 4. The marked dts-boxes of two processors, shared memory and the shared memory system

- The common memory request of the first processor $(\{r_1\},\rho)$ is only possible from the states $\tilde{s}_2, \tilde{s}_4, \tilde{s}_7$. In each of the states, the request probability is the sum of the execution probabilities for all multisets of activities containing $(\{r_1\},\rho)$. Thus, the *steady-state probability of the shared memory request from the first processor* is
$\tilde{\psi}_2 \sum_{\{\Gamma|(\{r_1\},\rho)\in\Gamma\}} PT(\Gamma,\tilde{s}_2) + \tilde{\psi}_4 \sum_{\{\Gamma|(\{r_1\},\rho)\in\Gamma\}} PT(\Gamma,\tilde{s}_4) +$
$\tilde{\psi}_7 \sum_{\{\Gamma|(\{r_1\},\rho)\in\Gamma\}} PT(\Gamma,\tilde{s}_7) =$
$\frac{2\rho^3(1-\rho)^2}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(\rho(1-\rho)+\rho^2) +$
$\frac{\rho(2-\rho)(1+\rho-\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(\rho(1-\rho^2)+\rho^3) +$
$\frac{\rho^2(2-3\rho+\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(\rho(1-\rho^2)+\rho^3) =$
$\frac{\rho^2(2+3\rho-4\rho^2-2\rho^3+2\rho^4)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}.$

Figure 4(a) presents the marked dts-boxes corresponding to the dynamic expressions of two processors and shared memory, i.e., $N_i = Box_{dts}(\overline{K_i})$ $(1 \leq i \leq 3)$. Figure 4(b) depicts the marked dts-box of the dynamic expression of the shared memory system, i.e., $N = Box_{dts}(\overline{K})$.

7.2. The abstract system and its reduction

Let us consider a modification of the generalized shared memory system with abstraction from identifiers of the processors. We call this system the abstract generalized shared memory one.

The static expression of the first processor is
$L_1 = [(\{x_1\},\rho) * ((\{r\},\rho);(\{b,y_1\},\rho);(\{e,z_1\},\rho)) * \mathsf{Stop}]$.

The static expression of the second processor is
$L_2 = [(\{x_2\},\rho) * ((\{r\},\rho);(\{b,y_2\},\rho);(\{e,z_2\},\rho)) * \mathsf{Stop}]$.

The static expression of the shared memory is
$L_3 = [(\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{\widehat{y_1}\},\rho);(\{\widehat{z_1}\},\rho))[](((\{\widehat{y_2}\},\rho);(\{\widehat{z_2}\},\rho))) * \mathsf{Stop}]$.

The static expression of the abstract shared memory generalized system with two processors is $L = (L_1 \| L_2 \| L_3)$ sy x_1 sy x_2 sy y_1 sy y_2 sy z_1 sy z_2 rs x_1 rs x_2 rs y_1 rs y_2 rs z_1 rs z_2.

$DR(\overline{L})$ resembles $DR(\overline{K})$, and $TS(\overline{L})$ is similar to $TS(\overline{K})$. We have $DTMC(\overline{L}) = DTMC(\overline{K})$. Thus, the TPMs and the steady-state PMFs for $DTMC(\overline{L})$ and $DTMC(\overline{K})$ coincide.

The first and second performance indices are the same for the concrete generalized system and its abstract modification. Let us consider the following performance index based on non-identified viewpoint to the processors.

- The common memory request of a processor $(\{r\},\rho)$ is only possible from the states $\tilde{s}_2, \tilde{s}_3, \tilde{s}_4, \tilde{s}_5, \tilde{s}_7$. In each of the states, the request probability is the sum of the execution probabilities for all multisets of activities containing $(\{r\},\rho)$. Thus, the *steady-state probability of the shared memory request from a processor* is $\tilde{\psi}_2 \sum_{\{\Gamma|(\{r\},\rho)\in\Gamma\}} PT(\Gamma,\tilde{s}_2) +$ $\tilde{\psi}_3 \sum_{\{\Gamma|(\{r\},\rho)\in\Gamma\}} PT(\Gamma,\tilde{s}_3) + \tilde{\psi}_4 \sum_{\{\Gamma|(\{r\},\rho)\in\Gamma\}} PT(\Gamma,\tilde{s}_4) +$ $\tilde{\psi}_5 \sum_{\{\Gamma|(\{r\},\rho)\in\Gamma\}} PT(\Gamma,\tilde{s}_5) + \tilde{\psi}_7 \sum_{\{\Gamma|(\{r\},\rho)\in\Gamma\}} PT(\Gamma,\tilde{s}_7) =$ $\frac{2\rho^3(1-\rho)^2}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(\rho(1-\rho)+\rho(1-\rho)+\rho^2) +$ $\frac{\rho(2-\rho)(1+\rho-\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(\rho(1-\rho^2)+\rho^3) +$ $\frac{\rho(2-\rho)(1+\rho-\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(\rho(1-\rho^2)+\rho^3) +$ $\frac{\rho^2(2-3\rho+\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(\rho(1-\rho^2)+\rho^3) +$ $\frac{\rho^2(2-3\rho+\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(\rho(1-\rho^2)+\rho^3) =$ $\frac{2\rho^2(2-\rho)(1+\rho-\rho^2)^2}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$.

We have $DR(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_1,\widetilde{\mathcal{K}}_2,\widetilde{\mathcal{K}}_3,\widetilde{\mathcal{K}}_4,\widetilde{\mathcal{K}}_5,\widetilde{\mathcal{K}}_6\}$, where $\widetilde{\mathcal{K}}_1 = \{\tilde{s}_1\}$ (the initial state), $\widetilde{\mathcal{K}}_2 = \{\tilde{s}_2\}$ (the system is activated and the memory is not requested), $\widetilde{\mathcal{K}}_3 = \{\tilde{s}_3,\tilde{s}_4\}$ (the memory is requested by one processor), $\widetilde{\mathcal{K}}_4 = \{\tilde{s}_5,\tilde{s}_7\}$ (the memory is allocated to a processor), $\widetilde{\mathcal{K}}_5 = \{\tilde{s}_6\}$ (the memory is requested by two processors), $\widetilde{\mathcal{K}}_6 = \{\tilde{s}_8,\tilde{s}_9\}$ (the memory is allocated to a processor and the memory is requested by another processor).

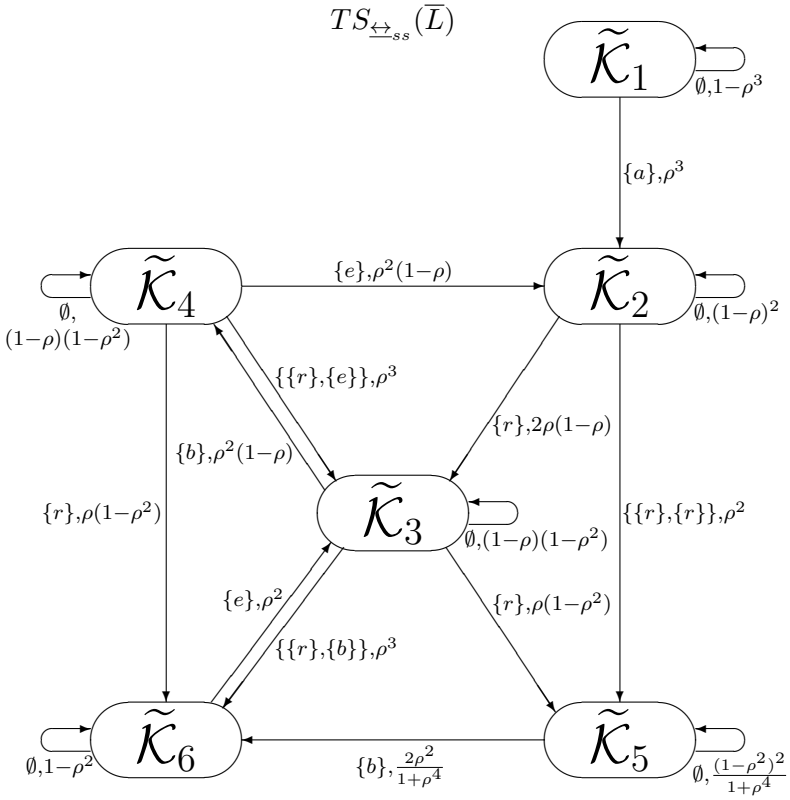

Figure 5. The quotient transition system of the abstract generalized shared memory system

Figure 5 presents the quotient transition system $TS_{\underline{\leftrightarrow}_{ss}}(\overline{L})$.
The quotient average sojourn time vector of $\overline{L}$ is

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

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

$$\widetilde{VAR}' = \left(\frac{1}{\rho^6}, \frac{1}{\rho^2(2-\rho)^2}, \frac{1}{\rho^2(1+\rho-\rho^2)^2}, \frac{1}{\rho^2(1+\rho-\rho^2)^2}, \frac{(1+\rho^4)^2}{4\rho^4}, \frac{1}{\rho^4}\right).$$

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

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

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

$$\tilde{\psi}' = \tfrac{1}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(0, 2\rho^3(1-\rho)^2, 2\rho(2-\rho)(1+\rho-\rho^2), 2\rho^2(2-3\rho+\rho^2), (1+\rho^4)(2+\rho-5\rho^2+\rho^3+\rho^4), 2(2+3\rho-6\rho^2+\rho^3+\rho^4)).$$

We can now calculate the main performance indices.

- The average recurrence time in the state $\widetilde{\mathcal{K}}_2$, where no processor requests the memory, called the *average system run-through*, is $\frac{1}{\tilde{\psi}'_2} = \frac{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}{2\rho^3(1-\rho)^2}$.
- The common memory is available only in the states $\widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_5$. The steady-state probability that the memory is available is $\tilde{\psi}'_2+\tilde{\psi}'_3+\tilde{\psi}'_5 = \frac{2\rho^3(1-\rho)^2}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8} + \frac{2\rho(2-\rho)(1+\rho-\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8} + \frac{(1+\rho^4)(2+\rho-5\rho^2+\rho^3+\rho^4)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8} = \frac{2+5\rho-3\rho^2-3\rho^3+\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$. Then the steady-state probability that the memory is used (i.e., not available), called the *shared memory utilization*, is $1 - \frac{2+5\rho-3\rho^2-3\rho^3+\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8} = \frac{4+6\rho-8\rho^2-4\rho^3+4\rho^4}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$.
- The common memory request of a processor $\{r\}$ is only possible from the states $\widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_4$. In each of the states, the request probability is the sum of the execution probabilities for all multisets of multiactions containing $\{r\}$. Thus, the *steady-state probability of the shared memory request from a processor* is $\tilde{\psi}'_2 \sum_{\{A,\widetilde{\mathcal{K}}|\{r\}\in A,\ \widetilde{\mathcal{K}}_2 \stackrel{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}) + \tilde{\psi}'_3 \sum_{\{A,\widetilde{\mathcal{K}}|\{r\}\in A,\ \widetilde{\mathcal{K}}_3 \stackrel{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}) + \tilde{\psi}'_4 \sum_{\{A,\widetilde{\mathcal{K}}|\{r\}\in A,\ \widetilde{\mathcal{K}}_4 \stackrel{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_4, \widetilde{\mathcal{K}}) = \frac{2\rho^3(1-\rho)^2}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(2\rho(1-\rho)+\rho^2) + \frac{2\rho(2-\rho)(1+\rho-\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(\rho(1-\rho^2)+\rho^3) + \frac{2\rho^2(2-3\rho+\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}(\rho(1-\rho^2)+\rho^3) = \frac{2\rho^2(2-\rho)(1+\rho-\rho^2)^2}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$.

One can see that the performance indices are the same for the complete and the quotient abstract generalized shared memory systems. The coincidence of the first and second performance indices obviously illustrates the result of Proposition 3. The coincidence of the third performance index is due to Theorem 2: one should just apply its result in the step traces $\{\{r\}\}$, $\{\{r\},\{r\}\}$, $\{\{r\},\{b\}\}$, $\{\{r\},\{e\}\}$ of the expression $\overline{L}$ and itself, and then sum the left and right parts of the three resulting equalities.

Let us consider what is the effect of quantitative changes of the parameter ρ upon performance of the quotient abstract generalized shared memory system in its steady state. Remember that $\rho \in (0;1)$ is the probability of every multiaction of the system. The closer is ρ to 0, the less is the probability to execute some activities at every discrete time step, hence, the system will most probably *stand idle*. The closer is ρ to 1, the greater is the probability to execute some activities at every discrete time step, hence, the system will most probably *operate*.

Since $\tilde{\psi}'_1 = 0$, only $\tilde{\psi}'_2 = \frac{2\rho^3(1-\rho)^2}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$, $\tilde{\psi}'_3 = \frac{2\rho(2-\rho)(1+\rho-\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$, $\tilde{\psi}'_4 = \frac{2\rho^2(2-3\rho+\rho^2)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$, $\tilde{\psi}'_5 = \frac{(1+\rho^4)(2+\rho-5\rho^2+\rho^3+\rho^4)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$, $\tilde{\psi}'_6 = \frac{2(2+3\rho-6\rho^2+\rho^3+\rho^4)}{6+11\rho-11\rho^2-7\rho^3+5\rho^4+3\rho^5-5\rho^6+\rho^7+\rho^8}$ depend on ρ. Figure 6 depicts the graphs of the steady-state probabilities $\tilde{\psi}'_2,\ \tilde{\psi}'_3,\ \tilde{\psi}'_4,\ \tilde{\psi}'_5,\ \tilde{\psi}'_6$ as functions of ρ. Remember that we do not allow $\rho = 0$ or $\rho = 1$.

It is easy to see that $\tilde{\psi}'_2,\ \tilde{\psi}'_3,\ \tilde{\psi}'_4$ tend to 0, and $\tilde{\psi}'_5$, increasing, tends to $\frac{1}{3}$, and $\tilde{\psi}'_6$, increasing, tends to $\frac{2}{3}$, when ρ approaches 0. Thus, the closer is ρ to 0, the greater is the probability that two processors require the memory or the memory is allocated to a processor and required by the other one, hence, we get *more unsatisfied memory requests*.

Further, $\tilde{\psi}'_2,\ \tilde{\psi}'_4,\ \tilde{\psi}'_5$ tend to 0, and $\tilde{\psi}'_3$, growing, tends to $\frac{1}{2}$, and $\tilde{\psi}'_6$, decreasing and slightly increasing, tends to $\frac{1}{2}$, when ρ approaches 1. Thus, the closer is ρ to 1, the greater is the probability that the memory is allocated to a processor (and not required by the other one), moreover, in general, the less is the probability that the memory is allocated to a processor and required by the other one, hence, we get *less unsatisfied memory requests*.

The maximal value of $\tilde{\psi}'_2$ is 0.0090 when $\rho = 0.6380$. In this case the probability that the system is activated and the memory is not required is maximal, i.e., the *maximal shared memory availability* is about 1%.

The maximal value of $\tilde{\psi}'_4$ is 0.0534 when $\rho = 0.6855$. In this case the probability that the memory is allocated to a processor (and not required by the other one) is maximal, i.e., the *maximal probability that, during interaction of a processor with the memory, there are no memory requests from the other processor (the maximal probability to have no unsatisfied memory requests during interaction of a processor with the memory)* is about 5%.

The minimal value of $\tilde{\psi}'_6$ is 0.4698 when $\rho = 0.8543$. In this case the probability that the memory is allocated to a processor and required by the other one is minimal, the *minimal probability that, during interaction of a processor with the memory, there are memory requests from the other processor (the minimal probability to have unsatisfied memory requests during interaction of a processor with the memory)* is about 47%.

Figure 7 depicts the graph of the average system run-through $\frac{1}{\tilde{\psi}'_2}$ as a function of ρ. One can see that the run-through tends to ∞ when ρ

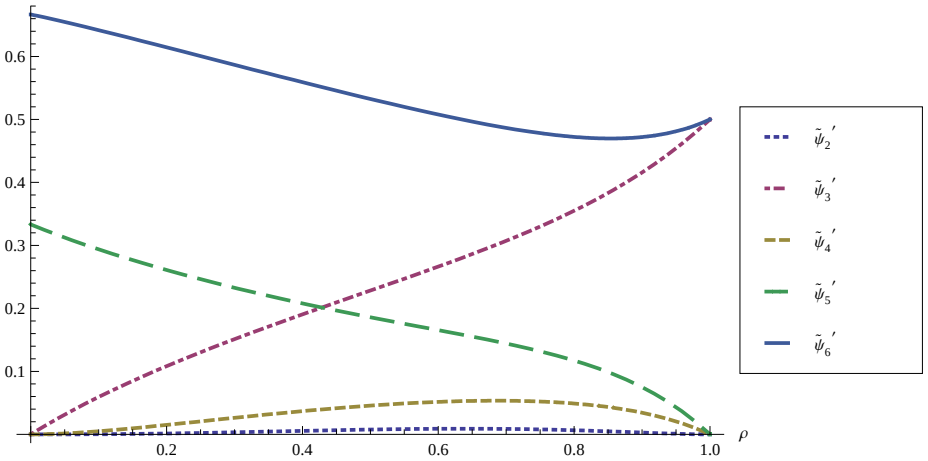

Figure 6. Steady-state probabilities $\tilde{\psi}_2'$, $\tilde{\psi}_3'$, $\tilde{\psi}_4'$, $\tilde{\psi}_5'$, $\tilde{\psi}_6'$ as functions of ρ

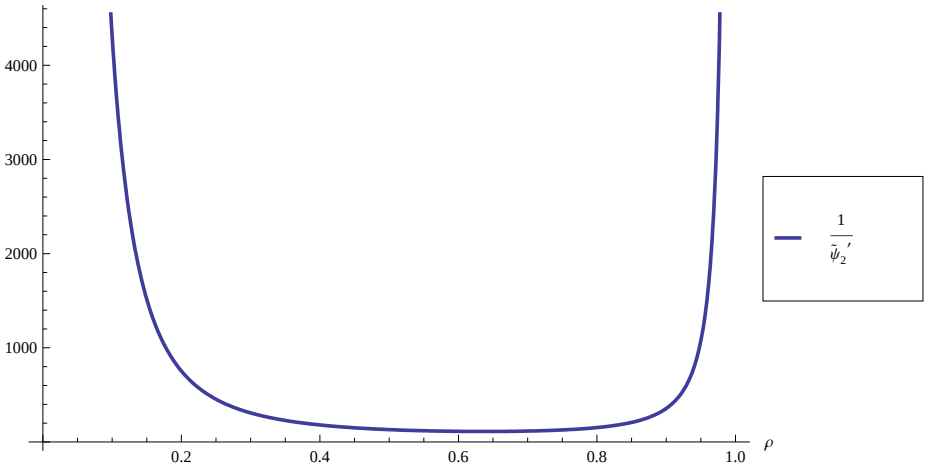

Figure 7. Average system run-through $\frac{1}{\tilde{\psi}_2'}$ as a function of ρ

approaches 0 or 1. Its minimal value 111.6834 is reached when $\rho = 0.6380$. To speed up operation of the system, one should take ρ closer to 0.6380.

The first graph in Figure 8 represents the shared memory utilization $1 - \tilde{\psi}_2' - \tilde{\psi}_3' - \tilde{\psi}_5'$ as a function of ρ. It is clear that the utilization tends to $\frac{1}{3}$ when ρ approaches 0, and it tends to $\frac{1}{2}$ when ρ approaches 1. Thus, the *minimal shared memory utilization* is about 33%. To increase the utilization, one should take ρ closer to 1.

The second graph in Figure 8 represents the steady-state probability of the shared memory request from a processor $\tilde{\psi}_2'\widetilde{\Sigma}_2' + \tilde{\psi}_3'\widetilde{\Sigma}_3' + \tilde{\psi}_4'\widetilde{\Sigma}_4'$, where $\widetilde{\Sigma}_i' = \sum_{\{A,\widetilde{\mathcal{K}} | \{r\} \in A,\ \widetilde{\mathcal{K}}_i \overset{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_i, \widetilde{\mathcal{K}}),\ i \in \{2,3,4\}$, as a function of ρ. One can see that the probability tends to 0 when ρ approaches 0 and it tends to

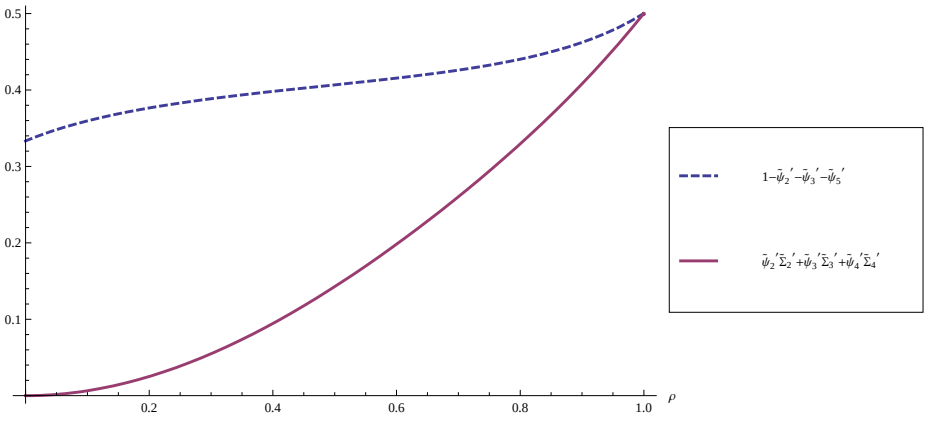

Figure 8. Some performance indices as functions of ρ

$\frac{1}{2}$ when ρ approaches 1. To increase the mentioned probability, one should take ρ closer to 1.

8. Conclusion

In this paper, within dtsPBC with iteration, a method of modeling, performance evaluation and performance preserving reduction of concurrent stochastic systems was proposed based on steady-state probabilities analysis. The transition systems and underlying DTMCs of expressions were reduced w.r.t. step stochastic bisimulation equivalence that guarantees identity of the stationary behaviour and thus preserves performance measures. The method was applied to the generalized shared memory system with a variable probability of activities. This probability was interpreted as a parameter of the performance index functions. The influence of the parameter value to the system's performance was analyzed with a goal of optimization.

We plan to investigate stochastic equivalences of dtsPBC which allow one to identify stochastic processes with similar behaviour that are differentiated by too strict notion of the semantic equivalence. Moreover, we would like to extend dtsPBC with recursion to enhance specification power of the calculus.

References

[1] Balbo G. Introduction to stochastic Petri nets // Lect. Notes Comp. Sci. – 2001. – Vol. 2090. – P. 84–155.

[2] 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.

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

[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] Hillston J. A Compositional Approach to Performance Modelling. – Cambridge University Press, Great Britain, 1996. – 158 p. – http://www.dcs.ed.ac.uk/pepa/book.pdf

[6] Hermanns H., Rettelbach M. Syntax, semantics, equivalences and axioms for MTIPP // Proc. of 2^{nd} Workshop on Process Algebras and Performance Modelling. – University of Erlangen, Germany, 1994. – P. 71–88 – (Arbeitsberichte des IMMD; Vol. 27).

[7] Macià H. sPBC: Una extensión Markoviana del Petri box calculus. – Ph.D. thesis, Departamento de Informática, Universidad de Castilla-La Mancha, Albacete, Spain, 2003. – 249 p. (In Spanish) – http://www.info-ab.uclm.es/retics/publications/2003/sPBCthesis03.pdf.

[8] Macià H.S., Valero V.R., Cazorla D.L., Cuartero F.G. Introducing the iteration in sPBC. – Department of Computer Science, University of Castilla-La Mancha, Albacete, Spain, September 2003. – 20 p. – (Technical Report; Vol. DIAB-03-01-37). – http://www.info-ab.uclm.es/descargas/tecnicalreports/DIAB-03-01-37/diab030137.zip

[9] Macià H., Valero V., Cazorla D., Cuartero F. 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

[10] Macià H., Valero V., Cuartero F., Ruiz M.C. sPBC: a Markovian extension of Petri box calculus with immediate multiactions // Fundamenta Informaticae. – IOS Press, Amsterdam, The Netherlands, 2008. – Vol. 87, No. 3–4. – P. 367–406.

[11] Macià H., Valero V., de Frutos D. 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, Germany: IEEE Computer Society Press, 2001. – P. 207–216. – http://www.info-ab.uclm.es/retics/publications/2001/pnpm01.ps

[12] Tarasyuk I.V. Discrete Time Stochastic Petri Box Calculus. – Carl von Ossietzky Universität Oldenburg, Germany, 2005. – 25 p. – (Berichte aus dem Department für Informatik; Vol. 3/05). – http://db.iis.nsk.su/persons/itar/dtspbcib_cov.pdf

[13] Tarasyuk I.V. Iteration in discrete time stochastic Petri box calculus // Bull. Novosibirsk Comp. Center. Ser. Computer Science. – Novosibirsk, 2006. – Iss. 24. – P. 129–148. – http://db.iis.nsk.su/persons/itar/dtsitncc.pdf

[14] Tarasyuk I.V. Stochastic Petri box calculus with discrete time // Fundamenta Informaticae. – IOS Press, Amsterdam, The Netherlands, 2007. – Vol. 76, No. 1–2. – P. 189–218.

[15] Tarasyuk I.V. Performance preserving equivalences for dtsPBC // Bull. Novosibirsk Comp. Center. Ser. Computer Science. – Novosibirsk, 2010. – Iss. 31. – P. 155–178. – http://db.iis.nsk.su/persons/itar/dtspencc.pdf.

A. Proof of Proposition 2

Like it has been done for strong equivalence in [5], we shall prove the following fact about step stochastic bisimulation. Let $\forall j \in \mathcal{J}\ \mathcal{R}_j : G \underline{\leftrightarrow}_{ss} G'$ 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 $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$.

Since $\forall j \in \mathcal{J}\ \mathcal{R}_j$ is an equivalence, by definition of $\mathcal{R}$, we get that $\mathcal{R}$ is also an equivalence. Let $j \in \mathcal{J}$, then, by definition of $\mathcal{R}$, $(s_1, s_2) \in \mathcal{R}_j$ implies $(s_1, s_2) \in \mathcal{R}$. Hence, $\forall \mathcal{H}_{jk} \in (DR(G) \cup DR(G'))/_{\mathcal{R}_j}\ \exists \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}\ \mathcal{H}_{jk} \subseteq \mathcal{H}$. Moreover, $\exists \mathcal{J}'\ \mathcal{H} = \cup_{k\in\mathcal{J}'}\mathcal{H}_{jk}$.

We denote $\mathcal{R}(n) = (\cup_{j\in\mathcal{J}}\mathcal{R}_j)^n$. Let $(s_1, s_2) \in \mathcal{R}$, then, by definition of $\mathcal{R}$, $\exists n > 0\ (s_1, s_2) \in \mathcal{R}(n)$. Let us prove $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$ by induction on n.

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

It remains to prove that $(s_1, s_2) \in \mathcal{R}$ implies $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$ $\forall A \in I\!N_f^{\mathcal{L}}\ PM_A(s_1, \mathcal{H}) = PM_A(s_2, \mathcal{H})$.

- $n = 1$

  In this case, $(s_1, s_2) \in \mathcal{R}$ implies $\exists j \in \mathcal{J}\ (s_1, s_2) \in \mathcal{R}_j$. Since $\mathcal{R}_j : G \underline{\leftrightarrow}_{ss} G'$, we get $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}\ \forall A \in I\!N_f^{\mathcal{L}}$

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

- $n \to n + 1$

  Suppose that $\forall m \leq n\ (s_1, s_2) \in \mathcal{R}(m)$ implies $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}\ \forall A \in I\!N_f^{\mathcal{L}}\ PM_A(s_1, \mathcal{H}) = PM_A(s_2, \mathcal{H})$.

  Then $(s_1, s_2) \in \mathcal{R}(n + 1)$ implies $\exists j \in \mathcal{J}\ (s_1, s_2) \in \mathcal{R}_j \circ \mathcal{R}(n)$, i.e., $\exists s_3 \in (DR(G) \cup DR(G'))$ such that $(s_1, s_3) \in \mathcal{R}_j$ and $(s_3, s_2) \in \mathcal{R}(n)$.

  Then, like for the case $n = 1$, we get $PM_A(s_1, \mathcal{H}) = PM_A(s_3, \mathcal{H})$. By the induction hypothesis, $PM_A(s_3, \mathcal{H}) = PM_A(s_2, \mathcal{H})$. Thus, $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}\ \forall A \in I\!N_f^{\mathcal{L}}$

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

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 mentioned above that $\mathcal{R}_{ss}(G, G') : G \underline{\leftrightarrow}_{ss} G'$. □