S©MR

СИБИРСКИЕ ЭЛЕКТРОННЫЕ
МАТЕМАТИЧЕСКИЕ ИЗВЕСТИЯ
Siberian Electronic Mathematical Reports
http://semr.math.nsc.ru
ISSN 1813-3304

Vol. 21, No. x, pp. 1–56 (2024) УДК 004.423.4, 519.217.2, 519.681
https://doi.org/10.33048/semi.2024.21.xxx MSC 60J10, 60J20, 60K15, 68Q55, 68Q85

PERFORMANCE ANALYSIS OF THE SHARED MEMORY SYSTEM IN STOCHASTIC PROCESS ALGEBRA DTSDPBC

I.V. TARASYUK

Communicated by

Abstract: Petri box calculus (PBC) is a well-known algebra of parallel processes with a Petri net semantics. Discrete time stochastic and deterministic PBC (dtsdPBC) extends PBC with discrete time stochastic and deterministic delays. dtsdPBC has a step operational semantics via labeled probabilistic transition systems and a Petri net denotational semantics via dtsd-boxes, a subclass of labeled discrete time stochastic and deterministic Petri nets. To evaluate performance in dtsdPBC, the underlying semi-Markov chains (SMCs) and (reduced) discrete time Markov chains (DTMCs and RDTMCs) of the process expressions are analyzed. Step stochastic bisimulation equivalence that compares the qualitative and quantitative behaviour is used for quotienting the transition systems, SMCs, DTMCs and RDTMCs of the process expressions while preserving their stationary behaviour and residence time properties.

We construct in dtsdPBC a case study that demonstrates how the method of modeling, performance analysis and behaviour reduction by quotienting for concurrent systems with discrete fixed and stochastic delays is applied to the generalized shared memory system with maintenance. Such a generalized system takes as variables the probabilities and weights from the standard one's specification. Then the variable generalized probabilities of the reduced

Tarasyuk, I.V., Performance analysis of the shared memory system in stochastic process algebra dtsdPBC.

Received May, 2, 2023, Published Month, xx, 2024.

quotient DTMC are treated as parameters to be adjusted for the performance optimization of the modeled system.

Keywords: Petri box calculus, discrete time, stochastic and deterministic delays, transition system, operational semantics, dtsd-box, denotational semantics, Markov chain, performance, stochastic bisimulation, quotient, shared memory system, maintenance, generalized probabilities and weights.

1 Introduction

Process calculi, like CSP [16], ACP [4] and CCS [31] are well-known formal models for specification of computing systems and analysis of their behaviour. In such process algebras (PAs), formulas describe processes, and verification of the functionality properties of their behaviour is accomplished at a syntactic level via equivalences, axioms and inference rules. In order to represent stochastic timing and analyze the performance properties, stochastic extensions of PAs were proposed, like MTIPP [14], PEPA [15] and EMPA [7]. Such stochastic process algebras (SPAs) specify actions which can occur (qualitative features) and associate with the actions the distribution parameters of their random delays (quantitative characteristics).

1.1. Petri box calculus (PBC). Petri box calculus (PBC) [9, 11, 10, 8] is a flexible and expressive process algebra developed as a tool for specification of the Petri nets (PNs) 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 PNs. Formulas of PBC are combined from multisets of elementary actions and their conjugates, called multiactions (*basic formulas*). The empty multiset of actions is interpreted as the silent multiaction specifying an invisible activity. The operational semantics of PBC is of step type, since its SOS rules have transitions with (multi)sets of activities, corresponding to simultaneous executions of activities (steps). A denotational semantics of PBC was proposed via a subclass of PNs with an interface and considered up to isomorphism, called Petri boxes. The extensions of PBC with a deterministic, a nondeterministic or a stochastic model of time exist.

1.2. Time extensions of PBC. A time extension of PBC with a nondeterministic time model, called time Petri box calculus (tPBC), was proposed in [18]. In tPBC, timing information is added by associating time intervals with instantaneous *actions*. tPBC has a step time operational semantics in terms of labeled transition systems. Its denotational semantics was defined in terms of a subclass of labeled time Petri nets (LtPNs), based on tPNs [30] and called time Petri boxes (ct-boxes).

Another time enrichment of PBC, called Timed Petri box calculus (TPBC), was defined in [26, 27], it accommodates a deterministic model of time. In

contrast to tPBC, multiactions of TPBC are not instantaneous, but have time durations. TPBC has a step timed operational semantics in terms of labeled transition systems. The denotational semantics of TPBC was defined in terms of a subclass of labeled Timed Petri nets (LTPNs), based on TPNs [36] and called Timed Petri boxes (T-boxes).

The third time extension of PBC, called arc time Petri box calculus (atPBC), was constructed in [34, 35], and it implements a nondeterministic time. In atPBC, multiactions are associated with time delay intervals. atPBC possesses a step time operational semantics in terms of labeled transition systems. Its denotational semantics was defined on a subclass of labeled arc time Petri nets (atPNs), based of those from [12, 13], where time restrictions are associated with the arcs, called arc time Petri boxes (at-boxes). tPBC, TPBC and atPBC, all adapt the discrete time approach, but TPBC has no immediate (multi)actions (those with zero delays).

1.3. Stochastic extensions of PBC. A stochastic extension of PBC, called stochastic Petri box calculus (sPBC), was proposed in [25, 21, 22]. In sPBC, multiactions have stochastic delays that follow (negative) exponential distribution. Each multiaction is equipped with a rate that is a parameter of the corresponding exponential distribution. The (instantaneous) execution of a stochastic multiaction is possible only after the corresponding stochastic time delay. The calculus has an interleaving operational semantics defined via transition systems labeled with multiactions and their rates. Its denotational semantics was defined on a subclass of labeled continuous time stochastic PNs, based on CTSPNs [28, 2] and called stochastic Petri boxes (s-boxes).

sPBC was enriched with immediate multiactions having zero delay in [23, 24]. We call such an extension generalized sPBC (gsPBC). An interleaving operational semantics of gsPBC was constructed via transition systems labeled with stochastic or immediate multiactions together with their rates or probabilities. A denotational semantics of gsPBC was defined via a subclass of labeled generalized stochastic PNs, based on GSPNs [28, 2, 3] and called generalized stochastic Petri boxes (gs-boxes).

In [37, 38, 39, 40], we presented a discrete time stochastic extension dtsPBC of the algebra PBC. In dtsPBC, the residence time in the process states is geometrically distributed. A step operational semantics of dtsPBC was constructed via labeled probabilistic transition systems. Its denotational semantics was defined in terms of a subclass of labeled discrete time stochastic PNs (LDTSPNs), based on DTSPNs [32, 33] and called discrete time stochastic Petri boxes (dts-boxes).

In [44, 45, 46, 47, 48], a calculus dtsiPBC was proposed as an extension with immediate multiactions of dtsPBC. Immediate multiactions increase the specification capability: they can model logical conditions, probabilistic branching, instantaneous probabilistic choices and activities whose durations are negligible in comparison with those of others. They are also used to specify urgent activities and the ones that are not relevant for performance

evaluation. The step operational semantics of dtsiPBC was constructed with the use of labeled probabilistic transition systems. Its denotational semantics was defined in terms of a subclass of labeled discrete time stochastic and immediate PNs (LDTSIPNs), called dtsi-boxes.

In [41, 42, 43], we defined dtsdPBC, an extension of dtsiPBC with deterministic multiactions. In dtsdPBC, besides the probabilities from the real-valued interval $(0; 1)$, applied to calculate discrete time delays of stochastic multiactions, also non-negative integers are used to specify fixed delays of deterministic multiactions (including zero delay, which is the case of immediate multiactions). To resolve conflicts among deterministic multiactions, they are additionally equipped with positive real-valued weights. As argued in [51, 49, 50], a combination of deterministic and stochastic delays fits well to model technical systems with constant (fixed) durations of the regular non-random activities and probabilistically distributed (stochastic) durations of the randomly occurring activities. dtsdPBC has a step operational semantics, defined via labeled probabilistic transition systems. The denotational semantics of dtsdPBC was defined in terms of a subclass of labeled discrete time stochastic and deterministic Petri nets (LDTSDPNs), called dtsd-boxes.

1.4. Our contributions. As a basis model, we take *discrete time stochastic and deterministic Petri box calculus* (dtsdPBC), presented in [41, 42, 43], featuring a step operational semantics. Here we do not consider the Petri net denotational semantics of the calculus, since it was extensively described in [41]. In that paper, a consistency of the operational and denotational semantics with respect to step stochastic bisimulation equivalence was proved. Hence, all the results established for the former can be readily transferred to the latter up to that equivalence.

In [42], with the *embedding* method, based on the embedded DTMC (EDTMC) specifying the state change probabilities, we constructed and solved the underlying stochastic process, which is a semi-Markov chain (SMC). The obtained stationary probability masses and average sojourn times in the states of the SMC were used to calculate the performance measures (indices) of interest. The alternative solution techniques were also developed, called *abstraction* and *elimination*, that are based respectively on the corresponding discrete time Markov chain (DTMC) and its reduction (RDTMC) by eliminating vanishing states (those with zero sojourn times).

In [41, 43], we proposed step stochastic bisimulation equivalence to identify algebraic processes with similar qualitative and quantitative behaviour. We established consistency of the operational and denotational semantics of dtsdPBC up to that equivalence. We examined the interrelations of the proposed notion with other equivalences of the algebra. The introduced equivalence was applied to reduce (by quotienting) the transition systems, SMCs, DTMCs and RDTMCs of the process expressions while preserving their qualitative and quantitative characteristics. We proved that the equivalence guarantees identity of the stationary behaviour and residence time

properties in the equivalence classes. This implies coincidence of the performance indices based on the steady-state probabilities and sojourn time averages for the complete and quotient behaviour.

In the present paper, we describe a case study of a system consisting of two processors and a common shared memory with maintenance that explains how to model concurrent systems within the calculus and analyze their performance, as well as how to reduce the systems behaviour while preserving their performance indices and making easier the performance evaluation. We study a generalized variant of the shared memory system by treating the probabilities and weights from the standard system's specification as variables (parameters) that possess general values. First, we consider a process expression of the concrete system that differentiates among the processors, and then we analyze its functionality and performance using the corresponding transition system, SMC, DTMC and RDTMC. Second, we model the abstract system that abstracts from the names of the processors (by making identical the actions from their specifications), to apply reduction by the equivalence. The quotients of the abstract system's behaviour (represented by the transition system, SMC, DTMC and RDTMC) by the step stochastic bisimulation equivalence are constructed. Third, the generalized probabilities of the reduced quotient DTMC (coinciding with the quotient RDTMC) are treated as parameters (or variables of the performance index functions) to be adjusted for the abstract system's performance optimization.

Thus, the main contributions of the paper are as follows.

- Performance analysis of the concrete generalized shared memory system with maintenance using its transition system, SMC, DTMC and RDTMC.
- Performance analysis of the abstract system via its quotient (by step stochastic bisimulation equivalence) transition system, SMC, DTMC, RDTMC.
- Performance optimization by adjusting the quotient RDTMC probabilities, treated as parameters of the abstract system's performance index functions.

1.5. Structure of the paper. In Section 2, the syntax of algebra dtsdPBC is proposed. In Section 3, the operational semantics of the calculus in terms of labeled probabilistic transition systems is presented. Step stochastic bisimulation equivalence is defined and investigated in Section 4. In Section 5, the equivalence quotients of the transition systems and corresponding Markov chains of the process expressions are constructed. In Section 6, the introduced equivalence is proved to preserve the stationary behaviour and residence time properties in the equivalence classes. In Section 7, the generalized shared memory system with maintenance is presented as a case study. Section 8 summarizes the results obtained and outlines future research.

2 Syntax

In this section, we define the syntax: activities, operations and expressions.

2.1. Activities and operations. Multiset allows identical elements in a set.

Definition 1. *Let X be a set. A finite* multiset (bag) *M over X is a mapping $M:X\to\mathbb{N}$ with $|\{x\in X\,|\,M(x)>0\}|<\infty$, i.e. it has a finite number of elements.*

The *set of all finite multisets* over a set X is $\mathbb{N}^X_{fin}$. Let $M, M'\in\mathbb{N}^X_{fin}$. The *cardinality* of M is $|M| = \sum_{x\in X} M(x)$. We write $x\in M$ if $M(x)>0$ and $M\subseteq M'$ if $\forall x\in X\ M(x)\le M'(x)$. We define $(M+M')(x) = M(x)+M'(x)$ and $(M-M')(x) = \max\{0, M(x)-M'(x)\}$. When $\forall x\in X,\ M(x)\le 1,\ M$ is seen as a proper set $M\subseteq X$. The *set of all subsets (powerset)* of X is 2^X.

Let $Act=\{a,b,\ldots\}$ be the set of *elementary actions*. Then $\widehat{Act}=\{\hat{a},\hat{b},\ldots\}$ is the set of *conjugated actions (conjugates)* with $\hat{a}\ne a$ and $\hat{\hat{a}}=a$. Let $\mathcal{A}=Act\cup\widehat{Act}$ be the set of *all actions*, and $\mathcal{L}=\mathbb{N}^{\mathcal{A}}_{fin}$ be the set of *all multiactions*. Here $\emptyset\in\mathcal{L}$ specifies an internal move, i.e. the execution of a multiaction without visible actions. The *alphabet* of $\alpha\in\mathcal{L}$ is $\mathcal{A}(\alpha)=\{x\in\mathcal{A}\mid\alpha(x)>0\}$.

A *stochastic multiaction* is a pair (α,ρ), where $\alpha\in\mathcal{L}$ and $\rho\in(0;1)$ is the *probability* of the multiaction α. This probability is interpreted as that of independent execution of the stochastic multiaction at the next discrete time moment. Such probabilities are used to calculate those to execute (possibly empty) sets of stochastic multiactions after one time unit delay. The probability 1 is left for (implicitly assigned to) waiting multiactions, i.e. positively delayed deterministic multiactions (to be defined later), which have weights to resolve conflicts with other waiting multiactions. Let $\mathcal{SL}$ be the set of *all stochastic multiactions*.

A *deterministic multiaction* is a pair $(\alpha,\natural^\theta_l)$, where $\alpha\in\mathcal{L}$, $\theta\in\mathbb{N}$ is the non-negative integer-valued *(fixed) delay* and $l\in\mathbb{R}_{>0}=(0;\infty)$ is the positive real-valued *weight* of the multiaction α. This weight is interpreted as a measure of importance (urgency, interest) or a bonus reward associated with execution of the deterministic multiaction at the moment when the corresponding delay has expired. Such weights are used to calculate the probabilities to execute sets of deterministic multiactions after their delays. An *immediate multiaction* is a deterministic multiaction with the delay 0 while a *waiting multiaction* is a deterministic multiaction with a positive delay. In case of no conflicts among waiting multiactions, whose remaining times to execute (RTEs) are equal to one time unit, they are executed with probability 1 at the next moment. Deterministic multiactions have a priority over stochastic ones while immediate multiactions have a priority over waiting ones. Different types of multiactions cannot participate together in some step (parallel execution). Let $\mathcal{DL}$ be the set of *all deterministic multiactions*, $\mathcal{IL}$ be the set of *all immediate multiactions* and $\mathcal{WL}$ be the set of *all waiting multiactions*. We have $\mathcal{DL}=\mathcal{IL}\cup\mathcal{WL}$.

The same multiaction $\alpha \in \mathcal{L}$ may have different probabilities, (fixed) delays and weights in the same specification. An *activity* is a stochastic or a deterministic multiaction. Let $\mathcal{SDL} = \mathcal{SL} \cup \mathcal{DL} = \mathcal{SL} \cup \mathcal{IL} \cup \mathcal{WL}$ be the set of *all activities*. The *alphabet* of an activity $(\alpha, \kappa) \in \mathcal{SDL}$ is $\mathcal{A}(\alpha, \kappa) = \mathcal{A}(\alpha)$. The *alphabet* of a multiset of activities $\Upsilon \in \mathbb{N}_{fin}^{\mathcal{SDL}}$ is $\mathcal{A}(\Upsilon) = \cup_{(\alpha,\kappa)\in\Upsilon}\mathcal{A}(\alpha)$.

Activities are combined into formulas (process expressions) by the operations of *sequence* ;, *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.

Sequence (sequential composition) and choice (composition) have a standard interpretation, like in other PAs, but parallelism (parallel composition) does not include synchronization, unlike the corresponding operation in CCS.

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: for $\alpha \in \mathcal{L}$ we define $f(\alpha) = \sum_{x\in\alpha} f(x) = \sum_{x\in\mathcal{A}} \alpha(x) f(x)$. Relabeling is extended to activities: for $(\alpha, \kappa) \in \mathcal{SDL}$ we define $f(\alpha, \kappa) = (f(\alpha), \kappa)$. Relabeling is extended to the multisets of activities: for $\Upsilon \in \mathbb{N}_{fin}^{\mathcal{SDL}}$ we define $f(\Upsilon) = \sum_{(\alpha,\kappa)\in\Upsilon} (f(\alpha), \kappa)$.

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

Let $\alpha, \beta \in \mathcal{L}$ be two multiactions such that for some elementary action $a \in Act$ we have $a \in \alpha$ and $\hat{a} \in \beta$, or $\hat{a} \in \alpha$ and $a \in \beta$. Then, synchronization of α and β by a is defined as $(\alpha \oplus_a \beta)(x) = \begin{cases} \alpha(x) + \beta(x) - 1, & x = a \text{ or } x = \hat{a}; \\ \alpha(x) + \beta(x), & \text{otherwise.} \end{cases}$
Activities are synchronized via their multiaction parts, i.e. the synchronization by a of two activities, whose multiaction parts α and β possess the above properties, results in the activity with the multiaction part $\alpha \oplus_a \beta$. We may synchronize activities of the same type only: either both stochastic multiactions or both deterministic ones *with the same delay*, since stochastic, waiting and immediate multiactions have different priorities, and diverse delays of waiting multiactions would contradict their joint timing. Note that the execution of immediate multiactions takes no time, unlike that of waiting or stochastic ones. Synchronization by a means that, for a given expression with a process behaviour containing two concurrent activities that can be synchronized by a, there exists also the behaviour that differs from the former only in that the two activities are replaced by the result of their synchronization.

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

2.2. Process expressions. Static expressions specify the structure of processes, i.e. how activities are combined by operations to construct the composite process-algebraic formulas. As for the PN intuition, static expressions correspond to unmarked LDTSDPNs [41]. A marking is the allocation of tokens in the places of a PN. Markings are used to describe dynamic behaviour of PNs in terms of transition firings.

We assume that every waiting multiaction has a countdown timer associated, whose value is the time left till the moment when the waiting multiaction can be executed. Therefore, besides standard (unstamped) waiting multiactions $(\alpha, \natural_l^\theta) \in \mathcal{WL}$, a special case of the *stamped* waiting multiactions should be considered in the definition of static expressions. Each (time) stamped waiting multiaction $(\alpha, \natural_l^\theta)^\delta$ has an extra superscript $\delta \in \{1, \ldots, \theta\}$ that specifies a time stamp indicating the *latest* value of the timer associated with that multiaction. The standard waiting multiactions have no time stamps, to demonstrate irrelevance of the timer values for them (for example, their timers have not yet started or have already finished). The notion of the alphabet part for (the multisets of) stamped waiting multiactions is defined like that for (the multisets of) unstamped waiting multiactions.

For simplicity, we do not assign the timer value superscripts δ to immediate multiactions, a special case of deterministic multiactions $(\alpha, \natural_l^\theta)$ with the delay $\theta = 0$ in the form of $(\alpha, \natural_l^0)$, since their timer values always equal to 0.

Definition 2. *Let* $(\alpha, \kappa) \in \mathcal{SDL}$, $(\alpha, \natural_l^\theta) \in \mathcal{WL}$, $\delta \in \{1, \ldots, \theta\}$ *and* $a \in Act$. *A* static expression *of dtsdPBC is*

$$E ::= (\alpha, \kappa) \mid (\alpha, \natural_l^\theta)^\delta \mid E; E \mid E[]E \mid E \| E \mid E[f] \mid E \,\mathsf{rs}\, a \mid E \,\mathsf{sy}\, a \mid [E{*}E{*}E].$$

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

To avoid technical difficulties with the iteration operator, we should not allow concurrency at the highest level of the second argument of iteration. This is not a severe restriction, since we can always prefix parallel expressions by an activity with the empty multiaction part. Relaxing the restriction can result in LDTSDPNs [41] which are not safe, like shown for PNs in [10]. A PN is *n-bounded* ($n \in \mathbb{N}$) if for all its reachable (from the initial marking by the sequences of transition firings) markings there are at most n tokens in every place, and a PN is *safe* if it is 1-bounded.

Definition 3. *Let* $(\alpha, \kappa) \in \mathcal{SDL}$, $(\alpha, \natural_l^\theta) \in \mathcal{WL}$, $\delta \in \{1, \ldots, \theta\}$ *and* $a \in Act$. *A* regular static expression *of dtsdPBC is*

$$E ::= (\alpha, \kappa) \,|\, (\alpha, \natural_l^\theta)^\delta \,|\, E; E \,|\, E[]E \,|\, E \| E \,|\, E[f] \,|\, E \,\mathsf{rs}\, a \,|\, E \,\mathsf{sy}\, a \,|\, [E * D * E],$$
$$\textit{where } D ::= (\alpha, \kappa) \,|\, (\alpha, \natural_l^\theta)^\delta \,|\, D; E \,|\, D[]D \,|\, D[f] \,|\, D \,\mathsf{rs}\, a \,|\, D \,\mathsf{sy}\, a \,|\, [D * D * E].$$

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

Let E be a regular static expression. The *underlying timer-free regular static expression* $\lfloor E$ of E is obtained by removing all timer value superscripts.

The set of *all stochastic multiactions (from the syntax) of* E is $\mathcal{SL}(E) = \{(\alpha, \rho) \mid (\alpha, \rho)$ is a subexpression of $E\}$. The set of *all immediate multiactions (from the syntax) of* E is $\mathcal{IL}(E) = \{(\alpha, \natural_l^0) \mid (\alpha, \natural_l^0)$ is a subexpression of $E\}$. The set of *all waiting multiactions (from the syntax) of* E is $\mathcal{WL}(E) = \{(\alpha, \natural_l^\theta) \mid (\alpha, \natural_l^\theta)$ or $(\alpha, \natural_l^\theta)^\delta$ is a subexpression of E for $\delta \in \{1, \ldots, \theta\}\}$. Thus, the set of *all deterministic multiactions (from the syntax) of* E is $\mathcal{DL}(E) = \mathcal{IL}(E) \cup \mathcal{WL}(E)$ and the set of *all activities (from the syntax) of* E is $\mathcal{SDL}(E) = \mathcal{SL}(E) \cup \mathcal{DL}(E) = \mathcal{SL}(E) \cup \mathcal{IL}(E) \cup \mathcal{WL}(E)$.

Dynamic expressions specify the states of processes, i.e. particular stages of the process behaviour. As for the Petri net intuition, dynamic expressions correspond to marked LDTSDPNs [41]. Dynamic expressions are obtained from static ones, by annotating them with upper or lower bars which specify the active components of the system at the current moment of time. The dynamic expression with upper bar (the overlined one) $\overline{E}$ denotes the *initial*, and that with lower bar (the underlined one) $\underline{E}$ denotes the *final* state of the process specified by a static expression E.

For every overlined stamped waiting multiaction $\overline{(\alpha, \natural_l^\theta)^\delta}$, the superscript $\delta \in \{1, \ldots, \theta\}$ specifies the *current* value of the *running* countdown timer associated with the waiting multiaction. That decreasing discrete timer is started with the *initial* value θ (the waiting multiaction delay) at the moment when the waiting multiaction becomes overlined. Then such a newly overlined stamped waiting multiaction $\overline{(\alpha, \natural_l^\theta)^\theta}$ is similar to the freshly overlined unstamped waiting multiaction $\overline{(\alpha, \natural_l^\theta)}$. Such similarity will be captured by the structural equivalence, defined later.

While the stamped waiting multiaction stays overlined with the process execution, the timer decrements by one discrete time unit with each global time tick until the timer value becomes 1. This means that one unit of time remains till execution of that multiaction (the remaining time to execute, RTE, equals one). Its execution should follow in the next moment with probability 1, in case there are no conflicting with it immediate multiactions or conflicting waiting multiactions whose RTEs equal to one, and it is not affected by restriction. An activity is affected by restriction, if it is within the scope of a restriction operation with the argument action, such that it or its conjugate is contained in the multiaction part of that activity.

Definition 4. *Let $E \in StatExpr$, $a \in Act$. A* dynamic expression *of dtsdPBC is*

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

Let $DynExpr$ denote the set of *all dynamic expressions* of dtsdPBC.

Let G be a dynamic expression. The *underlying static (line-free) expression* $\lfloor G \rfloor$ of G is obtained by removing from it all upper and lower bars. If the underlying static expression of a dynamic one is not regular, the corresponding LDTSDPN can be non-safe [41] (but it is 2-bounded in the worst case, like shown for PNs in [10]).

Definition 5. *A dynamic expression G is* regular *if $\lfloor G \rfloor$ is regular.*

$RegDynExpr$ denotes the set of *all regular dynamic expressions* of dtsdPBC.

Let G be a regular dynamic expression. The *underlying timer-free regular dynamic expression* $\lfloor\!\lfloor G$ of G is got by removing all timer value superscripts.

The set of *all stochastic (immediate or waiting, respectively) multiactions (from the syntax) of* G is defined as $\mathcal{SL}(G) = \mathcal{SL}(\lfloor G \rfloor)$ ($\mathcal{IL}(G) = \mathcal{IL}(\lfloor G \rfloor)$

or $\mathcal{WL}(G) = \mathcal{WL}(\lfloor G \rfloor)$, respectively). Thus, the set of *all deterministic multiactions (from the syntax) of* G is $\mathcal{DL}(G) = \mathcal{IL}(G) \cup \mathcal{WL}(G)$ and the set of *all activities (from the syntax) of* G is $\mathcal{SDL}(G) = \mathcal{SL}(G) \cup \mathcal{DL}(G) = \mathcal{SL}(G) \cup \mathcal{IL}(G) \cup \mathcal{WL}(G)$.

3 Operational semantics

In this section, we define the operational semantics via transition systems.

3.1. Inaction rules. The inaction rules for dynamic expressions describe their structural transformations in the form of $G \Rightarrow \widetilde{G}$ which do not change the states of the specified processes. The goal of those syntactic transformations is to obtain the well-structured resulting expressions called operative ones to which no inaction rules can be further applied. The application of an inaction rule to a dynamic expression does not lead to any discrete time tick or any transition firing in the corresponding LDTSDPN [41], hence, its current marking stays unchanged.

An application of every inaction rule does not require a 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 being overlined and underlined static ones, where $(\alpha, \natural_l^\theta) \in \mathcal{WL}$, $\delta \in \{1, \ldots, \theta\}$, $E, F, K \in RegStatExpr$ and $a \in Act$. The first inaction rule suggests that the timer value of each newly overlined waiting multiaction is set to its delay.

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

$\overline{(\alpha, \natural_l^\theta)} \Rightarrow \overline{(\alpha, \natural_l^\theta)^\theta}$	$\overline{E; F} \Rightarrow \overline{E}; F$	$\underline{E}; F \Rightarrow E; \overline{F}$
$E; \underline{F} \Rightarrow \underline{E; F}$	$\overline{E[]F} \Rightarrow \overline{E}[]F$	$\overline{E[]F} \Rightarrow E[]\overline{F}$
$\underline{E}[]F \Rightarrow \underline{E[]F}$	$E[]\underline{F} \Rightarrow \underline{E[]F}$	$\overline{E \| F} \Rightarrow \overline{E} \| \overline{F}$
$\underline{E} \| \underline{F} \Rightarrow \underline{E \| F}$	$\overline{E[f]} \Rightarrow \overline{E}[f]$	$\underline{E}[f] \Rightarrow \underline{E[f]}$
$\overline{E \text{ rs } a} \Rightarrow \overline{E} \text{ rs } a$	$\underline{E} \text{ rs } a \Rightarrow \underline{E \text{ rs } a}$	$\overline{E \text{ sy } a} \Rightarrow \overline{E} \text{ sy } a$
$\underline{E} \text{ sy } a \Rightarrow \underline{E \text{ sy } a}$	$\overline{[E * F * K]} \Rightarrow [\overline{E} * F * K]$	$[\underline{E} * F * K] \Rightarrow [E * \overline{F} * K]$
$[E * \underline{F} * K] \Rightarrow [E * \overline{F} * K]$	$[E * \underline{F} * K] \Rightarrow [E * F * \overline{K}]$	$[E * F * \underline{K}] \Rightarrow \underline{[E * F * K]}$

In Table 2, we introduce inaction rules for regular dynamic expressions in the arbitrary form, where $E, F \in RegStatExpr$, $G, H, \widetilde{G}, \widetilde{H} \in RegDynExpr$ and $a \in Act$. For brevity, two distinct inaction rules with the same premises are sometimes collated, resulting in the inaction rules with double conclusion.

Definition 6. *A regular dynamic expression* G *is* operative *if no inaction rule can be applied to it.*

Let $OpRegDynExpr$ denote the set of *all operative regular dynamic expressions* of dtsdPBC. Any dynamic expression can be always transformed into a (not necessarily unique) operative one by using the inaction rules.

TABLE 2. Inaction rules for arbitrary regular dynamic expressions

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

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

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

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

Definition 7. *The relation* $\approx = (\Rightarrow \cup \Leftarrow)^*$ *is a* structural equivalence *of dynamic expressions in dtsdPBC. 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 a backward direction.*

Let G be a dynamic expression. Then $[G]_\approx = \{H \mid G \approx H\}$ is the equivalence class of G with respect to the structural equivalence, called the (corresponding) *state*. Next, G is an *initial* dynamic expression, denoted by $init(G)$, if $\exists E \in RegStatExpr\ G \in [\overline{E}]_\approx$. Further, G is a *final* dynamic expression, denoted by $final(G)$, if $\exists E \in RegStatExpr\ G \in [\underline{E}]_\approx$.

Let G be a dynamic expression and $s = [G]_\approx$. The set of *all enabled stochastic multiactions of* s is $EnaSto(s) = \{(\alpha, \rho) \in \mathcal{SL} \mid \exists H \in s \cap OpRegDynExpr\ \overline{(\alpha, \rho)}$ is a subexpression of $H\}$. The set of *all enabled immediate multiactions of* s is $EnaImm(s) = \{(\alpha, \natural_l^0) \in \mathcal{IL} \mid \exists H \in s \cap OpRegDynExpr\ \overline{(\alpha, \natural_l^0)}$ is a subexpression of $H\}$. The set of *all enabled waiting multiactions of* s is $EnaWait(s) = \{(\alpha, \natural_l^\theta) \in \mathcal{WL} \mid \exists H \in s \cap OpRegDynExpr\ \overline{(\alpha, \natural_l^\theta)^\delta},\ \delta \in \{1, \ldots, \theta\},$ is a subexpression of $H\}$. The set of *all newly enabled waiting multiactions of* s is $EnaWaitNew(s) = \{(\alpha, \natural_l^\theta) \in \mathcal{WL} \mid \exists H \in s \cap OpRegDynExpr\ \overline{(\alpha, \natural_l^\theta)^\theta}$ is a subexpression of $H\}$.

The set of *all enabled deterministic multiactions of* s is $EnaDet(s) = EnaImm(s) \cup EnaWait(s)$ and the set of *all enabled activities of* s is $Ena(s) = EnaSto(s) \cup EnaDet(s) = EnaSto(s) \cup EnaImm(s) \cup EnaWait(s)$. Then $Ena(s) = Ena([G]_\approx)$ is an algebraic analogue of the set of all transitions enabled at the initial marking of the LDTSDPN [41] corresponding to G. The activities, resulted from synchronization, are not present in the syntax of the dynamic expressions. Their enabledness status can be recovered by observing that of the pair of synchronized activities from the syntax (they both should be enabled for enabling their synchronous product), even if they are affected by restriction after the synchronization.

Definition 8. *An operative dynamic expression* G *is* saturated (with the values of timers)*, if each enabled waiting multiaction of* $[G]_\approx$*, being superscribed with the value of its timer and possibly overlined, is the subexpression of* G*.*

Let *SaOpRegDynExpr* denote the set of *all saturated operative dynamic expressions* of dtsdPBC.

Proposition 1. *Any operative dynamic expression can be always transformed into the saturated one by a forward or a backward applying the inaction rules.*

Proof. See [41]. □

Thus, any dynamic expression can be transformed into a (not always unique) saturated operative one by (possibly reverse) applying the inaction rules.

Let G be a saturated operative dynamic expression. Then $\circlearrowleft G$ denotes the *timer decrement* operator $\circlearrowleft$, applied to G. The result is a saturated operative dynamic expression, obtained from G via decrementing by one all greater than 1 values of the timers associated with all (if any) stamped waiting multiactions from the syntax of G. Each such stamped waiting multiaction changes its timer value from $\delta \in \mathbb{N}_{\geq 1}$ in G to $\max\{1, \delta - 1\}$ in $\circlearrowleft G$. The timer decrement operator affects the (possibly overlined or underlined) stamped waiting multiactions being the subexpressions of G as: $(\alpha, \natural_l^\theta)^\delta$ is replaced with $(\alpha, \natural_l^\theta)^{\max\{1,\delta-1\}}$, and similarly for the overlined or underlined ones.

Note that when $\delta = 1$, we have $\max\{1, \delta - 1\} = \max\{1, 0\} = 1$, hence, the timer value $\delta = 1$ may remain unchanged for a stamped waiting multiaction that is not executed by some reason at the next time moment, but stays stamped. For example, that stamped waiting multiaction may be affected by restriction. If the timer values cannot be decremented with a time tick for all stamped waiting multiactions (if any) from G then $\circlearrowleft G = G$ and we obtain so-called *empty loop* transition, defined later.

The timer decrement operator keeps stamping of the waiting multiactions, since it may only decrease their timer values, and the stamped waiting multiactions stay stamped (with their timer values, possibly decremented by one).

3.2. Action and empty move rules. The action rules are applied when some activities are executed. With these rules we capture the prioritization among different types of multiactions. We also have the empty move rule, used to capture a delay of one discrete time unit when no immediate or waiting multiactions are executable. In this case, the empty multiset of activities is executed. The action and empty move rules will be used later to determine all multisets of activities which can be executed from the structural equivalence class of every dynamic expression (i.e. from the state of the corresponding process). This information together with that about probabilities or delays and weights of the activities to be executed from the current process state will be used to calculate the probabilities of such executions.

The action rules with stochastic (immediate or waiting, respectively) multiactions describe dynamic expression transformations in the form of $G \xrightarrow{\Gamma} \widetilde{G}$ ($G \xrightarrow{I} \widetilde{G}$ or $G \xrightarrow{W} \widetilde{G}$, respectively) due to execution of non-empty multisets Γ of stochastic (I of immediate or W of waiting, respectively) multiactions.

The rules represent possible state changes of the specified processes when some non-empty multisets of stochastic (immediate or waiting, respectively) multiactions are executed. The application of an action rule with stochastic (immediate or waiting, respectively) multiactions to a dynamic expression leads in the corresponding LDTSDPN [41] to a discrete time tick at which some stochastic or waiting transitions fire (or to the instantaneous firing of some immediate transitions) and possible change of the current marking. The current marking stays unchanged only if there is a self-loop produced by the iterative execution of a non-empty multiset, being one-element, since we allow no concurrency at the highest level of the second argument of iteration.

The empty move rule (applicable only when no immediate or waiting multiactions can be executed from the current state) describes dynamic expression transformations in the form of $G \xrightarrow{\emptyset} \circlearrowleft G$, called the *empty moves*, due to execution of the empty multiset of activities at a discrete time tick. When no timer values are decremented within G with the empty multiset execution at the next moment (for example, if G contains no stamped waiting multiactions), we have $\circlearrowleft G = G$. In such a case, the empty move from G is in the form of $G \xrightarrow{\emptyset} G$, called the *empty loop*. The application of the empty move rule to a dynamic expression leads to a discrete time tick in the corresponding LDTSDPN [41] at which no transitions fire and the current marking is not changed, but the timer values of the waiting transitions enabled at the marking (if any) are decremented by one. This is a new rule that has no prototype among inaction rules of PBC, since it represents a time delay.

Thus, an application of every action rule with stochastic or waiting multiactions or the empty move rule requires one discrete time unit delay, i.e. the execution of a (possibly empty) multiset of stochastic or (non-empty) multiset of waiting multiactions leading to the dynamic expression transformation described by the rule is accomplished instantly after one time unit. An application of every action rule with immediate multiactions does not take any time, i.e. the execution of a (non-empty) multiset of immediate multiactions is accomplished instantly at the current moment.

The expressions of dtsdPBC can contain identical activities. To avoid technical difficulties, such as calculation of the probabilities for multiple transitions, we can enumerate coinciding activities from left to right in the syntax of expressions. The new activities, resulted 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. We now define the numbering which encodes a binary tree with the leaves labeled by natural numbers.

Definition 9. *The* numbering *of expressions is* $\iota ::= n \mid (\iota)(\iota)$, *where* $n \in \mathbb{N}$.

Let Num denote the set of *all numberings* of expressions.

The new activities resulting from synchronizations 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 identify the mentioned instances. The *content* of a numbering $\iota \in Num$ is $Cont(\iota) = \begin{cases} \{\iota\}, & \iota \in \mathbb{N}; \\ Cont(\iota_1) \cup Cont(\iota_2), & \iota = (\iota_1)(\iota_2). \end{cases}$
After the enumeration, the multisets of activities from the expressions become proper sets. We suppose that the identical activities are enumerated when needed to avoid ambiguity. This enumeration is considered to be implicit.

Definition 10. *Let $G \in OpRegDynExpr$. We define $Can(G)$, the* set of all non-empty multisets of activities which can be potentially executed from G. *Let $(\alpha,\kappa)\in\mathcal{SDL}$, $E,F\in RegStatExpr$, $H\in OpRegDynExpr$ and $a\in Act$.*

(1) *If $final(G)$ then $Can(G)=\emptyset$.*
(2) *If $G=\overline{(\alpha,\kappa)}^{\delta}$ and $\kappa=\natural_l^\theta$, $\theta\in\mathbb{N}_{\geq 2}$, $l\in\mathbb{R}_{>0}$, $\delta\in\{2,\ldots,\theta\}$, then $Can(G)=\emptyset$.*
(3) *If $G=\overline{(\alpha,\kappa)}$ and $\kappa\in(0;1)$ or $\kappa=\natural_l^0$, $l\in\mathbb{R}_{>0}$, then $Can(G)=\{\{(\alpha,\kappa)\}\}$.*
(4) *If $G=\overline{(\alpha,\kappa)}^{1}$ and $\kappa=\natural_l^\theta$, $\theta\in\mathbb{N}_{\geq 1}$, $l\in\mathbb{R}_{>0}$, then $Can(G)=\{\{(\alpha,\kappa)\}\}$.*
(5) *If $\Upsilon\in Can(G)$ then $\Upsilon\in Can(G\circ E)$, $\Upsilon\in Can(E\circ G)$ $(\circ\in\{;,[]\})$, $\Upsilon\in Can(G\|H)$, $\Upsilon\in Can(H\|G)$, $f(\Upsilon)\in Can(G[f])$, $\Upsilon\in Can(G\ \mathsf{rs}\ a)$ $(when\ a,\hat{a}\notin\mathcal{A}(\Upsilon))$, $\Upsilon\in Can(G\ \mathsf{sy}\ a)$, $\Upsilon\in Can([G*E*F])$, $\Upsilon\in Can([E*G*F])$, $\Upsilon\in Can([E*F*G])$.*
(6) *If $\Upsilon\in Can(G)$ and $\Xi\in Can(H)$ then $\Upsilon+\Xi\in Can(G\|H)$.*
(7) *If $\Upsilon\in Can(G\ \mathsf{sy}\ a)$ and $(\alpha,\kappa),(\beta,\lambda)\in\Upsilon$ are different, $a\in\alpha$, $\hat{a}\in\beta$, then*
   (a) *$\Upsilon-\{(\alpha,\kappa),(\beta,\lambda)\}+\{(\alpha\oplus_a\beta,\kappa\cdot\lambda)\}\in Can(G\ \mathsf{sy}\ a)$ if $\kappa,\lambda\in(0;1)$;*
   (b) *$\Upsilon-\{(\alpha,\kappa),(\beta,\lambda)\}+\{(\alpha\oplus_a\beta,\natural_{l+m}^\theta)\}\in Can(G\ \mathsf{sy}\ a)$ if $\kappa=\natural_l^\theta$, $\lambda=\natural_m^\theta$, $\theta\in\mathbb{N}$, $l,m\in\mathbb{R}_{>0}$.*
   When we synchronize a multiset of activities in different orders, we get several activities with the same multiaction and probability or delay and weight parts, but different numberings with the same content. Then we only consider a single resulting activity.

If $\Upsilon\in Can(G)$ then by definition of $Can(G)$, $\forall\Xi\subseteq\Upsilon$, $\Xi\neq\emptyset$, we get $\Xi\in Can(G)$.

Let $G\in OpRegDynExpr$ and $Can(G)\neq\emptyset$. Obviously, if there are only stochastic (immediate or waiting, respectively) multiactions in the multisets from $Can(G)$ then these stochastic (immediate or waiting, respectively) multiactions can be executed from G. Otherwise, besides stochastic ones, there are also deterministic (immediate and/or waiting) multiactions in the multisets from $Can(G)$. By the note above, there are non-empty multisets of deterministic multiactions in $Can(G)$ as well, i.e. $\exists\Upsilon\in Can(G)\ \Upsilon\in\mathbb{N}_{fin}^{\mathcal{DL}}\setminus\{\emptyset\}$. In this case, no stochastic multiactions can be executed from G, even if $Can(G)$ contains non-empty multisets of stochastic multiactions, since deterministic multiactions have a priority over stochastic ones, and should be executed first. Further, if there are no stochastic, but both waiting and immediate multiactions in the multisets from $Can(G)$, then, analogously, no waiting multiactions can be executed from G, since immediate multiactions have a priority over waiting ones (besides that over stochastic ones).

When there are only waiting and, possibly, stochastic multiactions in the multisets from $Can(G)$ then only waiting ones can be executed from G. Then just *maximal* non-empty multisets of waiting multiactions can be executed from G, since all non-conflicting waiting multiactions cannot wait and they should occur at the next time moment with probability 1.

Definition 11. *Let $G \in OpRegDynExpr$. The* set of all non-empty multisets of activities which can be executed from G *is*

$$Now(G)=\begin{cases} Can(G)\cap\mathbb{N}^{\mathcal{IL}}_{fin}, & Can(G)\cap\mathbb{N}^{\mathcal{IL}}_{fin}\neq\emptyset;\\ \{W\in Can(G)\cap\mathbb{N}^{\mathcal{WL}}_{fin}\mid & (Can(G)\cap\mathbb{N}^{\mathcal{IL}}_{fin}=\emptyset)\wedge\\ \forall V\in Can(G)\cap\mathbb{N}^{\mathcal{WL}}_{fin}\ W\subseteq V\Rightarrow V=W\}, & (Can(G)\cap\mathbb{N}^{\mathcal{WL}}_{fin}\neq\emptyset);\\ Can(G), & otherwise. \end{cases}$$

Let $G \in OpRegDynExpr$. The expression G is *s-tangible (stochastically tangible)*, denoted by $stang(G)$, if $Now(G) \subseteq \mathbb{N}^{\mathcal{SL}}_{fin} \setminus \{\emptyset\}$. In particular, we have $stang(G)$, if $Now(G) = \emptyset$. The expression G is *w-tangible (waitingly tangible)*, denoted by $wtang(G)$, if $\emptyset \neq Now(G) \subseteq \mathbb{N}^{\mathcal{WL}}_{fin}\setminus\{\emptyset\}$. The expression G is *tangible*, denoted by $tang(G)$, if $stang(G)$ or $wtang(G)$, i.e. $Now(G) \subseteq (\mathbb{N}^{\mathcal{SL}}_{fin} \cup \mathbb{N}^{\mathcal{WL}}_{fin}) \setminus \{\emptyset\}$. Again, we particularly have $tang(G)$, if $Now(G) = \emptyset$. Otherwise, the expression G is *vanishing*, denoted by $vanish(G)$, and in this case $\emptyset \neq Now(G) \subseteq \mathbb{N}^{\mathcal{IL}}_{fin} \setminus \{\emptyset\}$. Note that the operative dynamic expressions from $[G]_{\approx}$ may have different types in general.

Let $G \in RegDynExpr$. We write $stang([G]_{\approx})$, if $\forall H \in [G]_{\approx} \cap OpRegDynExpr\ stang(H)$. We write $wtang([G]_{\approx})$, if $\exists H \in [G]_{\approx} \cap OpRegDynExpr\ wtang(H)$ and $\forall H' \in [G]_{\approx} \cap OpRegDynExpr\ tang(H')$. We write $tang([G]_{\approx})$, if $stang([G]_{\approx})$ or $wtang([G]_{\approx})$. Otherwise, we write $vanish([G]_{\approx})$, and in this case $\exists H \in [G]_{\approx} \cap OpRegDynExpr\ vanish(H)$.

In Table 3, we define the action and empty move rules. In the table, $(\alpha,\rho),(\beta,\chi)\in\mathcal{SL}$, $(\alpha,\natural^0_l),(\beta,\natural^0_m)\in\mathcal{IL}$ and $(\alpha,\natural^\theta_l),(\beta,\natural^\theta_m)\in\mathcal{WL}$. Further, $E,F\in RegStatExpr$, $G,H\in SatOpRegDynExpr$, $\widetilde{G},\widetilde{H}\in RegDynExpr$ and $a\in Act$. Next, $\Gamma,\Delta\in\mathbb{N}^{\mathcal{SL}}_{fin}\setminus\{\emptyset\}$, $\Gamma'\in\mathbb{N}^{\mathcal{SL}}_{fin}$, $I,J\in\mathbb{N}^{\mathcal{IL}}_{fin}\setminus\{\emptyset\}$, $I'\in\mathbb{N}^{\mathcal{IL}}_{fin}$, $V,W\in\mathbb{N}^{\mathcal{WL}}_{fin}\setminus\{\emptyset\}$, $V'\in\mathbb{N}^{\mathcal{WL}}_{fin}$ and $\Upsilon\in\mathbb{N}^{\mathcal{SDL}}_{fin}\setminus\{\emptyset\}$. We denote $\Upsilon_a=\{(\alpha,\kappa)\in\Upsilon\mid(a\in\alpha)\vee(\hat{a}\in\alpha)\}$.

We use the following abbreviations in the names of the rules from the table: "**E**" for "**E**mpty move", "**B**" for "**B**asis case", "**S**" for "**S**equence", "**C**" for "**C**hoice", "**P**" for "**P**arallel", "**L**" for "re**L**abeling", "**R**" for "**R**estriction", "**I**" for "**I**teraton" and "**Sy**" for "**Sy**nchronization". The first rule in the table is the empty move rule **E**. The other rules are the action rules, describing transformations of dynamic expressions, which are built using particular algebraic operations. If we cannot merge the rules with stochastic, immediate ans waiting multiactions in one rule for some operation then we get the coupled action rules. In such cases, the names of the action rules with stochastic multiactions have a suffix '**s**', those with immediate multiactions have a suffix '**i**', and those with waiting multiactions have a suffix '**w**'. For explanation of the rules in Table 3, see [41].

TABLE 3. Action and empty move rules

$$\mathbf{E}\ \frac{stang([G]_{\approx})}{G \xrightarrow{\emptyset} \circlearrowleft G} \qquad \mathbf{Bs}\ \overline{(\alpha,\rho)}\ \xrightarrow{\{(\alpha,\rho)\}}\ \underline{(\alpha,\rho)} \qquad \mathbf{Bi}\ \overline{(\alpha,\natural_l^0)}\ \xrightarrow{\{(\alpha,\natural_l^0)\}}\ \underline{(\alpha,\natural_l^0)} \qquad \mathbf{Bw}\ \overline{(\alpha,\natural_l^\theta)^1}\ \xrightarrow{\{(\alpha,\natural_l^\theta)\}}\ \underline{(\alpha,\natural_l^\theta)}$$

$$\mathbf{S}\ \frac{G \xrightarrow{\Upsilon} \widetilde{G}}{G; E \xrightarrow{\Upsilon} \widetilde{G}; E,\ E; G \xrightarrow{\Upsilon} E; \widetilde{G}} \qquad \mathbf{Cs}\ \frac{G \xrightarrow{\Gamma} \widetilde{G},\ \neg init(G) \vee (init(G) \wedge stang([\overline{E}]_{\approx}))}{G[]E \xrightarrow{\Gamma} \widetilde{G}[]\downarrow E,\ E[]G \xrightarrow{\Gamma} \downarrow E[]\widetilde{G}}$$

$$\mathbf{Ci}\ \frac{G \xrightarrow{I} \widetilde{G}}{G[]E \xrightarrow{I} \widetilde{G}[]\downarrow E,\ E[]G \xrightarrow{I} \downarrow E[]\widetilde{G}} \qquad \mathbf{Cw}\ \frac{G \xrightarrow{V} \widetilde{G},\ \neg init(G) \vee (init(G) \wedge tang([\overline{E}]_{\approx}))}{G[]E \xrightarrow{V} \widetilde{G}[]\downarrow E,\ E[]G \xrightarrow{V} \downarrow E[]\widetilde{G}}$$

$$\mathbf{P1s}\ \frac{G \xrightarrow{\Gamma} \widetilde{G},\ stang([H]_{\approx})}{G\|H \xrightarrow{\Gamma} \widetilde{G}\|\circlearrowleft H,\ H\|G \xrightarrow{\Gamma} \circlearrowleft H\|\widetilde{G}} \qquad \mathbf{P1i}\ \frac{G \xrightarrow{I} \widetilde{G}}{G\|H \xrightarrow{I} \widetilde{G}\|H,\ H\|G \xrightarrow{I} H\|\widetilde{G}}$$

$$\mathbf{P1w}\ \frac{G \xrightarrow{V} \widetilde{G},\ stang([H]_{\approx})}{G\|H \xrightarrow{V} \widetilde{G}\|\circlearrowleft H,\ H\|G \xrightarrow{V} \circlearrowleft H\|\widetilde{G}} \qquad \mathbf{P2s}\ \frac{G \xrightarrow{\Gamma} \widetilde{G},\ H \xrightarrow{\Delta} \widetilde{H}}{G\|H \xrightarrow{\Gamma+\Delta} \widetilde{G}\|\widetilde{H}} \qquad \mathbf{P2i}\ \frac{G \xrightarrow{I} \widetilde{G},\ H \xrightarrow{J} \widetilde{H}}{G\|H \xrightarrow{I+J} \widetilde{G}\|\widetilde{H}}$$

$$\mathbf{P2w}\ \frac{G \xrightarrow{V} \widetilde{G},\ H \xrightarrow{W} \widetilde{H}}{G\|H \xrightarrow{V+W} \widetilde{G}\|\widetilde{H}} \qquad \mathbf{L}\ \frac{G \xrightarrow{\Upsilon} \widetilde{G}}{G[f] \xrightarrow{f(\Upsilon)} \widetilde{G}[f]} \qquad \mathbf{R}\ \frac{G \xrightarrow{\Upsilon} \widetilde{G}}{G\ \mathsf{rs}\ a \xrightarrow{\Upsilon-\Upsilon_a} \widetilde{G}\ \mathsf{rs}\ a}$$

$$\mathbf{I1}\ \frac{G \xrightarrow{\Upsilon} \widetilde{G}}{[G * E * F] \xrightarrow{\Upsilon} [\widetilde{G} * E * F]} \qquad \mathbf{I2s}\ \frac{G \xrightarrow{\Gamma} \widetilde{G},\ \neg init(G) \vee (init(G) \wedge stang([\overline{F}]_{\approx}))}{[E * G * F] \xrightarrow{\Gamma} [E * \widetilde{G} * \downarrow F],\ [E * F * G] \xrightarrow{\Gamma} [E * \downarrow F * \widetilde{G}]}$$

$$\mathbf{I2i}\ \frac{G \xrightarrow{I} \widetilde{G}}{[E * G * F] \xrightarrow{I} [E * \widetilde{G} * \downarrow F],\ [E * F * G] \xrightarrow{I} [E * \downarrow F * \widetilde{G}]}$$

$$\mathbf{I2w}\ \frac{G \xrightarrow{V} \widetilde{G},\ \neg init(G) \vee (init(G) \wedge tang([\overline{F}]_{\approx}))}{[E * G * F] \xrightarrow{V} [E * \widetilde{G} * \downarrow F],\ [E * F * G] \xrightarrow{V} [E * \downarrow F * \widetilde{G}]}$$

$$\mathbf{Sy1}\ \frac{G \xrightarrow{\Upsilon} \widetilde{G}}{G\ \mathsf{sy}\ a \xrightarrow{\Upsilon} \widetilde{G}\ \mathsf{sy}\ a} \qquad \mathbf{Sy2s}\ \frac{G\ \mathsf{sy}\ a \xrightarrow{\Gamma'+\{(\alpha,\rho)\}+\{(\beta,\chi)\}} \widetilde{G}\ \mathsf{sy}\ a,\ a \in \alpha,\ \hat{a} \in \beta}{G\ \mathsf{sy}\ a \xrightarrow{\Gamma'+\{(\alpha \oplus_a \beta, \rho \cdot \chi)\}} \widetilde{G}\ \mathsf{sy}\ a}$$

$$\mathbf{Sy2i}\ \frac{G\ \mathsf{sy}\ a \xrightarrow{I'+\{(\alpha,\natural_l^0)\}+\{(\beta,\natural_m^0)\}} \widetilde{G}\ \mathsf{sy}\ a,\ a \in \alpha,\ \hat{a} \in \beta}{G\ \mathsf{sy}\ a \xrightarrow{I'+\{(\alpha \oplus_a \beta, \natural_{l+m}^0)\}} \widetilde{G}\ \mathsf{sy}\ a}$$

$$\mathbf{Sy2w}\ \frac{G\ \mathsf{sy}\ a \xrightarrow{V'+\{(\alpha,\natural_l^\theta)\}+\{(\beta,\natural_m^\theta)\}} \widetilde{G}\ \mathsf{sy}\ a,\ a \in \alpha,\ \hat{a} \in \beta}{G\ \mathsf{sy}\ a \xrightarrow{V'+\{(\alpha \oplus_a \beta, \natural_{l+m}^\theta)\}} \widetilde{G}\ \mathsf{sy}\ a}$$

Notice that the timers of all waiting multiactions that lose their enabledness when a state change occurs become inactive (turned off) and their values become irrelevant while the timers of all those preserving their enabledness continue running with their stored values. Hence, we adapt the *enabling memory* policy [29, 1, 2, 3] when the process states are changed and the enabledness of deterministic multiactions is possibly modified (immediate multiactions may be seen as those with the timers displaying a single value 0, so we do not need to store their values). Then the timer values of waiting multiactions are taken as the enabling memory variables.

Like in [18], we are interested in the dynamic expressions, inferred by applying the inaction rules (also in the reverse direction) and action rules from the overlined static expressions, such that no stamped (superscribed

with the timer values) waiting multiaction is a subexpression of them. The reason is to ensure that time proceeds uniformly and only enabled waiting multiactions are stamped. We call such dynamic expressions reachable, by analogy with the reachable states of LDTSDPNs [41].

Definition 12. *A dynamic expression G is* reachable*, if there exists a static expression E without timer value superscripts, such that $\overline{E} \approx G$ or $\overline{E} \approx G_0 \stackrel{\Upsilon_1}{\to} H_1 \approx G_1 \stackrel{\Upsilon_2}{\to} \ldots \stackrel{\Upsilon_n}{\to} H_n \approx G$ for some $\Upsilon_1, \ldots, \Upsilon_n \in \mathbb{N}^{\mathcal{SDL}}_{fin}$.*

We now consider the enabledness of the stamped waiting multiactions.

Proposition 2. *Let G be a reachable dynamic expression. Then only waiting multiactions from $EnaWait([G]_\approx)$ are stamped in G.*

Proof. See [41]. □

3.3. Transition systems. We now construct labeled probabilistic transition systems associated with dynamic expressions. The transition systems are used to define the operational semantics of dynamic expressions.

Let G be a dynamic expression and $s = [G]_\approx$. The set of *all multisets of activities executable in* s is defined as $Exec(s) = \{\Upsilon \mid \exists H \in s\ \exists \widetilde{H}\ H \stackrel{\Upsilon}{\to} \widetilde{H}\}$. Here $H \stackrel{\Upsilon}{\to} \widetilde{H}$ is an inference by the rules from Table 3. It can be proved by induction on the structure of expressions that $\Upsilon \in Exec(s) \setminus \{\emptyset\}$ implies $\exists H \in s\ \Upsilon \in Now(H)$. The reverse statement does not hold, since the preconditions in the action rules disable executions of the activities with the lower-priority types from every $H \in s$, see [41].

The state s is *s-tangible (stochastically tangible)*, denoted by $stang(s)$, if $Exec(s) \subseteq \mathbb{N}^{\mathcal{SL}}_{fin}$. For an s-tangible state s we always have $\emptyset \in Exec(s)$ by rule **E**, hence, we may have $Exec(s) = \{\emptyset\}$. The state s is *w-tangible (waitingly tangible)*, denoted by $wtang(s)$, if $Exec(s) \subseteq \mathbb{N}^{\mathcal{WL}}_{fin} \setminus \{\emptyset\}$. The state s is *tangible*, denoted by $tang(s)$, if $stang(s)$ or $wtang(s)$, i.e. $Exec(s) \subseteq \mathbb{N}^{\mathcal{SL}}_{fin} \cup \mathbb{N}^{\mathcal{WL}}_{fin}$. Again, for a tangible state s we may have $\emptyset \in Exec(s)$ and $Exec(s) = \{\emptyset\}$. Otherwise, the state s is *vanishing*, denoted by $vanish(s)$, and in this case $Exec(s) \subseteq \mathbb{N}^{\mathcal{IL}}_{fin} \setminus \{\emptyset\}$.

If $\Upsilon \in Exec(s)$ and $\Upsilon \in \mathbb{N}^{\mathcal{SL}}_{fin} \cup \mathbb{N}^{\mathcal{IL}}_{fin}$ then by rules **P2s**, **P2i**, **Sy2s**, **Sy2i** and definition of $Exec(s)$ $\forall \Xi \subseteq \Upsilon$, $\Xi \neq \emptyset$, we have $\Xi \in Exec(s)$, i.e. $2^\Upsilon \setminus \{\emptyset\} \subseteq Exec(s)$.

Definition 13. *The* derivation set *of a dynamic expression G, denoted by $DR(G)$, is the minimal set such that*

- $[G]_\approx \in DR(G)$*;*
- *if $[H]_\approx \in DR(G)$ and $\exists \Upsilon\ H \stackrel{\Upsilon}{\to} \widetilde{H}$ then $[\widetilde{H}]_\approx \in DR(G)$.*

The set of *all s-tangible states from* $DR(G)$ is denoted by $DR_{ST}(G)$, and the set of *all w-tangible states from* $DR(G)$ is denoted by $DR_{WT}(G)$. The set of *all tangible states from* $DR(G)$ is denoted by $DR_T(G) = DR_{ST}(G) \cup$

$DR_{WT}(G)$. The set of *all vanishing states from* $DR(G)$ is denoted by $DR_V(G)$. Then $DR(G) = DR_T(G) \cup DR_V(G) = DR_{ST}(G) \cup DR_{WT}(G) \cup DR_V(G)$.

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

Let $\Upsilon \in Exec(s) \setminus \{\emptyset\}$. The *probability that the multiset of stochastic multiactions* Υ *is ready for execution in* s or the *weight of the multiset of deterministic multiactions* Υ *which is ready for execution in* s is

$$PF(\Upsilon, s) = \begin{cases} \displaystyle\prod_{(\alpha,\rho)\in\Upsilon} \rho \cdot \prod_{\{\{(\beta,\chi)\}\in Exec(s) \mid (\beta,\chi)\notin\Upsilon\}} (1-\chi), & s \in DR_{ST}(G); \\ \displaystyle\sum_{(\alpha,\natural_l^\theta)\in\Upsilon} l, & s \in DR_{WT}(G) \cup DR_V(G). \end{cases}$$

In the case $\Upsilon = \emptyset$ and $s \in DR_{ST}(G)$ we define

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

Let $\Upsilon \in Exec(s)$. Besides Υ, other multisets of activities may be ready for execution in s, hence, a normalization is needed to calculate the execution probability. The *probability to execute the multiset of activities* Υ *in* s is

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

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

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

Definition 14. *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* $L_G = \mathbb{N}_{fin}^{\mathcal{SDL}} \times (0; 1]$*;*
- *the set of* transitions *is* $\mathcal{T}_G = \{(s, (\Upsilon, PT(\Upsilon, s)), \tilde{s}) \mid s, \tilde{s} \in DR(G), \exists H \in s\ \exists \widetilde{H} \in \tilde{s}\ H \xrightarrow{\Upsilon} \widetilde{H}\}$*;*
- *the* initial state *is* $s_G = [G]_\approx$.

The transition system $TS(G)$ associated with a dynamic expression G describes all the steps (parallel executions) that occur at discrete time moments with some (one-step) probability and consist of multisets of activities. Every step consisting of stochastic (waiting, respectively) multiactions or the empty step (i.e. that consisting of the empty multiset of activities) occurs instantly after one discrete time unit delay. Each step consisting of immediate multiactions occurs instantly without any delay. The step can change the current state to a different 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 $(s, (\Upsilon, \mathcal{P}), \tilde{s}) \in \mathcal{T}_G$ will be

written as $s \xrightarrow{\Upsilon}_{\mathcal{P}} \tilde{s}$. It is interpreted as follows: the probability to change from state s to $\tilde{s}$ as a result of executing Υ is $\mathcal{P}$.

From every s-tangible state the empty multiset of activities can always be executed by rule **E**. Hence, for s-tangible states, Υ may be the empty multiset, and its execution only decrements by one the timer values (if any) of the current state. Then we have a transition $s \xrightarrow{\emptyset}_{\mathcal{P}} \circlearrowleft s$ from an s-tangible state s to the tangible state $\circlearrowleft s = [\circlearrowleft H]_{\approx}$ for $H \in s \cap SatOpRegDynExpr$. Since structurally equivalent saturated operative dynamic expressions remain so after decreasing by one their timers, $\circlearrowleft s$ is unique for each s and the definition is correct. Thus, $\circlearrowleft s$ corresponds to applying the empty move rule to an arbitrary saturated operative dynamic expression from s, followed by taking the structural equivalence class of the result. We have to keep track of such executions, called the *empty moves*, since they affect the timers and have non-zero probabilities. This follows from the definition of $PF(\emptyset, s)$ and the fact that the probabilities of stochastic multiactions belong to the interval $(0; 1)$. When it holds $\circlearrowleft H = H$ for $H \in s \cap SatOpRegDynExpr$, we obtain $\circlearrowleft s = s$. Then the empty move from s is in the form of $s \xrightarrow{\emptyset}_{\mathcal{P}} s$, called the *empty loop*. For w-tangible and vanishing states Υ cannot be the empty multiset, since we must execute some immediate (waiting) multiactions from them at the current (next) moment.

The step probabilities belong to the interval $(0; 1]$, being 1 when the only transition from an s-tangible state s is the empty move one $s \xrightarrow{\emptyset}_1 \circlearrowleft s$, or if there is a single transition from a w-tangible or a vanishing state. We write $s \xrightarrow{\Upsilon} \tilde{s}$ if $\exists \mathcal{P}\ s \xrightarrow{\Upsilon}_{\mathcal{P}} \tilde{s}$ and $s \to \tilde{s}$ if $\exists \Upsilon\ s \xrightarrow{\Upsilon} \tilde{s}$.
Isomorphism is a coincidence of systems up to renaming of their components.

Definition 15. *Let for dynamic expressions G, G', $TS(G) = (S_G, L_G, \mathcal{T}_G, s_G)$, $TS(G') = (S_{G'}, L_{G'}, \mathcal{T}_{G'}, s_{G'})$. 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*

(1) *β is a bijection such that $\beta(s_G) = s_{G'}$;*
(2) *$\forall s, \tilde{s} \in S_G\ \forall \Upsilon\ s \xrightarrow{\Upsilon}_{\mathcal{P}} \tilde{s} \ \Leftrightarrow\ \beta(s) \xrightarrow{\Upsilon}_{\mathcal{P}} \beta(\tilde{s})$.*

Two transition systems $TS(G)$ and $TS(G')$ are isomorphic*, denoted by $TS(G) \simeq TS(G')$, if $\exists \beta : TS(G) \simeq TS(G')$.*

Definition 16. *Two dynamic expressions G and G' are* equivalent with respect to transition systems*, denoted by $G =_{ts} G'$, if $TS(G) \simeq TS(G')$.*

Let $N = (P_N, T_N, W_N, D_N, \Omega_N, \mathcal{L}_N, Q_N)$ be a LDTSDPN [41] and $Q, \widetilde{Q}$ be its states. Then the average sojourn time $SJ(Q)$, the sojourn time variance $VAR(Q)$, the probabilities $PM^*(Q, \widetilde{Q})$, the transition relation $Q \twoheadrightarrow_{\mathcal{P}} \widetilde{Q}$, the EDTMC $EDTMC(N)$, the underlying SMC $SMC(N)$ and the steady-state PMF for it are defined like the corresponding notions for dynamic expressions in [42]. Every marked and clocked plain dtsd-box [41] can be interpreted as an LDTSDPN. Therefore, we can evaluate performance with the LDTSDPNs corresponding to dtsd-boxes and then transfer the results to the latter.

4 Stochastic equivalences

The semantic equivalence $=_{ts}$ is too discriminating in many cases, hence, we need weaker equivalence notions. These equivalences should possess the following necessary properties. First, any two equivalent processes must have the same sequences of multisets of multiactions, which are the multiaction parts of the activities executed in steps starting from the initial states of the processes. Second, for every such sequence, its execution probabilities within both processes must coincide. Third, the desired equivalence should preserve the branching structure of computations, i.e. the points of choice of an external observer between several extensions of a particular computation should be taken into account. In this section, we define one such notion: 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 $[G]_{\approx}$ and $[G']_{\approx}$ of their transition systems should be related by a bisimulation having the following transfer property: if two states are related then in each of them the same multisets of multiactions can occur, leading with the identical overall probability from each of the two states to *the same equivalence class* for every such multiset.

We follow the approaches of [17, 20, 14, 15, 7, 5, 6], but we implement step semantics instead of interleaving one considered in these papers. Recall also that we use the generative probabilistic transition systems, like in [17], in contrast to the reactive model, treated in [20], and we take transition probabilities instead of transition rates from [14, 15, 7, 5, 6]. Thus, step stochastic bisimulation equivalence that we define further is (in the probabilistic sense) comparable only with interleaving probabilistic bisimulation equivalence from [17], and our relation is obviously stronger.

In the definition below, we consider $\mathcal{L}(\Upsilon) \in \mathbb{N}^{\mathcal{L}}_{fin}$ for $\Upsilon \in \mathbb{N}^{\mathcal{SIL}}_{fin}$, i.e. (possibly empty) multisets of multiactions. The multiactions can be empty as well. In this case, $\mathcal{L}(\Upsilon)$ contains the elements $\emptyset$, but 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 $A \in \mathbb{N}^{\mathcal{L}}_{fin}$, we write $s \xrightarrow{A}_{\mathcal{P}} \mathcal{H}$, 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

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

We write $s \xrightarrow{A} \mathcal{H}$ if $\exists \mathcal{P}\ s \xrightarrow{A}_{\mathcal{P}} \mathcal{H}$. Further, we write $s \rightarrow_{\mathcal{P}} \mathcal{H}$ if $\exists A\ s \xrightarrow{A} \mathcal{H}$, 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

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

For $\tilde{s} \in DR(G)$, we write $s \xrightarrow{A}_{\mathcal{P}} \tilde{s}$ if $s \xrightarrow{A}_{\mathcal{P}} \{\tilde{s}\}$ and $s \xrightarrow{A} \tilde{s}$ if $\exists \mathcal{P}\ s \xrightarrow{A}_{\mathcal{P}} \tilde{s}$.

The mean value formula for the geometrical distribution allows us to calculate the average sojourn time in $s \in DR_T(G)$ as $SJ(s) = \frac{1}{1-PM(s,s)}$. The average sojourn time in each vanishing state $s \in DR_V(G)$ is $SJ(s) = 0$. Let $s \in DR(G)$.

The *average sojourn time in the state* s is

$$SJ(s) = \begin{cases} \frac{1}{1-PM(s,s)}, & s \in DR_T(G); \\ 0, & s \in DR_V(G). \end{cases}$$

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

$$VAR(s) = \begin{cases} \frac{PM(s,s)}{(1-PM(s,s))^2}, & s \in DR_T(G); \\ 0, & s \in DR_V(G). \end{cases}$$

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

Definition 17. *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) $(s_1, s_2) \in \mathcal{R}$ *implies* $SJ(s_1) = 0 \ \Leftrightarrow\ SJ(s_2) = 0$ *and*

$$\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}\ \forall A \in \mathbb{N}^{\mathcal{L}}_{fin}\ s_1 \xrightarrow{A}_{\mathcal{P}} \mathcal{H} \ \Leftrightarrow\ s_2 \xrightarrow{A}_{\mathcal{P}} \mathcal{H}.$$

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

The condition $SJ(s_1) = 0 \ \Leftrightarrow\ SJ(s_2) = 0$ in item 2 of the definition above is needed to make difference between w-tangible states (all having at least one time unit sojourn times) and vanishing states (all having zero sojourn times). Both from w-tangible and vanishing states, no empty moves can be made, unlike s-tangible states, from which empty moves are always possible. When comparing dynamic expressions for step stochastic bisimulation equivalence, we can use empty moves only to make difference between s-tangible and other (w-tangible or vanishing) states.

We now define the multiaction transition systems, whose transitions are labeled with the multisets of multiactions from the corresponding activities.

Definition 18. *Let* G *be a dynamic expression. The* (labeled probabilistic) multiaction transition system *of* G *is a quadruple* $TS_{\mathcal{L}}(G) = (S_{\mathcal{L}}, L_{\mathcal{L}}, \mathcal{T}_{\mathcal{L}}, s_{\mathcal{L}})$*, where*

- $S_{\mathcal{L}} = DR(G)$;
- $L_{\mathcal{L}} = \mathbb{N}^{\mathcal{L}}_{fin} \times (0;1]$;
- $\mathcal{T}_{\mathcal{L}} = \{(s,(A,PM_A(s,\{\tilde{s}\})),\tilde{s}) \mid s,\tilde{s} \in DR(G),\ s \stackrel{A}{\rightarrow} \tilde{s}\}$;
- $s_{\mathcal{L}} = [G]_{\approx}$.

The transition $(s,(A,\mathcal{P}),\tilde{s}) \in \mathcal{T}_{\mathcal{L}}$ will be written as $s \stackrel{A}{\rightarrow}_{\mathcal{P}} \tilde{s}$.

Let G and G' be dynamic expressions and $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$. Then the relation $\mathcal{R}$ can be interpreted as a step stochastic bisimulation between the transition systems $TS_{\mathcal{L}}(G)$ and $TS_{\mathcal{L}}(G')$, denoted by $\mathcal{R} : TS_{\mathcal{L}}(G) \underline{\leftrightarrow}_{ss} TS_{\mathcal{L}}(G')$, which is defined by analogy (excepting step semantics) with interleaving probabilistic bisimulation on generative probabilistic transition systems from [17].

The following proposition states that every step stochastic bisimulation binds s-tangible states only with s-tangible ones, and the same is valid for w-tangible states, as well as for vanishing states.

Proposition 3. *Let G and G' be dynamic expressions and $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$. Then $\mathcal{R} \subseteq (DR_{ST}(G) \cup DR_{ST}(G'))^2 \uplus (DR_{WT}(G) \cup DR_{WT}(G'))^2 \uplus (DR_V(G) \cup DR_V(G'))^2$.*

Proof. See [41]. □

Proposition 3 implies $\mathcal{R} \subseteq (DR_T(G) \cup DR_T(G'))^2 \uplus (DR_V(G) \cup DR_V(G'))^2$, since $DR_T(G) = DR_{ST}(G) \uplus DR_{WT}(G)$ and $DR_T(G') = DR_{ST}(G') \uplus DR_{WT}(G')$.

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 4. *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 [41]. □

The next theorem shows that both the semantics are bisimulation equivalent.

Theorem 1. *For any static expression E, $TS(\overline{E}) \underline{\leftrightarrow}_{ss} RG(Box_{dtsd}(\overline{E}))$.*

Proof. See [41]. □

We now compare the discrimination power of the stochastic equivalences.

Theorem 2. *For dynamic expressions G and G' the* strict *implications hold:*

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

Proof. See [41]. □

5 Reduction modulo equivalences

The equivalences which we proposed can be used to reduce transition systems and SMCs of expressions (reachability graphs and SMCs of dtsd-boxes). Reductions of graph-based models, like transition systems, reachability graphs and SMCs, 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 behavioural properties. Thus, the reduction allows one to simplify the functional and performance analysis.

We now define the quotient transition systems and Markov chains (SMCs, DTMCs, RDTMCs).

An *autobisimulation* is a bisimulation between an expression and itself. For a dynamic expression G and a step stochastic autobisimulation $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G$, let $\mathcal{K} \in DR(G)/_{\mathcal{R}}$ and $s_1, s_2 \in \mathcal{K}$. We have $\forall \widetilde{\mathcal{K}} \in DR(G)/_{\mathcal{R}}\ \forall A \in \mathbb{N}^{\mathcal{L}}_{fin}\ s_1 \stackrel{A}{\rightarrow}_{\mathcal{P}} \widetilde{\mathcal{K}} \ \Leftrightarrow\ s_2 \stackrel{A}{\rightarrow}_{\mathcal{P}} \widetilde{\mathcal{K}}$. The equality 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 $\exists A\ \mathcal{K} \stackrel{A}{\rightarrow} \widetilde{\mathcal{K}}$. Similar arguments allow us to write $\mathcal{K} \rightarrow_{\mathcal{P}} \widetilde{\mathcal{K}}$, where $\mathcal{P} = PM(\mathcal{K}, \widetilde{\mathcal{K}}) = PM(s_1, \widetilde{\mathcal{K}}) = PM(s_2, \widetilde{\mathcal{K}})$.

By the note after Proposition 3, $\mathcal{R} \subseteq (DR_T(G))^2 \uplus (DR_V(G))^2$. Hence, $\forall \mathcal{K} \in DR(G)/_{\mathcal{R}}$, all states from $\mathcal{K}$ are tangible, when $\mathcal{K} \in DR_T(G)/_{\mathcal{R}}$, or all of them are vanishing, when $\mathcal{K} \in DR_V(G)/_{\mathcal{R}}$.

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

$$SJ_{\mathcal{R}}(\mathcal{K}) = \begin{cases} \frac{1}{1-PM(\mathcal{K},\mathcal{K})}, & \mathcal{K} \in DR_T(G)/_{\mathcal{R}}; \\ 0, & \mathcal{K} \in DR_V(G)/_{\mathcal{R}}. \end{cases}$$

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

$$VAR_{\mathcal{R}}(\mathcal{K}) = \begin{cases} \frac{PM(\mathcal{K},\mathcal{K})}{(1-PM(\mathcal{K},\mathcal{K}))^2}, & \mathcal{K} \in DR_T(G)/_{\mathcal{R}}; \\ 0, & \mathcal{K} \in DR_V(G)/_{\mathcal{R}}. \end{cases}$$

The *sojourn time variance vector for the equivalence classes (with respect to $\mathcal{R}$) 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 4, $\mathcal{R}_{ss}(G)$ is the largest step stochastic autobisimulation on G. Based on the equivalence classes with respect to $\mathcal{R}_{ss}(G)$, the quotient (by $\underline{\leftrightarrow}_{ss}$) transition systems and the quotient (by $\underline{\leftrightarrow}_{ss}$) underlying SMCs of expressions can be defined. The mentioned equivalence classes become the quotient states. The average sojourn time in a quotient state is that in the corresponding equivalence class. 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 19. *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*

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

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

Let G be a dynamic expression. We define the relation $\mathcal{R}_{\mathcal{L}ss}(G) = \{(s, \mathcal{K}), (\mathcal{K}, s) \mid s \in \mathcal{K} \in DR(G)/_{\mathcal{R}_{ss}(G)}\}^+$, where $^+$ is the transitive closure operation. One can see that $\mathcal{R}_{\mathcal{L}ss}(G) \subseteq (DR(G) \cup DR(G)/_{\mathcal{R}_{ss}(G)})^2$ is an equivalence relation that partitions the set $DR(G) \cup DR(G)/_{\mathcal{R}_{ss}(G)}$ to the equivalence classes $\mathcal{L}_1, \ldots, \mathcal{L}_n$, defined as $\mathcal{L}_i = \mathcal{K}_i \cup \{\mathcal{K}_i\}$ $(1 \leq i \leq n)$, where $DR(G)/_{\mathcal{R}_{ss}(G)} = \{\mathcal{K}_1, \ldots, \mathcal{K}_n\}$. The relation $\mathcal{R}_{\mathcal{L}ss}(G)$ can be interpreted as a step stochastic bisimulation between the transition systems $TS_{\mathcal{L}}(G)$ and $TS_{\underline{\leftrightarrow}_{ss}}(G)$, denoted by $\mathcal{R}_{\mathcal{L}ss}(G) : TS_{\mathcal{L}}(G) \underline{\leftrightarrow}_{ss} TS_{\underline{\leftrightarrow}_{ss}}(G)$, which is defined by analogy (excepting step semantics) with interleaving probabilistic bisimulation on generative probabilistic transition systems from [17]. From this viewpoint, $\mathcal{R}_{\mathcal{L}ss}(G)$ is also the union of all step stochastic bisimulations and largest step stochastic bisimulation between $TS_{\mathcal{L}}(G)$ and $TS_{\underline{\leftrightarrow}_{ss}}(G)$.

The quotient (by $\underline{\leftrightarrow}_{ss}$) reachability graphs are defined similarly to the quotient transition systems. Let $\simeq$ denote isomorphism between quotient transition systems and quotient reachability graphs that binds their initial states. The following proposition establishes a connection between quotient (by $\underline{\leftrightarrow}_{ss}$) transition systems of the overlined static expressions and quotient reachability graphs of their dtsd-boxes.

Proposition 5. *For any static expression E,*

$$TS_{\underline{\leftrightarrow}_{ss}}(\overline{E}) \simeq RG_{\underline{\leftrightarrow}_{ss}}(Box_{dtsd}(\overline{E})).$$

Proof. See [43]. □

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

Let G be a dynamic expression and $\mathcal{K}, \widetilde{\mathcal{K}} \in DR(G)/_{\mathcal{R}_{ss}(G)}$. The transition system $TS_{\underline{\leftrightarrow}_{ss}}(G)$ can have self-loops from an equivalence class to itself with a non-zero probability. Then the current equivalence class remains unchanged.

Let $\mathcal{K} \rightarrow \mathcal{K}$. The *probability to stay in $\mathcal{K}$ due to k $(k \geq 1)$ self-loops* is

$$PM(\mathcal{K}, \mathcal{K})^k.$$

The *quotient (by $\underline{\leftrightarrow}_{ss}$) self-loops abstraction factor in the equivalence class* $\mathcal{K}$ is

$$SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) = \begin{cases} \frac{1}{1-PM(\mathcal{K},\mathcal{K})}, & \mathcal{K} \to \mathcal{K}; \\ 1, & \text{otherwise.} \end{cases}$$

The *quotient (by $\underline{\leftrightarrow}_{ss}$) self-loops abstraction vector* of G, denoted by $SL_{\underline{\leftrightarrow}_{ss}}$, has the elements $SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}),\ \mathcal{K} \in DR(G)/_{\mathcal{R}_{ss}(G)}$.

Let $\mathcal{K} \to \widetilde{\mathcal{K}}$ and $\mathcal{K} \neq \widetilde{\mathcal{K}}$, i.e. $PM(\mathcal{K},\mathcal{K}) < 1$. The *probability to move from $\mathcal{K}$ to $\widetilde{\mathcal{K}}$ by executing any multiset of activities after possible self-loops* is

$$PM^*(\mathcal{K},\widetilde{\mathcal{K}}) = \left\{ \begin{array}{ll} PM(\mathcal{K},\widetilde{\mathcal{K}}) \sum_{k=0}^{\infty} PM(\mathcal{K},\mathcal{K})^k = \frac{PM(\mathcal{K},\widetilde{\mathcal{K}})}{1-PM(\mathcal{K},\mathcal{K})}, & \mathcal{K} \to \mathcal{K}; \\ PM(\mathcal{K},\widetilde{\mathcal{K}}), & \text{otherwise;} \end{array} \right\} =$$
$$SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) PM(\mathcal{K},\widetilde{\mathcal{K}}).$$

The value $k = 0$ in the summation is the case when no self-loops occur.

Let $\mathcal{K} \in DR_T(G)/_{\mathcal{R}_{ss}(G)}$. If there exist self-loops from $\mathcal{K}$ (i.e. if $\mathcal{K} \to \mathcal{K}$) then $PM(\mathcal{K},\mathcal{K}) > 0$ and $SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) = \frac{1}{1-PM(\mathcal{K},\mathcal{K})} = SJ_{\underline{\leftrightarrow}_{ss}}(\mathcal{K})$. Otherwise, if there exist no self-loops from $\mathcal{K}$ then $PM(\mathcal{K},\mathcal{K}) = 0$ and $SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) = 1 = \frac{1}{1-PM(\mathcal{K},\mathcal{K})} = SJ_{\underline{\leftrightarrow}_{ss}}(\mathcal{K})$. Thus, $\forall \mathcal{K} \in DR_T(G)/_{\mathcal{R}_{ss}(G)}\ SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) = SJ_{\underline{\leftrightarrow}_{ss}}(\mathcal{K})$, hence, $\forall \mathcal{K} \in DR_T(G)/_{\mathcal{R}_{ss}(G)}$ with $PM(\mathcal{K},\mathcal{K}) < 1$ it holds $PM^*(\mathcal{K},\widetilde{\mathcal{K}}) = SJ_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) PM(\mathcal{K},\widetilde{\mathcal{K}})$. Note that the self-loops from the equivalence classes of tangible states are of the empty or non-empty type, the latter produced by iteration, since empty loops are not possible from the equivalence classes of w-tangible states, but they are possible from the equivalence classes of s-tangible states, while non-empty loops are possible from the equivalence classes of both s-tangible and w-tangible states.

Let $\mathcal{K} \in DR_V(G)/_{\mathcal{R}_{ss}(G)}$. We have $\forall \mathcal{K} \in DR_V(G)/_{\mathcal{R}_{ss}(G)}\ SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) \neq SJ_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) = 0$ and $\forall \mathcal{K} \in DR_V(G)/_{\mathcal{R}_{ss}(G)}$ with $PM(\mathcal{K},\mathcal{K}) < 1$ it holds $PM^*(\mathcal{K},\widetilde{\mathcal{K}}) = SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) PM(\mathcal{K},\widetilde{\mathcal{K}})$. If there exist self-loops from $\mathcal{K}$ then $PM^*(\mathcal{K},\widetilde{\mathcal{K}}) = \frac{PM(\mathcal{K},\widetilde{\mathcal{K}})}{1-PM(\mathcal{K},\mathcal{K})}$ when $PM(\mathcal{K},\mathcal{K}) < 1$. Otherwise, if there exist no self-loops from $\mathcal{K}$ then $PM^*(\mathcal{K},\widetilde{\mathcal{K}}) = PM(\mathcal{K},\widetilde{\mathcal{K}})$. Note that the self-loops from the equivalence classes of vanishing states are always of the non-empty type, produced by iteration, since empty loops are not possible from the equivalence classes of vanishing states.

Definition 20. *Let G be a dynamic expression. The* quotient (by $\underline{\leftrightarrow}_{ss}$) EDTMC *of G, denoted by $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$, has the state space $DR(G)/_{\mathcal{R}_{ss}(G)}$, the initial state $[[G]_{\approx}]_{\mathcal{R}_{ss}(G)}$ and the transitions $\mathcal{K} \twoheadrightarrow_{\mathcal{P}} \widetilde{\mathcal{K}}$, if $\mathcal{K} \to \widetilde{\mathcal{K}}$ and $\mathcal{K} \neq \widetilde{\mathcal{K}}$, where $\mathcal{P} = PM^*(\mathcal{K},\widetilde{\mathcal{K}})$; or $\mathcal{K} \twoheadrightarrow_1 \mathcal{K}$, if $PM(\mathcal{K},\mathcal{K}) = 1$.*

The quotient (by $\underline{\leftrightarrow}_{ss}$) underlying SMC *of G, denoted by $SMC_{\underline{\leftrightarrow}_{ss}}(G)$, has the EDTMC $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$ and the sojourn time in every $\mathcal{K} \in DR_T(G)/_{\mathcal{R}_{ss}(G)}$ is geometrically distributed with the parameter $1-PM(\mathcal{K},\mathcal{K})$ while the sojourn time in every $\mathcal{K} \in DR_V(G)/_{\mathcal{R}_{ss}(G)}$ is equal to zero.*

The steady-state PMFs $\psi^*_{\underline{\leftrightarrow}_{ss}}$ for $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$ and $\varphi_{\underline{\leftrightarrow}_{ss}}$ for $SMC_{\underline{\leftrightarrow}_{ss}}(G)$ are defined like ψ^* for $EDTMC(G)$ and φ for $SMC(G)$ [42], respectively.

Let $\simeq$ denote isomorphism between SMCs that binds their initial states, where two SMCs are isomorphic if their EDTMCs are so and the sojourn times in the isomorphic states of the EDTMCs are identically distributed. The next proposition establishes a connection between quotient (by $\underline{\leftrightarrow}_{ss}$) SMCs of the overlined static expressions and quotient SMCs of their dtsd-boxes.

Proposition 6. *For any static expression* E

$$SMC_{\underline{\leftrightarrow}_{ss}}(\overline{E}) \simeq SMC_{\underline{\leftrightarrow}_{ss}}(Box_{dtsd}(\overline{E})).$$

Proof. See [43]. □

The quotients of both transition systems and underlying SMCs are the minimal reductions of the mentioned objects modulo step stochastic bisimulations. The quotients can be used to simplify analysis of system properties which are preserved by $\underline{\leftrightarrow}_{ss}$, since potentially less states should be examined for it.

Let us consider quotient (by $\underline{\leftrightarrow}_{ss}$) DTMCs of expressions based on the state change probabilities $PM(\mathcal{K}, \widetilde{\mathcal{K}})$.

Definition 21. *Let* G *be a dynamic expression. The* quotient (by $\underline{\leftrightarrow}_{ss}$) DTMC *of* G*, denoted by* $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$*, has the state space* $DR(G)/_{\mathcal{R}_{ss}(G)}$*, the initial state* $[[G]_{\approx}]_{\mathcal{R}_{ss}(G)}$ *and the transitions* $\mathcal{K} \rightarrow_{\mathcal{P}} \widetilde{\mathcal{K}}$*, where* $\mathcal{P} = PM(\mathcal{K}, \widetilde{\mathcal{K}})$.

The steady-state PMF $\psi_{\underline{\leftrightarrow}_{ss}}$ for $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ is defined like the corresponding notion ψ for $DTMC(G)$ [42].

Eliminating equivalence classes (with respect to $\mathcal{R}_{ss}(G)$) of vanishing states from the quotient (by $\underline{\leftrightarrow}_{ss}$) DTMCs of results in their reductions.

Definition 22. *The* reduced quotient (by $\underline{\leftrightarrow}_{ss}$) DTMC *of* G*, denoted by* $RDTMC_{\underline{\leftrightarrow}_{ss}}(G)$*, is defined like* $RDTMC(G)$ *in* [42]*, but it is constructed from* $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ *instead of* $DTMC(G)$.

The steady-state PMF $\psi^{\diamond}_{\underline{\leftrightarrow}_{ss}}$ for $RDTMC_{\underline{\leftrightarrow}_{ss}}(G)$ is defined like the corresponding notion $\psi^{\diamond}$ for $RDTMC(G)$ [42].

The relationships between the steady-state PMFs $\psi_{\underline{\leftrightarrow}_{ss}}$ and $\psi^*_{\underline{\leftrightarrow}_{ss}}$, $\varphi_{\underline{\leftrightarrow}_{ss}}$ and $\psi_{\underline{\leftrightarrow}_{ss}}$, as well as $\varphi_{\underline{\leftrightarrow}_{ss}}$ and $\psi^{\diamond}_{\underline{\leftrightarrow}_{ss}}$, are the same as those between their "non-quotient" versions in Proposition 3, Proposition 4 and Proposition 5 [42].

6 Stationary behaviour

Let us examine how the proposed equivalences can be used to compare the behaviour of stochastic processes in their steady states. We shall consider only formulas specifying stochastic processes with infinite behavior, i.e. expressions with the iteration operator. Note that the iteration operator does not guarantee infiniteness of behaviour, since there can exist a deadlock (blocking) within the body (the second argument) of iteration when the

corresponding subprocess does not reach its final state by some reasons. In particular, consider the expression $\mathsf{Stop} = (\{g\}, \frac{1}{2})\ \mathsf{rs}\ g$ specifying the non-terminating process that performs only empty loops with probability 1. If the body of iteration contains the Stop expression then the iteration will be "broken". On the other hand, the iteration body can be left after a finite number of its repeated executions and then the iteration termination is started. To avoid executing any activities after the iteration body, we take Stop as the termination argument of iteration.

We consider only the process expressions such that their underlying SMCs contain exactly one closed communication class of states, and this class should be ergodic to ensure uniqueness of the stationary distribution, which is also the limiting one. The states not belonging to that class do not disturb the uniqueness, since the closed communication class is single, hence, they all are transient. Then, for each transient state, the steady-state probability to be in it is zero while the steady-state probability to enter into the ergodic class starting from that state is equal to one.

6.1. Steady state, residence time and equivalences. The next proposition shows that, for two dynamic expressions related by $\underline{\leftrightarrow}_{ss}$, the steady-state probabilities to enter into an equivalence class coincide. Hence, the mean recurrence time for an equivalence class is the same for both expressions.

Proposition 7. *Let G, G' be dynamic expressions with $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$ and φ be the steady-state PMF for $SMC(G)$, φ' be the steady-state PMF for $SMC(G')$. Then $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$*

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

Proof. The standard proof is analogous to that of Proposition 6 from [48]. The alternative proof is in [43]. □

Let G be a dynamic expression and φ be the steady-state PMF for $SMC(G)$, $\varphi_{\underline{\leftrightarrow}_{ss}}$ be the steady-state PMF for $SMC_{\underline{\leftrightarrow}_{ss}}(G)$. By Proposition 7 (modified for $\mathcal{R}_{\mathcal{L}ss}(G)$), we have $\forall \mathcal{K} \in DR(G)/_{\mathcal{R}_{ss}(G)}$

$$\varphi_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) = \sum_{s \in \mathcal{K}} \varphi(s).$$

Thus, for every equivalence class $\mathcal{K} \in DR(G)/_{\mathcal{R}_{ss}(G)}$, the value of $\varphi_{\underline{\leftrightarrow}_{ss}}$ for $\mathcal{K}$ is the sum of all values of φ corresponding to the states from $\mathcal{K}$.

By Proposition 7, $\underline{\leftrightarrow}_{ss}$ preserves the quantitative properties of the stationary behaviour (the level of SMCs). We now intend to demonstrate that the qualitative properties of the stationary behaviour based on the multiaction labels are preserved as well (the level of transition systems).

Definition 23. *A* derived step trace *of a dynamic expression G is a chain $\Sigma = A_1 \cdots A_n \in (\mathbb{N}^{\mathcal{L}}_{fin})^*$, where $\exists s \in DR(G)\ s \xrightarrow{\Upsilon_1} s_1 \xrightarrow{\Upsilon_2} \cdots \xrightarrow{\Upsilon_n} s_n,\ \mathcal{L}(\Upsilon_i) = A_i$*

$(1 \le i \le n)$. *Then the* probability to execute the derived step trace Σ in s *is*

$$PT(\Sigma, s) = \sum_{\{\Upsilon_1,\ldots,\Upsilon_n \mid s=s_0 \stackrel{\Upsilon_1}{\to} s_1 \stackrel{\Upsilon_2}{\to} \cdots \stackrel{\Upsilon_n}{\to} s_n,\ \mathcal{L}(\Upsilon_i)=A_i\ \ (1\le i\le n)\}} \prod_{i=1}^{n} PT(\Upsilon_i, s_{i-1}).$$

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

Theorem 3. *Let G, G' be dynamic expressions with $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$ and φ be the steady-state PMF for $SMC(G)$, φ' be the steady-state PMF for $SMC(G')$ and Σ be a derived step trace of G and G'. Then $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$*

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

Proof. The proof is analogous to that of Theorem 4 from [48]. □

Let G be a dynamic expression, φ be the steady-state PMF for $SMC(G)$, $\varphi_{\underline{\leftrightarrow}_{ss}}$ be the steady-state PMF for $SMC_{\underline{\leftrightarrow}_{ss}}(G)$ and Σ be a derived step trace of G. By Theorem 3 (modified for $\mathcal{R}_{\mathcal{L}ss}(G)$), we get $\forall \mathcal{K} \in DR(G)/_{\mathcal{R}_{ss}(G)}$

$$\varphi_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) PT(\Sigma, \mathcal{K}) = \sum_{s \in \mathcal{K}} \varphi(s) PT(\Sigma, s),$$

where $\forall s \in \mathcal{K}\ PT(\Sigma, \mathcal{K}) = PT(\Sigma, s)$.

The following proposition demonstrates that, for two dynamic expressions related by $\underline{\leftrightarrow}_{ss}$, the sojourn time averages in an equivalence class coincide, as well as the sojourn time variances in it.

Proposition 8. *Let G, G' be dynamic expressions with $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$. Then $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$*

$$SJ_{\mathcal{R} \cap (DR(G))^2}(\mathcal{H} \cap DR(G)) = SJ_{\mathcal{R} \cap (DR(G'))^2}(\mathcal{H} \cap DR(G')),$$
$$VAR_{\mathcal{R} \cap (DR(G))^2}(\mathcal{H} \cap DR(G)) = VAR_{\mathcal{R} \cap (DR(G'))^2}(\mathcal{H} \cap DR(G')).$$

Proof. The proof is analogous to that of Proposition 7 from [48]. □

6.2. Preservation of performance and simplification of its analysis. Many performance indices are based on the steady-state probabilities to enter into a set of similar states or, after coming in it, to start a derived step trace from this set. Some of the indices are calculated using the average or the variance of sojourn time in a set of similar states. The similarity of states is captured by an equivalence relation, hence, the sets are the equivalence classes. Proposition 7, Theorem 3 and Proposition 8 guarantee coincidence of the mentioned indices for the expressions related by $\underline{\leftrightarrow}_{ss}$. Thus, $\underline{\leftrightarrow}_{ss}$ (hence, all the stronger equivalences we have considered) preserves performance of stochastic systems modeled by expressions of dtsdPBC.

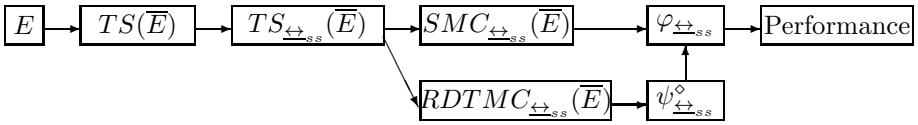

FIG. 1. Equivalence-based simplification of performance evaluation

It is easier to evaluate performance using an SMC with less states, since in this case the size of the TPM will be smaller, and we shall solve systems of less equations to calculate the steady-state probabilities. The reasoning above validates the following method of performance analysis simplification.

(1) The modeled system is specified by a static expression of dtsdPBC.
(2) The transition system of the expression is constructed.
(3) After treating the transition system for self-similarity, a step stochastic autobisimulation equivalence for the expression is determined.
(4) The quotient SMC is derived from the quotient transition system.
(5) The stationary probabilities and performance indices are calculated.

The limitation of the method above is its applicability only to the expressions such that their underlying SMCs contain exactly one closed communication class of states, and this class should also be ergodic to ensure uniqueness of the stationary distribution. If an SMC contains several closed communication classes of states that are all ergodic then several stationary distributions may exist, which depend on the initial PMF. There is an analytical method to determine stationary probabilities for SMCs of this kind as well [19]. The underlying SMC of every process expression has only one initial PMF (that at the time moment 0), hence, the stationary distribution will be unique in this case too. The general steady-state probabilities are then calculated as the sum of the stationary probabilities of all the ergodic classes of states, weighted by the probabilities to enter into these classes, starting from the initial state and passing through some transient states. In addition, it is worth applying the method only to the systems with similar subprocesses.

Alternatively, the results at the end of Section 5 allow us to simplify the steps 4 and 5 of the method above by constructing the reduced quotient DTMC (instead of the quotient underlying SMC) from the quotient transition system, followed by calculating the stationary probabilities of the quotient underlying SMC using that DTMC, and then obtaining the performance indices. In more detail, the quotient transition system $TS_{\underline{\leftrightarrow}_{ss}}(\overline{E})$ provides the information both about the probabilities to move between the equivalence classes of states $PM(\mathcal{K},\widetilde{\mathcal{K}})$ and about the equivalence classes of vanishing states $DR_V(\overline{E})/_{\mathcal{R}_{ss}(\overline{E})}$. That information is used to construct the reordered quotient transition probability matrix (TPM) $\mathbf{P}_{r\underline{\leftrightarrow}_{ss}}$, from which the TPM $\mathbf{P}^{\diamond}_{\underline{\leftrightarrow}_{ss}}$ for $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{E})$ is further obtained.

Figure 1 presents the main stages of the standard and alternative equivalence-based simplification of performance evaluation described above.

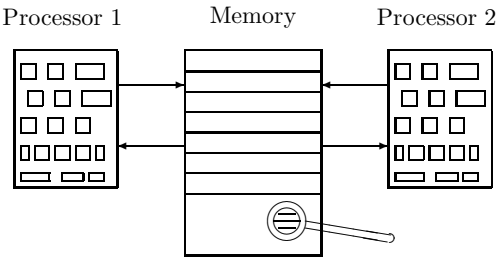

FIG. 2. The diagram of the shared memory system with maintenance

7 Generalized shared memory system with maintenance

Let us consider a model of two processors accessing a common shared memory described in [29, 2, 3] in the continuous time setting on GSPNs. We shall analyze this shared memory system in the discrete time stochastic setting of dtsdPBC, where concurrent execution of activities is possible, while no two transitions of a GSPN may fire simultaneously (in parallel). We also add to the system a feature of the memory maintenance. Our *generalized* model parameterizes the *standard* shared memory system by treating the probabilities and weights from its specification as variables (parameters). The model behaves 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 after which the instantaneous decision is made, if the memory is available. When the decision is made in favour of a processor, it starts acquisition of the memory and the other processor should wait until the former one ends its memory operations, and the system returns to the state with both active processors and available common memory. If the memory is available and not required then its maintenance can be initiated, followed by the short memory service works (for example, the checksum test) during a fixed period of time, after which the memory becomes available again. If the memory requirement and its maintenance initiation happen at the same time then the service works start and no decision on the memory allocation is made while the memory is maintained. The diagram of the system is depicted in Figure 2.

7.1. The concrete system. The meaning of actions from the dtsdPBC expressions which will specify the system modules is as follows. The action a corresponds to the system activation. The action c specifies the memory maintenance initiation. The action e means the short memory service (taking a fixed time of 1 unit). The actions r_i $(1 \leq i \leq 2)$ represent the common memory request (whose probability is 10 times greater than that of the maintenance initiation) of processor i. The actions d_i correspond to the (instantaneous) decision on the memory allocation in favour of the processor i. The actions m_i represent 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. For $a_1, \ldots, a_n \in Act$ ($n \in \mathbb{N}$), we shall abbreviate $\mathsf{sy}\ a_1 \cdots \mathsf{sy}\ a_n\ \mathsf{rs}\ a_1 \cdots \mathsf{rs}\ a_n$ to $\mathsf{sr}\ (a_1, \ldots, a_n)$.

We take general values for all multiaction probabilities and weights in the specification. Let all stochastic multiactions have the same generalized probability $\rho \in (0;1)$ and all deterministic ones have the same generalized weight $l \in \mathbb{R}_{>0}$. The resulting specification K of the generalized shared memory system with maintenance is as follows.

The static expression of the first processor is

$$K_1 = [(\{x_1\}, \rho) * ((\{r_1\}, \rho); (\{d_1, y_1\}, \natural_l^0); (\{m_1, z_1\}, \rho)) * \mathsf{Stop}].$$

The static expression of the second processor is

$$K_2 = [(\{x_2\}, \rho) * ((\{r_2\}, \rho); (\{d_2, y_2\}, \natural_l^0); (\{m_2, z_2\}, \rho)) * \mathsf{Stop}].$$

The static expression of the shared memory is

$$K_3 = [(\{a, \widehat{x_1}, \widehat{x_2}\}, \rho) * ((((\{c\}, \tfrac{\rho}{10}); (\{e\}, \natural_l^1))[]((\{\widehat{y_1}\}, \natural_l^0); (\{\widehat{z_1}\}, \rho))[] (((\{\widehat{y_2}\}, \natural_l^0); (\{\widehat{z_2}\}, \rho))) * \mathsf{Stop}].$$

The static expression of the generalized shared memory system with maintenance is

$$K = (K_1 \| K_2 \| K_3)\ \mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2).$$

Let us illustrate an effect of synchronization. As a result of the synchronization of immediate multiactions $(\{d_i, y_i\}, \natural_l^0)$ and $(\{\widehat{y_i}\}, \natural_l^0)$ we get $(\{d_i\}, \natural_{2l})$ $(1 \le i \le 2)$. The synchronization of stochastic multiactions $(\{m_i, z_i\}, \rho)$ and $(\{\widehat{z_i}\}, \rho)$ produces $(\{m_i\}, \rho^2)$ $(1 \le i \le 2)$. The result of synchronization of $(\{a, \widehat{x_1}, \widehat{x_2}\}, \rho)$ with $(\{x_1\}, \rho)$ is $(\{a, \widehat{x_2}\}, \rho^2)$, and that of synchronization of $(\{a, \widehat{x_1}, \widehat{x_2}\}, \rho)$ with $(\{x_2\}, \rho)$ is $(\{a, \widehat{x_1}\}, \rho^2)$. After applying synchronization to $(\{a, \widehat{x_2}\}, \rho^2)$ and $(\{x_2\}, \rho)$, as well as to $(\{a, \widehat{x_1}\}, \rho^2)$ and $(\{x_1\}, \rho)$, we get the same activity $(\{a\}, \rho^3)$.

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

$$\tilde{s}_1 = [([\overline{(\{x_1\}, \rho)} * ((\{r_1\}, \rho); (\{d_1, y_1\}, \natural_l^0); (\{m_1, z_1\}, \rho)) * \mathsf{Stop}] \| [\overline{(\{x_2\}, \rho)} * ((\{r_2\}, \rho); (\{d_2, y_2\}, \natural_l^0); (\{m_2, z_2\}, \rho)) * \mathsf{Stop}] \| [\overline{(\{a, \widehat{x_1}, \widehat{x_2}\}, \rho)} * ((((\{c\}, \tfrac{\rho}{10}); (\{e\}, \natural_l^1))[]((\{\widehat{y_1}\}, \natural_l^0); (\{\widehat{z_1}\}, \rho))[] (((\{\widehat{y_2}\}, \natural_l^0); (\{\widehat{z_2}\}, \rho))) * \mathsf{Stop}])\ \mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},$$

$$\tilde{s}_2 = [([(\{x_1\}, \rho) * \overline{((\{r_1\}, \rho)}; (\{d_1, y_1\}, \natural_l^0); (\{m_1, z_1\}, \rho)) * \mathsf{Stop}] \| [(\{x_2\}, \rho) * \overline{((\{r_2\}, \rho)}; (\{d_2, y_2\}, \natural_l^0); (\{m_2, z_2\}, \rho)) * \mathsf{Stop}] \| [(\{a, \widehat{x_1}, \widehat{x_2}\}, \rho) * \overline{((((\{c\}, \tfrac{\rho}{10}); (\{e\}, \natural_l^1))[]((\{\widehat{y_1}\}, \natural_l^0); (\{\widehat{z_1}\}, \rho))[] (((\{\widehat{y_2}\}, \natural_l^0); (\{\widehat{z_2}\}, \rho)))} * \mathsf{Stop}])\ \mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},$$

$$\tilde{s}_3 = [([(\{x_1\}, \rho) * \overline{((\{r_1\}, \rho)}; (\{d_1, y_1\}, \natural_l^0); (\{m_1, z_1\}, \rho)) * \mathsf{Stop}] \| [(\{x_2\}, \rho) * \overline{((\{r_2\}, \rho)}; (\{d_2, y_2\}, \natural_l^0); (\{m_2, z_2\}, \rho)) * \mathsf{Stop}] \| [(\{a, \widehat{x_1}, \widehat{x_2}\}, \rho) * ((((\{c\}, \tfrac{\rho}{10}); \overline{(\{e\}, \natural_l^1)^1})[]((\{\widehat{y_1}\}, \natural_l^0); (\{\widehat{z_1}\}, \rho))[] (((\{\widehat{y_2}\}, \natural_l^0); (\{\widehat{z_2}\}, \rho))) * \mathsf{Stop}])\ \mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},$$

$$\tilde{s}_4 = [([(\{x_1\},\rho) * ((\{r_1\},\rho); \overline{(\{d_1,y_1\},\natural_l^0)}; (\{m_1,z_1\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{x_2\},\rho) * (\overline{(\{r_2\},\rho)}; (\{d_2,y_2\},\natural_l^0); (\{m_2,z_2\},\rho)) * \mathsf{Stop}]\|$$
$$[(\overline{\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{c\},\tfrac{\rho}{10}); (\{e\},\natural_l^1))[]((\{\widehat{y_1}\},\natural_l^0); (\{\widehat{z_1}\},\rho))[]}$$
$$\overline{((\{\widehat{y_2}\},\natural_l^0); (\{\widehat{z_2}\},\rho)))} * \mathsf{Stop}])\ \mathsf{sr}\ (x_1,x_2,y_1,y_2,z_1,z_2)]_{\approx},$$

$$\tilde{s}_5 = [([(\{x_1\},\rho) * (\overline{(\{r_1\},\rho)}; \overline{(\{d_1,y_1\},\natural_l^0)}; (\{m_1,z_1\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{x_2\},\rho) * ((\{r_2\},\rho); \overline{(\{d_2,y_2\},\natural_l^0)}; (\{m_2,z_2\},\rho)) * \mathsf{Stop}]\|$$
$$[(\overline{\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{c\},\tfrac{\rho}{10}); (\{e\},\natural_l^1))[]((\{\widehat{y_1}\},\natural_l^0); (\{\widehat{z_1}\},\rho))[]}$$
$$\overline{((\{\widehat{y_2}\},\natural_l^0); (\{\widehat{z_2}\},\rho)))} * \mathsf{Stop}])\ \mathsf{sr}\ (x_1,x_2,y_1,y_2,z_1,z_2)]_{\approx},$$

$$\tilde{s}_6 = [([(\{x_1\},\rho) * ((\{r_1\},\rho); \overline{(\{d_1,y_1\},\natural_l^0)}; (\{m_1,z_1\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{x_2\},\rho) * ((\{r_2\},\rho); \overline{(\{d_2,y_2\},\natural_l^0)}; (\{m_2,z_2\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{c\},\tfrac{\rho}{10}); \overline{(\{e\},\natural_l^1)^1})[]((\{\widehat{y_1}\},\natural_l^0); (\{\widehat{z_1}\},\rho))[]$$
$$((\{\widehat{y_2}\},\natural_l^0); (\{\widehat{z_2}\},\rho))) * \mathsf{Stop}])\ \mathsf{sr}\ (x_1,x_2,y_1,y_2,z_1,z_2)]_{\approx},$$

$$\tilde{s}_7 = [([(\{x_1\},\rho) * ((\{r_1\},\rho); \overline{(\{d_1,y_1\},\natural_l^0)}; (\{m_1,z_1\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{x_2\},\rho) * (\overline{(\{r_2\},\rho)}; (\{d_2,y_2\},\natural_l^0); (\{m_2,z_2\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{c\},\tfrac{\rho}{10}); \overline{(\{e\},\natural_l^1)^1})[]((\{\widehat{y_1}\},\natural_l^0); (\{\widehat{z_1}\},\rho))[]$$
$$((\{\widehat{y_2}\},\natural_l^0); (\{\widehat{z_2}\},\rho))) * \mathsf{Stop}])\ \mathsf{sr}\ (x_1,x_2,y_1,y_2,z_1,z_2)]_{\approx},$$

$$\tilde{s}_8 = [([(\{x_1\},\rho) * (\overline{(\{r_1\},\rho)}; \overline{(\{d_1,y_1\},\natural_l^0)}; (\{m_1,z_1\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{x_2\},\rho) * ((\{r_2\},\rho); \overline{(\{d_2,y_2\},\natural_l^0)}; (\{m_2,z_2\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{c\},\tfrac{\rho}{10}); \overline{(\{e\},\natural_l^1)^1})[]((\{\widehat{y_1}\},\natural_l^0); (\{\widehat{z_1}\},\rho))[]$$
$$((\{\widehat{y_2}\},\natural_l^0); (\{\widehat{z_2}\},\rho))) * \mathsf{Stop}])\ \mathsf{sr}\ (x_1,x_2,y_1,y_2,z_1,z_2)]_{\approx},$$

$$\tilde{s}_9 = [([(\{x_1\},\rho) * ((\{r_1\},\rho); \overline{(\{d_1,y_1\},\natural_l^0)}; (\{m_1,z_1\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{x_2\},\rho) * ((\{r_2\},\rho); \overline{(\{d_2,y_2\},\natural_l^0)}; (\{m_2,z_2\},\rho)) * \mathsf{Stop}]\|$$
$$[(\overline{\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{c\},\tfrac{\rho}{10}); (\{e\},\natural_l^1))[]((\{\widehat{y_1}\},\natural_l^0); (\{\widehat{z_1}\},\rho))[]}$$
$$\overline{((\{\widehat{y_2}\},\natural_l^0); (\{\widehat{z_2}\},\rho)))} * \mathsf{Stop}])\ \mathsf{sr}\ (x_1,x_2,y_1,y_2,z_1,z_2)]_{\approx},$$

$$\tilde{s}_{10} = [([(\{x_1\},\rho) * ((\{r_1\},\rho); (\{d_1,y_1\},\natural_l^0); \overline{(\{m_1,z_1\},\rho)}) * \mathsf{Stop}]\|$$
$$[(\{x_2\},\rho) * (\overline{(\{r_2\},\rho)}; (\{d_2,y_2\},\natural_l^0); (\{m_2,z_2\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{c\},\tfrac{\rho}{10}); (\{e\},\natural_l^1))[]((\{\widehat{y_1}\},\natural_l^0); \overline{(\{\widehat{z_1}\},\rho)})[]$$
$$((\{\widehat{y_2}\},\natural_l^0); (\{\widehat{z_2}\},\rho))) * \mathsf{Stop}])\ \mathsf{sr}\ (x_1,x_2,y_1,y_2,z_1,z_2)]_{\approx},$$

$$\tilde{s}_{11} = [([(\{x_1\},\rho) * (\overline{(\{r_1\},\rho)}; (\{d_1,y_1\},\natural_l^0); (\{m_1,z_1\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{x_2\},\rho) * ((\{r_2\},\rho); (\{d_2,y_2\},\natural_l^0); \overline{(\{m_2,z_2\},\rho)}) * \mathsf{Stop}]\|$$
$$[(\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{c\},\tfrac{\rho}{10}); (\{e\},\natural_l^1))[]((\{\widehat{y_1}\},\natural_l^0); (\{\widehat{z_1}\},\rho))[]$$
$$((\{\widehat{y_2}\},\natural_l^0); \overline{(\{\widehat{z_2}\},\rho)})) * \mathsf{Stop}])\ \mathsf{sr}\ (x_1,x_2,y_1,y_2,z_1,z_2)]_{\approx},$$

$$\tilde{s}_{12} = [([(\{x_1\},\rho) * ((\{r_1\},\rho);\underline{(\{d_1,y_1\},\natural_l^0)};\overline{(\{m_1,z_1\},\rho)}) * \mathsf{Stop}]\|$$
$$[(\{x_2\},\rho) * ((\{r_2\},\rho);(\{d_2,y_2\},\natural_l^0);(\{m_2,z_2\},\rho)) * \mathsf{Stop}]\|$$
$$[(\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{c\},\tfrac{\rho}{10});(\{e\},\natural_l^1))[]((\{\widehat{y_1}\},\natural_l^0);\overline{(\{\widehat{z_1}\},\rho)}))[]$$
$$((\{\widehat{y_2}\},\natural_l^0);(\{\widehat{z_2}\},\rho))) * \mathsf{Stop}])\ \mathsf{sr}\ (x_1,x_2,y_1,y_2,z_1,z_2)]_{\approx},$$

$$\tilde{s}_{13} = [([(\{x_1\},\rho) * ((\{r_1\},\rho);\overline{(\{d_1,y_1\},\natural_l^0);(\{m_1,z_1\},\rho)}) * \mathsf{Stop}]\|$$
$$[(\{x_2\},\rho) * ((\{r_2\},\rho);(\{d_2,y_2\},\natural_l^0);\overline{(\{m_2,z_2\},\rho)}) * \mathsf{Stop}]\|$$
$$[(\{a,\widehat{x_1},\widehat{x_2}\},\rho) * ((((\{c\},\tfrac{\rho}{10});(\{e\},\natural_l^1))[]((\{\widehat{y_1}\},\natural_l^0);(\{\widehat{z_1}\},\rho))[]$$
$$((\{\widehat{y_2}\},\natural_l^0);\overline{(\{\widehat{z_2}\},\rho)})) * \mathsf{Stop}])\ \mathsf{sr}\ (x_1,x_2,y_1,y_2,z_1,z_2)]_{\approx},$$

We have $DR_{ST}(\overline{K}) = \{\tilde{s}_1,\tilde{s}_2,\tilde{s}_{10},\tilde{s}_{11},\tilde{s}_{12},\tilde{s}_{13}\}$, $DR_{WT}(\overline{K}) = \{\tilde{s}_3,\tilde{s}_6,\tilde{s}_7,\tilde{s}_8\}$ and $DR_V(\overline{K}) = \{\tilde{s}_4,\tilde{s}_5,\tilde{s}_9\}$.

The states are interpreted as follows: $\tilde{s}_1$ is the initial state in which the system is not activated; $\tilde{s}_2$: the system is activated and the memory is not requested and its maintenance is not initiated; $\tilde{s}_3$: the memory maintenance is initiated; $\tilde{s}_4$: the memory is requested by the first processor; $\tilde{s}_5$: the memory is requested by the second processor; $\tilde{s}_6$: the memory maintenance is initiated and the memory is requested by two processors; $\tilde{s}_7$: the memory maintenance is initiated and the memory is requested by the first processor; $\tilde{s}_8$: the memory maintenance is initiated and the memory is requested by the second processor; $\tilde{s}_9$: the memory is requested by two processors; $\tilde{s}_{10}$: the memory is allocated to the first processor; $\tilde{s}_{11}$: the memory is allocated to the second processor; $\tilde{s}_{12}$: the memory is allocated to the first processor and requested by the second processor; $\tilde{s}_{13}$: the memory is allocated to the second processor and requested by the first processor.

In Figure 3, the transition system $TS(\overline{K})$ is presented. In Figure 4, the underlying SMC $SMC(\overline{K})$ is depicted. Note that, in step semantics, we can execute the following activities in parallel: $(\{r_1\},\rho),(\{r_2\},\rho)$, as well as $(\{r_1\},\rho),(\{m_2\},\rho^2)$, and $(\{r_2\},\rho),(\{m_1\},\rho^2)$. We can also execute in parallel $(\{r_1\},\rho),(\{c\},\frac{\rho}{10})$, as well as $(\{r_2\},\rho),(\{c\},\frac{\rho}{10})$, and even $(\{r_1\},\rho),(\{r_2\},\rho),(\{c\},\frac{\rho}{10})$. The states $\tilde{s}_6,\tilde{s}_7,\tilde{s}_8,\tilde{s}_9$ only exist in step semantics, since they are reachable exclusively by executing all three activities $(\{r_1\},\rho),(\{r_2\},\rho),(\{c\},\frac{\rho}{10})$ or any pair of them in parallel.

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

$$\widetilde{SJ} = \left(\tfrac{1}{\rho^3},\tfrac{10}{\rho(21-12\rho+\rho^2)},1,0,0,1,1,1,0,\tfrac{1}{\rho(1+\rho-\rho^2)},\tfrac{1}{\rho(1+\rho-\rho^2)},\tfrac{1}{\rho^2},\tfrac{1}{\rho^2}\right).$$

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

$$\widetilde{VAR}=\left(\tfrac{1-\rho^3}{\rho^6},\tfrac{10(10-\rho)(1-\rho)^2}{\rho^2(21-12\rho+\rho^2)^2},0,0,0,0,0,0,0,\tfrac{(1-\rho^2)(1-\rho)}{\rho^2(1+\rho-\rho^2)^2},\tfrac{(1-\rho^2)(1-\rho)}{\rho^2(1+\rho-\rho^2)^2},\tfrac{1-\rho^2}{\rho^4},\tfrac{1-\rho^2}{\rho^4}\right).$$

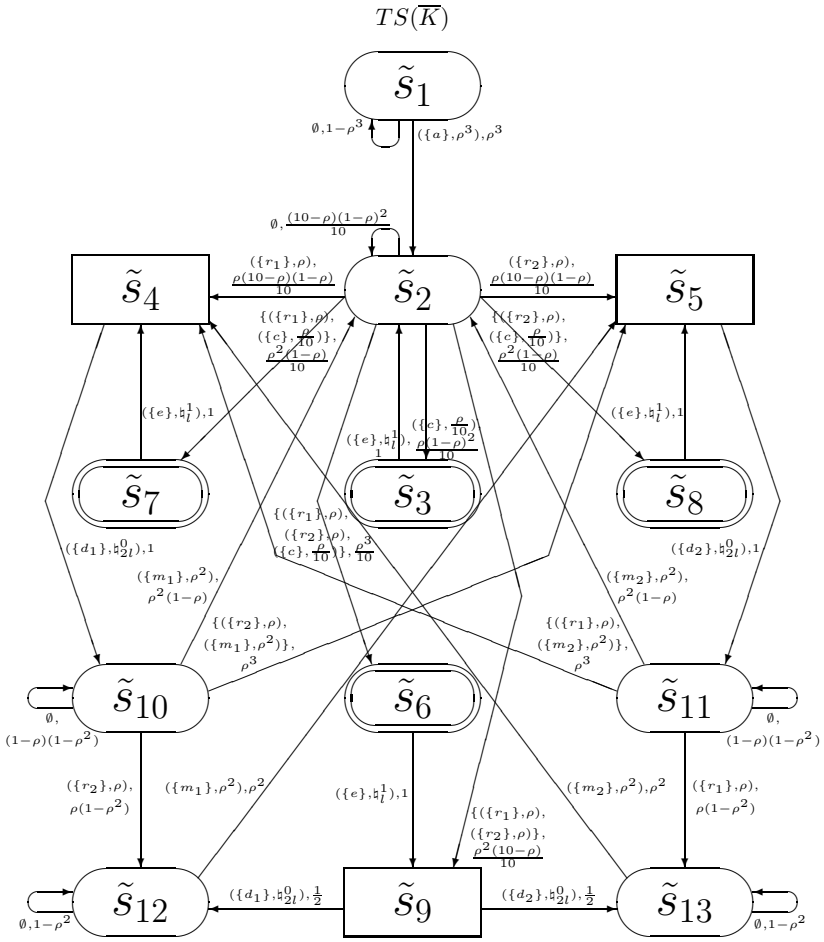

FIG. 3. The transition system of the generalized shared memory system with maintenance

Let us denote $\chi = 21 - 12\rho + \rho^2$, $\theta = 1 + \rho - \rho^2$ and $\mu = 10 - \rho$. The TPM for $EDTMC(\overline{K})$ is

$$\widetilde{\mathbf{P}}^* = \begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{(1-\rho)^2}{\chi} & \frac{\mu(1-\rho)}{\chi} & \frac{\mu(1-\rho)}{\chi} & \frac{\rho^2}{\chi} & \frac{\rho(1-\rho)}{\chi} & \frac{\rho(1-\rho)}{\chi} & \frac{\rho\mu}{\chi} & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\
0 & \frac{\rho(1-\rho)}{\theta} & 0 & 0 & \frac{\rho^2}{\theta} & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1-\rho^2}{\theta} & 0 \\
0 & \frac{\rho(1-\rho)}{\theta} & 0 & \frac{\rho^2}{\theta} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1-\rho^2}{\theta} \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}.$$

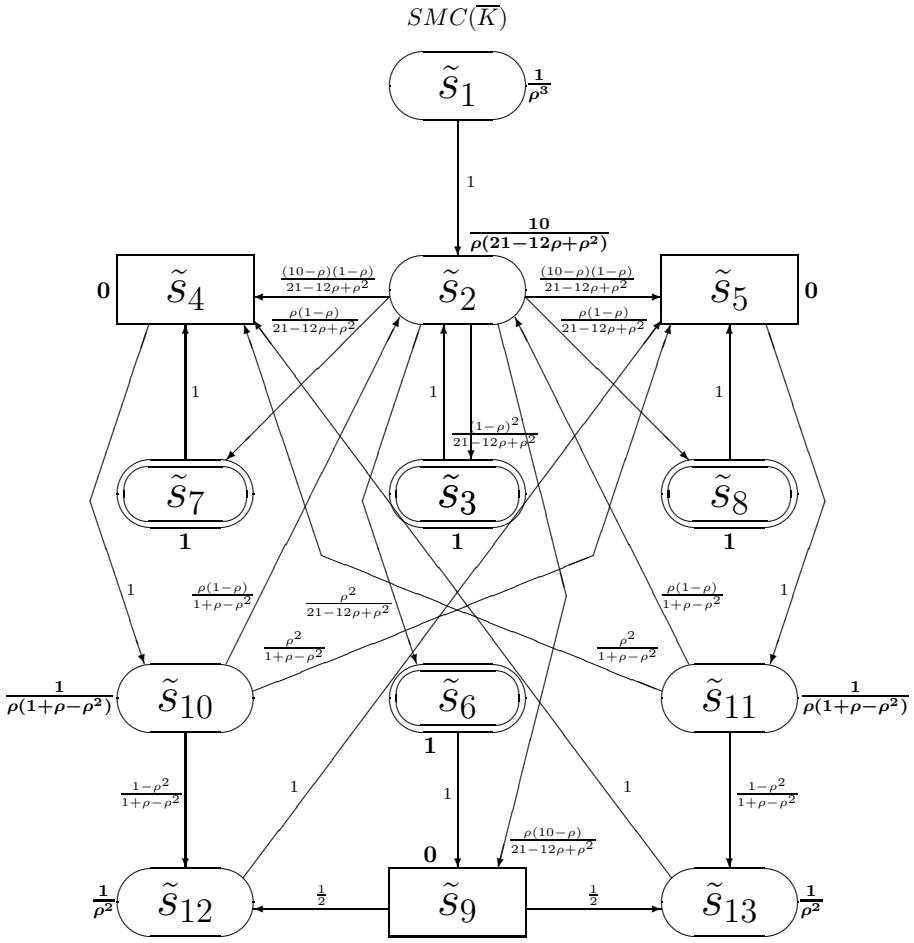

FIG. 4. The underlying SMC of the generalized shared memory system with maintenance

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

$$\tilde{\psi}^*=\tfrac{1}{60+32\rho-94\rho^2+23\rho^3-\rho^4}(0,\rho(1-\rho)(21-12\rho+\rho^2),\rho(1-\rho)^3,5(2-\rho)(1+\rho-\rho^2),$$
$$5(2-\rho)(1+\rho-\rho^2),\rho^3(1-\rho),\rho^2(1-\rho)^2,\rho^2(1-\rho)^2,10\rho^2(1-\rho),$$
$$5(2-\rho)(1+\rho-\rho^2),5(2-\rho)(1+\rho-\rho^2),5(1-\rho)(2+\rho),5(1-\rho)(2+\rho)).$$

The steady-state PMF $\tilde{\psi}^*$ weighted by $\widetilde{SJ}$ is

$$\tfrac{1}{\rho^2(60+32\rho-94\rho^2+23\rho^3-\rho^4)}(0,10\rho^2(1-\rho),\rho^3(1-\rho)^3,0,0,\rho^5(1-\rho),\rho^4(1-\rho)^2,$$
$$\rho^4(1-\rho)^2,0,5\rho(2-\rho),5\rho(2-\rho),5(1-\rho)(2+\rho),5(1-\rho)(2+\rho)).$$

It remains to normalize the steady-state weighted PMF by dividing it by the sum of its components

$$\tilde{\psi}^*\widetilde{SJ}^T=\frac{20+10\rho-10\rho^2-9\rho^3-\rho^4}{\rho^2(60+32\rho-94\rho^2+23\rho^3-\rho^4)}.$$

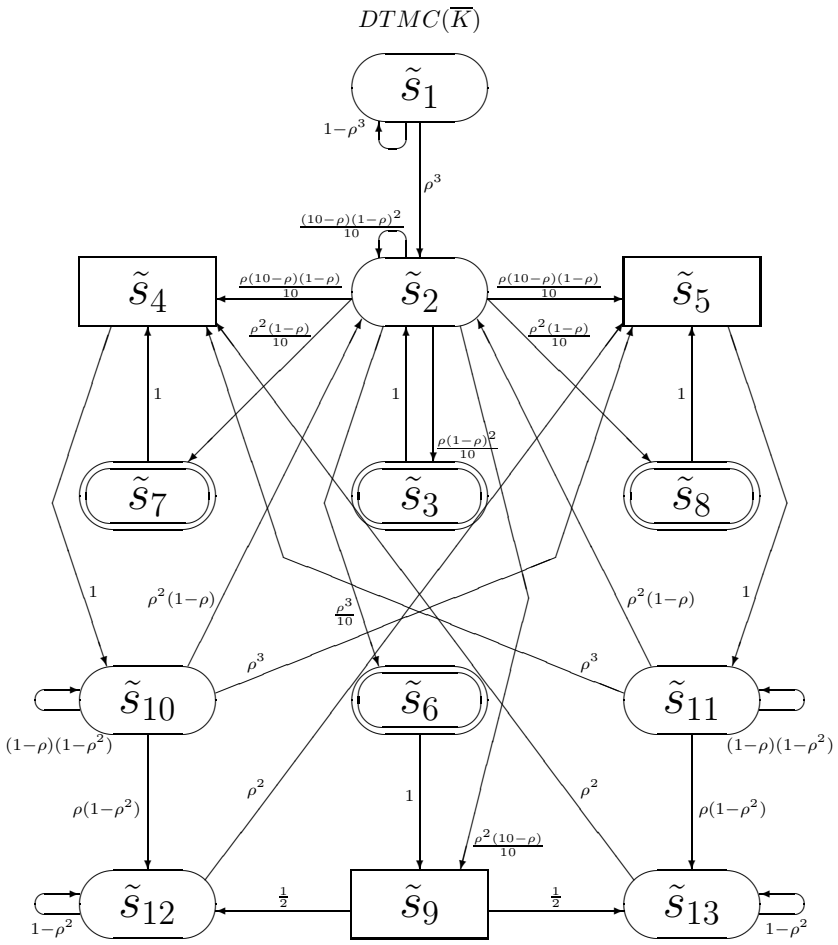

FIG. 5. The DTMC of the generalized shared memory system with maintenance

Thus, the steady-state PMF for $SMC(\overline{K})$ is

$$\tilde{\varphi}=\tfrac{1}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(0,10\rho^2(1-\rho),\rho^3(1-\rho)^3,0,0,\rho^5(1-\rho),\rho^4(1-\rho)^2,\\ \rho^4(1-\rho)^2,0,5\rho(2-\rho),5\rho(2-\rho),5(1-\rho)(2+\rho),5(1-\rho)(2+\rho)).$$

Otherwise, from $TS(\overline{K})$, we can construct the DTMC of $\overline{K}$, $DTMC(\overline{K})$, and then calculate $\tilde{\varphi}$ using it.

In Figure 5, the DTMC $DTMC(\overline{K})$ is depicted.

Let us denote $\mu = 10 - \rho$. The TPM for $DTMC(\overline{K})$ is

$$\widetilde{\mathbf{P}}=\begin{pmatrix}
1-\rho^3 & \rho^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & \frac{\mu(1-\rho)^2}{10} & \frac{\rho(1-\rho)^2}{10} & \frac{\rho\mu(1-\rho)}{10} & \frac{\rho\mu(1-\rho)}{10} & \frac{\rho^3}{10} & \frac{\rho^2(1-\rho)}{10} & \frac{\rho^2(1-\rho)}{10} & \frac{\rho^2\mu}{10} & 0 & 0 & 0 & 0\\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2}\\
0 & \rho^2(1-\rho) & 0 & 0 & \rho^3 & 0 & 0 & 0 & 0 & (1-\rho)(1-\rho^2) & 0 & \rho(1-\rho^2) & 0\\
0 & \rho^2(1-\rho) & 0 & \rho^3 & 0 & 0 & 0 & 0 & 0 & 0 & (1-\rho)(1-\rho^2) & 0 & \rho(1-\rho^2)\\
0 & 0 & 0 & 0 & \rho^2 & 0 & 0 & 0 & 0 & 0 & 0 & 1-\rho^2 & 0\\
0 & 0 & 0 & \rho^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1-\rho^2
\end{pmatrix}.$$

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

$$\tilde{\psi} = \tfrac{1}{20+10\rho+10\rho^2+\rho^3-21\rho^4}(0, 10\rho^2(1-\rho), \rho^3(1-\rho)^3, 5\rho^2(2-\rho)(1+\rho-\rho^2),\\ 5\rho^2(2-\rho)(1+\rho-\rho^2), \rho^5(1-\rho), \rho^4(1-\rho)^2, \rho^4(1-\rho)^2, 10\rho^4(1-\rho),\\ 5\rho(2-\rho), 5\rho(2-\rho), 5(1-\rho)(2+\rho), 5(1-\rho)(2+\rho)).$$

Remember that $DR_T(\overline{K}) = DR_{ST}(\overline{K}) \cup DR_{WT}(\overline{K}) = \{\tilde{s}_1, \tilde{s}_2, \tilde{s}_3, \tilde{s}_6, \tilde{s}_7, \tilde{s}_8, \tilde{s}_{10}, \tilde{s}_{11}, \tilde{s}_{12}, \tilde{s}_{13}\}$ and $DR_V(\overline{K}) = \{\tilde{s}_4, \tilde{s}_5, \tilde{s}_9\}$. Hence,

$$\textstyle\sum_{\tilde{s}\in DR_T(\overline{K})} \tilde{\psi}(\tilde{s}) = \tilde{\psi}(\tilde{s}_1) + \tilde{\psi}(\tilde{s}_2) + \tilde{\psi}(\tilde{s}_3) + \tilde{\psi}(\tilde{s}_6) + \tilde{\psi}(\tilde{s}_7) + \tilde{\psi}(\tilde{s}_8) +\\ \tilde{\psi}(\tilde{s}_{10}) + \tilde{\psi}(\tilde{s}_{11}) + \tilde{\psi}(\tilde{s}_{12}) + \tilde{\psi}(\tilde{s}_{13}) = \tfrac{20+10\rho-10\rho^2-9\rho^3-\rho^4}{20+10\rho+10\rho^2+\rho^3-21\rho^4}.$$

By Proposition 4 from [42], we have

$$\begin{aligned}
\tilde{\varphi}(\tilde{s}_1) &= 0 \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = 0,\\
\tilde{\varphi}(\tilde{s}_2) &= \tfrac{10\rho^2(1-\rho)}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
\tilde{\varphi}(\tilde{s}_3) &= \tfrac{\rho^3(1-\rho)^3}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{\rho^3(1-\rho)^3}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
\tilde{\varphi}(\tilde{s}_4) &= 0,\\
\tilde{\varphi}(\tilde{s}_5) &= 0,\\
\tilde{\varphi}(\tilde{s}_6) &= \tfrac{\rho^5(1-\rho)}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{\rho^5(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
\tilde{\varphi}(\tilde{s}_7) &= \tfrac{\rho^4(1-\rho)^2}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{\rho^4(1-\rho)^2}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
\tilde{\varphi}(\tilde{s}_8) &= \tfrac{\rho^4(1-\rho)^2}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{\rho^4(1-\rho)^2}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
\tilde{\varphi}(\tilde{s}_9) &= 0,\\
\tilde{\varphi}(\tilde{s}_{10}) &= \tfrac{5\rho(2-\rho)}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{5\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
\tilde{\varphi}(\tilde{s}_{11}) &= \tfrac{5\rho(2-\rho)}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{5\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
\tilde{\varphi}(\tilde{s}_{12}) &= \tfrac{5(1-\rho)(2+\rho)}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{5(1-\rho)(2+\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
\tilde{\varphi}(\tilde{s}_{13}) &= \tfrac{5(1-\rho)(2+\rho)}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{5(1-\rho)(2+\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}.
\end{aligned}$$

Thus, the steady-state PMF for $SMC(\overline{K})$ is

$$\tilde{\varphi} = \tfrac{1}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(0, 10\rho^2(1-\rho), \rho^3(1-\rho)^3, 0, 0, \rho^5(1-\rho), \rho^4(1-\rho)^2,\\ \rho^4(1-\rho)^2, 0, 5\rho(2-\rho), 5\rho(2-\rho), 5(1-\rho)(2+\rho), 5(1-\rho)(2+\rho)).$$

This coincides with the result obtained with the use of $\tilde{\psi}^*$ and $\widetilde{SJ}$.

Alternatively, from $TS(\overline{K})$, we can construct the reduced DTMC of $\overline{K}$, $RDTMC(\overline{K})$, and then calculate $\tilde{\varphi}$ using it.

Remember that $DR_{ST}(\overline{K}) = \{\tilde{s}_1, \tilde{s}_2, \tilde{s}_{10}, \tilde{s}_{11}, \tilde{s}_{12}, \tilde{s}_{13}\}$, $DR_{WT}(\overline{K}) = \{\tilde{s}_3, \tilde{s}_6, \tilde{s}_7, \tilde{s}_8\}$, $DR_V(\overline{K}) = \{\tilde{s}_4, \tilde{s}_5, \tilde{s}_9\}$. We reorder the elements of $DR(\overline{K})$, by moving vanishing states to the first positions and s-tangible states to the last positions: $\tilde{s}_4, \tilde{s}_5, \tilde{s}_9, \tilde{s}_3, \tilde{s}_6, \tilde{s}_7, \tilde{s}_8, \tilde{s}_1, \tilde{s}_2, \tilde{s}_{10}, \tilde{s}_{11}, \tilde{s}_{12}, \tilde{s}_{13}$.

Let us denote $\mu = 10 - \rho$. The reordered TPM for $DTMC(\overline{K})$ is

$$\widetilde{\mathbf{P}}_r = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1-\rho^3 & \rho^3 & 0 & 0 & 0 & 0 \\
\frac{\rho\mu(1-\rho)}{10} & \frac{\rho\mu(1-\rho)}{10} & \frac{\rho^2\mu}{10} & \frac{\rho(1-\rho)^2}{10} & \frac{\rho^3}{10} & \frac{\rho^2(1-\rho)}{10} & \frac{\rho^2(1-\rho)}{10} & 0 & \frac{\mu(1-\rho)^2}{10} & 0 & 0 & 0 & 0 \\
0 & \rho^3 & 0 & 0 & 0 & 0 & 0 & 0 & \rho^2(1-\rho) & (1-\rho)(1-\rho^2) & 0 & \rho(1-\rho^2) & 0 \\
\rho^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \rho^2(1-\rho) & 0 & (1-\rho)(1-\rho^2) & 0 & \rho(1-\rho^2) \\
0 & \rho^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1-\rho^2 & 0 \\
\rho^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1-\rho^2
\end{pmatrix}.$$

The result of the decomposing $\widetilde{\mathbf{P}}_r$ are the matrices

$$\widetilde{\mathbf{C}} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},\ \widetilde{\mathbf{D}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix},\ \widetilde{\mathbf{E}} = \begin{pmatrix}
0 & 0 & 0 \\
0 & 0 & 1 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
\frac{\rho\mu(1-\rho)}{10} & \frac{\rho\mu(1-\rho)}{10} & \frac{\rho^2\mu}{10} \\
0 & \rho^3 & 0 \\
\rho^3 & 0 & 0 \\
0 & \rho^2 & 0 \\
\rho^2 & 0 & 0
\end{pmatrix},$$

$$\widetilde{\mathbf{F}} = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1-\rho^3 & \rho^3 & 0 & 0 & 0 & 0 \\
\frac{\rho(1-\rho)^2}{10} & \frac{\rho^3}{10} & \frac{\rho^2(1-\rho)}{10} & \frac{\rho^2(1-\rho)}{10} & 0 & \frac{\mu(1-\rho)^2}{10} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \rho^2(1-\rho) & (1-\rho)(1-\rho^2) & 0 & \rho(1-\rho^2) & 0 \\
0 & 0 & 0 & 0 & 0 & \rho^2(1-\rho) & 0 & (1-\rho)(1-\rho^2) & 0 & \rho(1-\rho^2) \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1-\rho^2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1-\rho^2
\end{pmatrix}.$$

Since $\widetilde{\mathbf{C}}^1 = \mathbf{0}$, we have $\forall k > 0,\ \widetilde{\mathbf{C}}^k = \mathbf{0}$, hence, $l = 0$ and there are no loops among vanishing states. Then

$$\widetilde{\mathbf{G}} = \sum_{k=0}^{l} \widetilde{\mathbf{C}}^k = \widetilde{\mathbf{C}}^0 = \mathbf{I}.$$

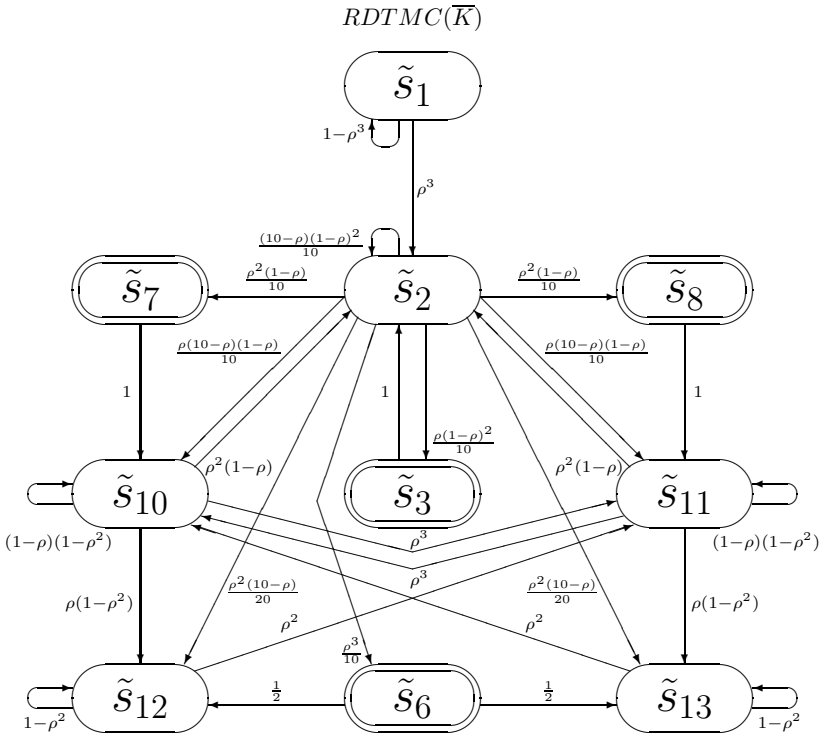

FIG. 6. The reduced DTMC of the generalized shared memory system with maintenance

Further, the TPM for $RDTMC(\overline{K})$ is

$$\widetilde{\mathbf{P}}^{\diamond} = \widetilde{\mathbf{F}} + \widetilde{\mathbf{E}}\widetilde{\mathbf{G}}\widetilde{\mathbf{D}} = \widetilde{\mathbf{F}} + \widetilde{\mathbf{E}}\mathbf{I}\widetilde{\mathbf{D}} = \widetilde{\mathbf{F}} + \widetilde{\mathbf{E}}\widetilde{\mathbf{D}} =$$

$$\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1-\rho^3 & \rho^3 & 0 & 0 & 0 & 0 \\
\frac{\rho(1-\rho)^2}{10} & \frac{\rho^3}{10} & \frac{\rho^2(1-\rho)}{10} & \frac{\rho^2(1-\rho)}{10} & 0 & \frac{\mu(1-\rho)^2}{10} & \frac{\rho\mu(1-\rho)}{10} & \frac{\rho\mu(1-\rho)}{10} & \frac{\rho^2\mu}{20} & \frac{\rho^2\mu}{20} \\
0 & 0 & 0 & 0 & 0 & \rho^2(1-\rho) & (1-\rho)(1-\rho^2) & \rho^3 & \rho(1-\rho^2) & 0 \\
0 & 0 & 0 & 0 & 0 & \rho^2(1-\rho) & \rho^3 & (1-\rho)(1-\rho^2) & 0 & \rho(1-\rho^2) \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \rho^2 & 1-\rho^2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \rho^2 & 0 & 0 & 1-\rho^2
\end{pmatrix}.$$

In Figure 6, the reduced DTMC $RDTMC(\overline{K})$ is presented.
Then the steady-state PMF for $RDTMC(\overline{K})$ is

$$\tilde{\psi}^{\diamond} = \tfrac{1}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(\rho^3(1-\rho)^3, \rho^5(1-\rho), \rho^4(1-\rho)^2, \rho^4(1-\rho)^2, 0,$$
$$10\rho^2(1-\rho), 5\rho(2-\rho), 5\rho(2-\rho), 5(1-\rho)(2+\rho), 5(1-\rho)(2+\rho)).$$

Note that $\tilde{\psi}^{\diamond} = (\tilde{\psi}^{\diamond}(\tilde{s}_3), \tilde{\psi}^{\diamond}(\tilde{s}_6), \tilde{\psi}^{\diamond}(\tilde{s}_7), \tilde{\psi}^{\diamond}(\tilde{s}_8), \tilde{\psi}^{\diamond}(\tilde{s}_1), \tilde{\psi}^{\diamond}(\tilde{s}_2), \tilde{\psi}^{\diamond}(\tilde{s}_{10}), \tilde{\psi}^{\diamond}(\tilde{s}_{11}), \tilde{\psi}^{\diamond}(\tilde{s}_{12}), \tilde{\psi}^{\diamond}(\tilde{s}_{13}))$. By Proposition 5 from [42], we have

$$\begin{aligned}
&\tilde{\varphi}(\tilde{s}_1) = 0, && \tilde{\varphi}(\tilde{s}_2) = \tfrac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
&\tilde{\varphi}(\tilde{s}_3) = \tfrac{\rho^3(1-\rho)^3}{20+10\rho-10\rho^2-9\rho^3-\rho^4}, && \tilde{\varphi}(\tilde{s}_4) = 0,\\
&\tilde{\varphi}(\tilde{s}_5) = 0, && \tilde{\varphi}(\tilde{s}_6) = \tfrac{\rho^5(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
&\tilde{\varphi}(\tilde{s}_7) = \tfrac{\rho^4(1-\rho)^2}{20+10\rho-10\rho^2-9\rho^3-\rho^4}, && \tilde{\varphi}(\tilde{s}_8) = \tfrac{\rho^4(1-\rho)^2}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
&\tilde{\varphi}(\tilde{s}_9) = 0, && \tilde{\varphi}(\tilde{s}_{10}) = \tfrac{5\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
&\tilde{\varphi}(\tilde{s}_{11}) = \tfrac{5\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}, && \tilde{\varphi}(\tilde{s}_{12}) = \tfrac{(1-\rho)(2+\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\
&\tilde{\varphi}(\tilde{s}_{13}) = \tfrac{(1-\rho)(2+\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}.
\end{aligned}$$

Thus, the steady-state PMF for $SMC(\overline{K})$ is

$$\tilde{\varphi} = \tfrac{1}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(0, 10\rho^2(1-\rho), \rho^3(1-\rho)^3, 0, 0, \rho^5(1-\rho), \rho^4(1-\rho)^2, \rho^4(1-\rho)^2, 0, 5\rho(2-\rho), 5\rho(2-\rho), 5(1-\rho)(2+\rho), 5(1-\rho)(2+\rho)).$$

This coincides with the result obtained with the use of $\tilde{\psi}^*$ and $\widetilde{SJ}$.

We can now calculate the main performance indices.

- The average recurrence time in the state $\tilde{s}_2$, where no processor requests the memory and its maintenance is not initiated, called the *average system run-through*, is $\frac{1}{\tilde{\varphi}_2} = \frac{20+10\rho-10\rho^2-9\rho^3-\rho^4}{10\rho^2(1-\rho)}$.
- The system is not activated only in the state $\tilde{s}_1$. Then the steady-state probability that the system is activated is $1 - \tilde{\varphi}_1 = 1 - 0 = 1$. The common memory is only available in the states $\tilde{s}_2, \tilde{s}_4, \tilde{s}_5, \tilde{s}_9$. Then the steady-state probability that the memory is available is $\tilde{\varphi}_2 + \tilde{\varphi}_4 + \tilde{\varphi}_5 + \tilde{\varphi}_9 = \frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4} + 0 + 0 + 0 = \frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$. The common memory is maintained only in the states $\tilde{s}_3, \tilde{s}_6, \tilde{s}_7, \tilde{s}_8$. Then the steady-state probability that the memory is maintained is $\tilde{\varphi}_3 + \tilde{\varphi}_6 + \tilde{\varphi}_7 + \tilde{\varphi}_8 = \frac{\rho^3(1-\rho)^3}{20+10\rho-10\rho^2-9\rho^3-\rho^4} + \frac{\rho^5(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4} + \frac{\rho^4(1-\rho)^2}{20+10\rho-10\rho^2-9\rho^3-\rho^4} + \frac{\rho^4(1-\rho)^2}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \frac{\rho^3(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$. Thus, the steady-state probability that the memory is used (i.e. neither available nor maintained), called the *shared memory utilization*, is $1 - \frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4} - \frac{\rho^3(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \frac{10(2+\rho-2\rho^2)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$.
- After activation of the system, we leave the state $\tilde{s}_1$ for ever, and the common memory is either requested or allocated or maintained in every remaining state, with exception of $\tilde{s}_2$. Thus, the *rate with which the necessity (also for maintenance) of shared memory emerges* coincides with the rate of leaving $\tilde{s}_2$, calculated as $\frac{\tilde{\varphi}_2}{SJ_2} = \frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4} \cdot \frac{\rho(21-12\rho+\rho^2)}{10} = \frac{\rho^3(1-\rho)(21-12\rho+\rho^2)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$.
- The parallel common memory request of two processors $\{(\{r_1\}, \rho), (\{r_2\}, \rho)\}$ is only possible from the state $\tilde{s}_2$. In this state, the request probability is the sum of the execution probabilities for all multisets

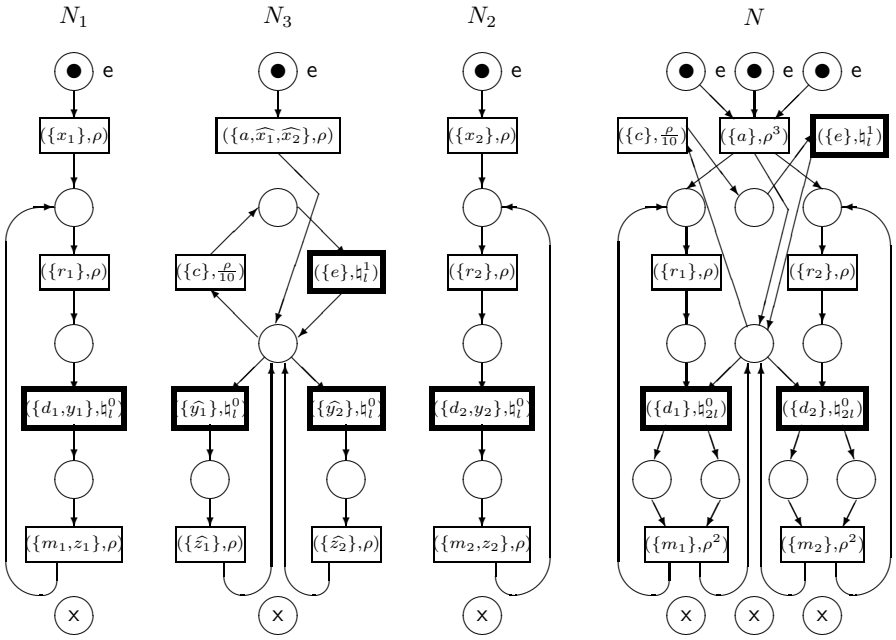

FIG. 7. The marked dtsd-boxes of the generalized two processors, shared memory and shared memory system with maintenance

of activities containing both $(\{r_1\},\rho)$ and $(\{r_2\},\rho)$. The *steady-state probability of the shared memory request from two processors* is $\tilde{\varphi}_2 \sum_{\{\Upsilon|(\{(\{r_1\},\rho),(\{r_2\},\rho)\}\subseteq\Upsilon\}} PT(\Upsilon,\tilde{s}_2) = \frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}\left(\frac{\rho^2(10-\rho)}{10}+\frac{\rho^3}{10}\right) = \frac{10\rho^4(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$.

- The common memory request of the first processor $(\{r_1\},\rho)$ is possible from the states $\tilde{s}_2, \tilde{s}_{11}$. In each of them, the request probability is the sum of the execution probabilities for all sets of activities containing $(\{r_1\},\rho)$. The *steady-state probability of the shared memory request from the first processor* is $\tilde{\varphi}_2 \sum_{\{\Upsilon|(\{r_1\},\rho)\in\Upsilon\}} PT(\Upsilon,\tilde{s}_2) + \tilde{\varphi}_{11} \sum_{\{\Upsilon|(\{r_1\},\rho)\in\Upsilon\}} PT(\Upsilon,\tilde{s}_{11}) = \frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}\left(\frac{\rho(10-\rho)(1-\rho)}{10}+\frac{\rho^2(1-\rho)}{10}+\frac{\rho^2(10-\rho)}{10}+\frac{\rho^3}{10}\right)+ \frac{5\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(\rho(1-\rho^2)+\rho^3) = \frac{5\rho^2(2+\rho-2\rho^2)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$.

In Figure 7, the marked dtsd-boxes corresponding to the dynamic expressions of the generalized two processors, shared memory and shared memory system with maintenance are presented, i.e. $N_i = Box_{dtsd}(\overline{K_i})$ $(1 \le i \le 3)$ and $N = Box_{dtsd}(\overline{K})$.

7.2. The abstract system and its reduction. Let us consider a modification of the generalized shared memory system with maintenance via abstraction from identifiers of the processors, i.e. such that the processors are indistinguishable. For example, we can just see that a processor requires

memory or the memory is allocated to it but cannot observe which processor is it. We call this system the abstract generalized shared memory system with maintenance. To implement the abstraction, we replace the actions r_i, d_i, m_i $(1 \leq i \leq 2)$ in the system specification by r, d, m, respectively.

The static expression of the first processor is

$$L_1 = [(\{x_1\}, \rho) * ((\{r\}, \rho); (\{d, y_1\}, \natural_l^0); (\{m, z_1\}, \rho)) * \mathsf{Stop}].$$

The static expression of the second processor is

$$L_2 = [(\{x_2\}, \rho) * ((\{r\}, \rho); (\{d, y_2\}, \natural_l^0); (\{m, z_2\}, \rho)) * \mathsf{Stop}].$$

The static expression of the shared memory is

$$L_3 = [(\{a, \widehat{x_1}, \widehat{x_2}\}, \rho) * ((((\{c\}, \tfrac{\rho}{10}); (\{e\}, \natural_l^1))[]((\{\widehat{y_1}\}, \natural_l^0); (\{\widehat{z_1}\}, \rho))[] ((\{\widehat{y_2}\}, \natural_l^0); (\{\widehat{z_2}\}, \rho))) * \mathsf{Stop}].$$

The static expression of the abstract generalized shared memory system with maintenance is

$$L = (L_1 \| L_2 \| L_3)\ \mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2).$$

$DR(\overline{L}) = \{\tilde{s}'_1, \ldots, \tilde{s}'_{13}\}$ resembles $DR(\overline{K})$, and $TS(\overline{L})$ is similar to $TS(\overline{K})$. We have $SMC(\overline{L}) \simeq SMC(\overline{K})$. Thus, the average sojourn time vectors of $\overline{L}$ and $\overline{K}$, the TPMs and the steady-state PMFs for $EDTMC(\overline{L})$ and $EDTMC(\overline{K})$ coincide.

The first, second, third and fourth performance indices are the same for the generalized system and its abstract modification. Let us consider the following performance index which is specific to the abstract system.

- The common memory request of a processor $(\{r\}, \rho)$ is possible from the states $\tilde{s}'_2, \tilde{s}'_{10}, \tilde{s}'_{11}$. In each of them, the request probability is the sum of the execution probabilities for all sets of activities containing $(\{r\}, \rho)$. The *steady-state probability of the shared memory request from a processor* is $\tilde{\varphi}_2 \sum_{\{\Upsilon | (\{r\}, \rho) \in \Upsilon\}} PT(\Upsilon, \tilde{s}'_2) +$ $\tilde{\varphi}_{10} \sum_{\{\Upsilon | (\{r\}, \rho) \in \Upsilon\}} PT(\Upsilon, \tilde{s}'_{10}) + \tilde{\varphi}_{11} \sum_{\{\Upsilon | (\{r\}, \rho) \in \Upsilon\}} PT(\Upsilon, \tilde{s}'_{11}) =$ $\frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4} \left(\frac{\rho(10-\rho)(1-\rho)}{10} + \frac{\rho(10-\rho)(1-\rho)}{10} + \frac{\rho^2(1-\rho)}{10} + \frac{\rho^2(1-\rho)}{10} + \frac{\rho^2(10-\rho)}{10} + \frac{\rho^3}{10} \right) + \frac{5\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(\rho(1-\rho^2)+\rho^3) +$ $\frac{5\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(\rho(1-\rho^2)+\rho^3) = \frac{10\rho^2(2-\rho)(1+\rho-\rho^2)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}.$

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

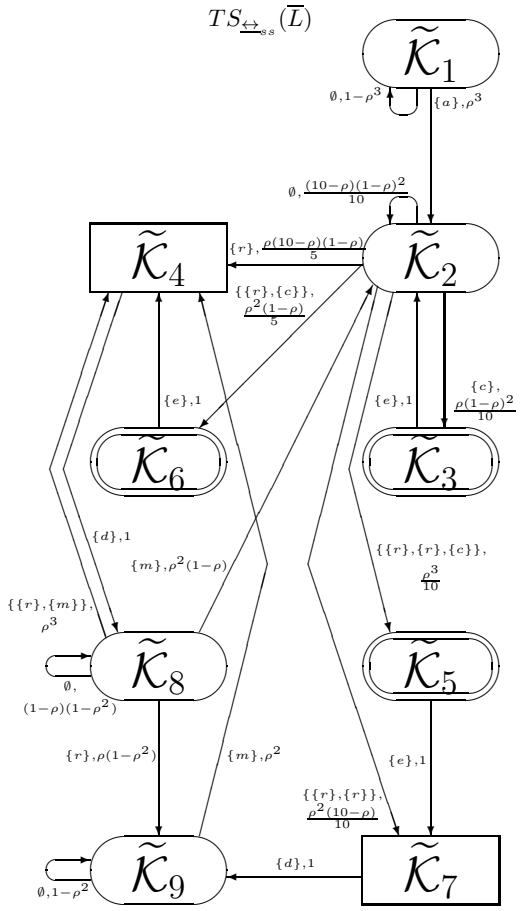

FIG. 8. The quotient transition system of the abstract generalized shared memory system with maintenance

$\widetilde{\mathcal{K}}_8 = \{\tilde{s}'_{10}, \tilde{s}'_{11}\}$ (the memory is allocated to a processor), $\widetilde{\mathcal{K}}_9 = \{\tilde{s}'_{12}, \tilde{s}'_{13}\}$ (the memory is allocated to a processor and requested by another processor).

We have $DR_{ST}(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_1, \widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_8, \widetilde{\mathcal{K}}_9\}$, $DR_{WT}(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_5, \widetilde{\mathcal{K}}_6\}$, $DR_V(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_4, \widetilde{\mathcal{K}}_7\}$.

In Figure 8, the quotient transition system $TS_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is presented. In Figure 9, the quotient underlying SMC $SMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is depicted. Note that, in step semantics, we may execute the following multiactions in parallel: $\{r\}, \{r\}$, as well as $\{r\}, \{m\}$. We can also execute in parallel $\{r\}, \{c\}$, and even $\{r\}, \{r\}, \{c\}$. The states $\widetilde{\mathcal{K}}_5, \widetilde{\mathcal{K}}_6, \widetilde{\mathcal{K}}_7$ only exist in step semantics, since they are reachable exclusively by executing all three multiactions $\{r\}, \{r\}, \{c\}$ or any pair of them in parallel.

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

$$\widetilde{SJ}' = \left(\frac{1}{\rho^3}, \frac{10}{\rho(21-12\rho+\rho^2)}, 1, 0, 1, 1, 0, \frac{1}{\rho(1+\rho-\rho^2)}, \frac{1}{\rho^2}\right).$$

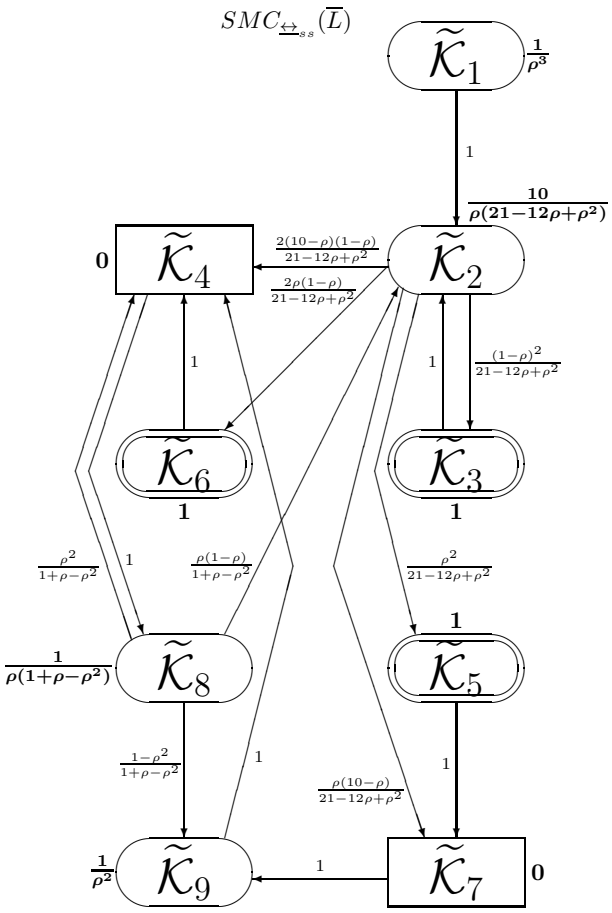

FIG. 9. The quotient underlying SMC of the abstract generalized shared memory system with maintenance

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

$$\widetilde{VAR}' = \left(\frac{1-\rho^3}{\rho^6}, \frac{10(10-\rho)(1-\rho)^2}{\rho^2(21-12\rho+\rho^2)^2}, 0, 0, 0, 0, 0, \frac{(1-\rho^2)(1-\rho)}{\rho^2(1+\rho-\rho^2)^2}, \frac{1-\rho^2}{\rho^4} \right).$$

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

$$\widetilde{\mathbf{P}}'^* = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{(1-\rho)^2}{21-12\rho+\rho^2} & \frac{2(10-\rho)(1-\rho)}{21-12\rho+\rho^2} & \frac{\rho^2}{21-12\rho+\rho^2} & \frac{2\rho(1-\rho)}{21-12\rho+\rho^2} & \frac{\rho(10-\rho)}{21-12\rho+\rho^2} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & \frac{\rho(1-\rho)}{1+\rho-\rho^2} & 0 & \frac{\rho^2}{1+\rho-\rho^2} & 0 & 0 & 0 & 0 & \frac{1-\rho^2}{1+\rho-\rho^2} \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

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

$$\tilde{\psi}'^{*} = \tfrac{1}{60+32\rho-94\rho^2+23\rho^3-\rho^4}(0, \rho(1-\rho)(21-12\rho+\rho^2), \rho(1-\rho)^3, \\ 10(2-\rho)(1+\rho-\rho^2), \rho^3(1-\rho), 2\rho^2(1-\rho)^2, 10\rho^2(1-\rho), \\ 10(2-\rho)(1+\rho-\rho^2), 10(1-\rho)(2+\rho)).$$

The steady-state PMF $\tilde{\psi}'^{*}$ weighted by $\widetilde{SJ}'$ is

$$\tfrac{1}{60+32\rho-94\rho^2+23\rho^3-\rho^4}(0, 10(1-\rho), \rho(1-\rho)^3, 0, \rho^3(1-\rho), 2\rho^2(1-\rho)^2, 0, \\ 10(2-\rho), 10(1-\rho)(2+\rho)).$$

It remains to normalize the steady-state weighted PMF by dividing it by the sum of its components

$$\tilde{\psi}'^{*}\widetilde{SJ}'^{T} = \frac{20+10\rho-10\rho^2-9\rho^3-\rho^4}{\rho^2(60+32\rho-94\rho^2+23\rho^3-\rho^4)}.$$

Thus, the steady-state PMF for $SMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is

$$\tilde{\varphi}' = \tfrac{1}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(0, 10\rho^2(1-\rho), \rho^3(1-\rho)^3, 0, \rho^5(1-\rho), 2\rho^4(1-\rho)^2, \\ 0, 10\rho(2-\rho), 10(1-\rho)(2+\rho)).$$

Otherwise, from $TS_{\underline{\leftrightarrow}_{ss}}(\overline{L})$, we can construct the quotient DTMC of $\overline{L}$, $DTMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$, and then calculate $\tilde{\varphi}'$ using it.

In Figure 10, the quotient DTMC $DTMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is depicted.

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

$$\widetilde{\mathbf{P}}' = \begin{pmatrix} 1-\rho^3 & \rho^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{(10-\rho)(1-\rho)^2}{10} & \frac{\rho(1-\rho)^2}{10} & \frac{\rho(10-\rho)(1-\rho)}{5} & \frac{\rho^3}{10} & \frac{\rho^2(1-\rho)}{5} & \frac{\rho^2(10-\rho)}{10} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & \rho^2(1-\rho) & 0 & \rho^3 & 0 & 0 & 0 & (1-\rho)(1-\rho^2) & \rho(1-\rho^2) \\ 0 & 0 & 0 & \rho^2 & 0 & 0 & 0 & 0 & 1-\rho^2 \end{pmatrix}.$$

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

$$\tilde{\psi}' = \tfrac{1}{20+10\rho+10\rho^2+\rho^3-21\rho^4}(0, 10\rho^2(1-\rho), \rho^3(1-\rho)^3, 10\rho^2(2-\rho)(1+\rho-\rho^2), \\ \rho^5(1-\rho), 2\rho^4(1-\rho)^2, 10\rho^4(1-\rho), 10\rho(2-\rho), 10(1-\rho)(2+\rho)).$$

Remember that $DR_T(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = DR_{ST}(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} \cup DR_{WT}(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_1, \widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_5, \widetilde{\mathcal{K}}_6, \widetilde{\mathcal{K}}_8, \widetilde{\mathcal{K}}_9\}$ and $DR_V(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_4, \widetilde{\mathcal{K}}_7\}$. Hence,

$$\textstyle\sum_{\widetilde{\mathcal{K}} \in DR_T(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})}} \tilde{\psi}'(\widetilde{\mathcal{K}}) = \tilde{\psi}'(\widetilde{\mathcal{K}}_1) + \tilde{\psi}'(\widetilde{\mathcal{K}}_2) + \tilde{\psi}'(\widetilde{\mathcal{K}}_3) + \tilde{\psi}'(\widetilde{\mathcal{K}}_5) + \tilde{\psi}'(\widetilde{\mathcal{K}}_6) + \\ \tilde{\psi}'(\widetilde{\mathcal{K}}_8) + \tilde{\psi}'(\widetilde{\mathcal{K}}_9) = \tfrac{20+10\rho-10\rho^2-9\rho^3-\rho^4}{20+10\rho+10\rho^2+\rho^3-21\rho^4}.$$

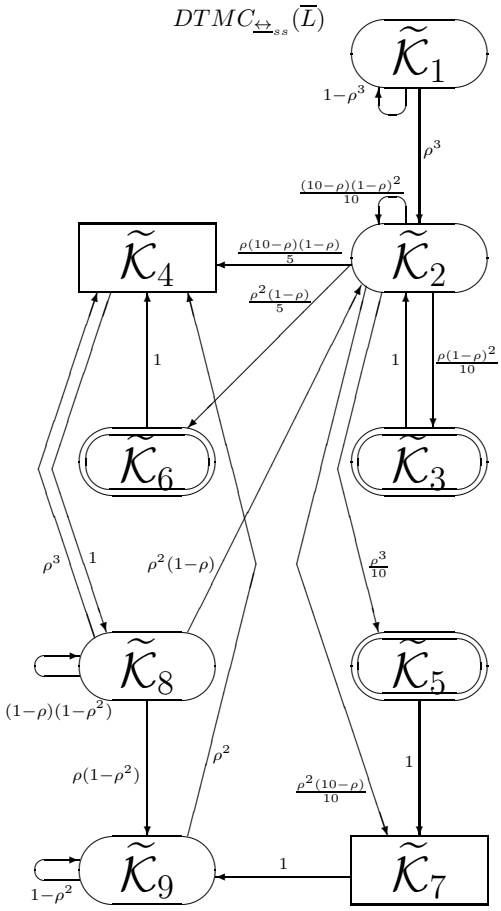

FIG. 10. The quotient DTMC of the abstract generalized shared memory system with maintenance

By the "quotient" analogue of Proposition 4 from [42], we have

$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_1) = 0 \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = 0,$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_2) = \tfrac{10\rho^2(1-\rho)}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_3) = \tfrac{\rho^3(1-\rho)^3}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{\rho^3(1-\rho)^3}{20+10\rho-10\rho^2-9\rho^3-\rho^4},$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_4) = 0,$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_5) = \tfrac{\rho^5(1-\rho)}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{\rho^5(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_6) = \tfrac{2\rho^4(1-\rho)^2}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{2\rho^4(1-\rho)^2}{20+10\rho-10\rho^2-9\rho^3-\rho^4},$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_7) = 0,$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_8) = \tfrac{10\rho(2-\rho)}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{10\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4},$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_9) = \tfrac{10(1-\rho)(2+\rho)}{20+10\rho+10\rho^2+\rho^3-21\rho^4} \cdot \tfrac{20+10\rho+10\rho^2+\rho^3-21\rho^4}{20+10\rho-10\rho^2-9\rho^3-\rho^4} = \tfrac{10(1-\rho)(2+\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}.$$

Thus, the steady-state PMF for $SMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is

$$\tilde{\varphi}' = \tfrac{1}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(0, 10\rho^2(1-\rho), \rho^3(1-\rho)^3, 0, \rho^5(1-\rho), 2\rho^4(1-\rho)^2, 0, 10\rho(2-\rho), 10(1-\rho)(2+\rho)).$$

This coincides with the result obtained with the use of $\tilde{\psi}'^*$ and $\widetilde{SJ}'$. **Alternatively**, from $TS_{\underline{\leftrightarrow}_{ss}}(\overline{L})$, we can construct the reduced quotient DTMC of $\overline{L}$, $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$, and then calculate $\tilde{\varphi}'$ using it. By Proposition 9 from [43], it coincides with the quotient reduced DTMC of $\overline{L}$, the quotient of $RDTMC(\overline{L})$.

Remember that $DR_{ST}(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_1, \widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_8, \widetilde{\mathcal{K}}_9\}$, $DR_{WT}(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_5, \widetilde{\mathcal{K}}_6\}$, $DR_V(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_4, \widetilde{\mathcal{K}}_7\}$. We reorder the elements of $DR(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})}$, by moving the equivalence classes of vanishing states to the first positions and those of s-tangible states to the last positions: $\widetilde{\mathcal{K}}_4, \widetilde{\mathcal{K}}_7, \widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_5, \widetilde{\mathcal{K}}_6, \widetilde{\mathcal{K}}_1, \widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_8, \widetilde{\mathcal{K}}_9$.

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

$$\widetilde{\mathbf{P}}'_r = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1-\rho^3 & \rho^3 & 0 & 0 \\ \frac{\rho(10-\rho)(1-\rho)}{5} & \frac{\rho^2(10-\rho)}{10} & \frac{\rho(1-\rho)^2}{10} & \frac{\rho^3}{10} & \frac{\rho^2(1-\rho)}{5} & 0 & \frac{(10-\rho)(1-\rho)^2}{10} & 0 & 0 \\ \rho^3 & 0 & 0 & 0 & 0 & 0 & \rho^2(1-\rho) & (1-\rho)(1-\rho^2) & \rho(1-\rho^2) \\ \rho^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1-\rho^2 \end{pmatrix}.$$

The result of the decomposing $\widetilde{\mathbf{P}}'_r$ are the matrices

$$\widetilde{\mathbf{C}}' = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \ \widetilde{\mathbf{D}}' = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \ \widetilde{\mathbf{E}}' = \begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 0 \\ \frac{\rho(10-\rho)(1-\rho)}{5} & \frac{\rho^2(10-\rho)}{10} \\ \rho^3 & 0 \\ \rho^2 & 0 \end{pmatrix},$$

$$\widetilde{\mathbf{F}}' = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1-\rho^3 & \rho^3 & 0 & 0 \\ \frac{\rho(1-\rho)^2}{10} & \frac{\rho^3}{10} & \frac{\rho^2(1-\rho)}{5} & 0 & \frac{(10-\rho)(1-\rho)^2}{10} & 0 & 0 \\ 0 & 0 & 0 & 0 & \rho^2(1-\rho) & (1-\rho)(1-\rho^2) & \rho(1-\rho^2) \\ 0 & 0 & 0 & 0 & 0 & 0 & 1-\rho^2 \end{pmatrix}.$$

Since $\widetilde{\mathbf{C}}'^1 = \mathbf{0}$, we have $\forall k > 0$, $\widetilde{\mathbf{C}}'^k = \mathbf{0}$, hence, $l = 0$ and there are no loops among vanishing states. Then

$$\widetilde{\mathbf{G}}' = \sum_{k=0}^{l} \widetilde{\mathbf{C}}'^l = \widetilde{\mathbf{C}}'^0 = \mathbf{I}.$$

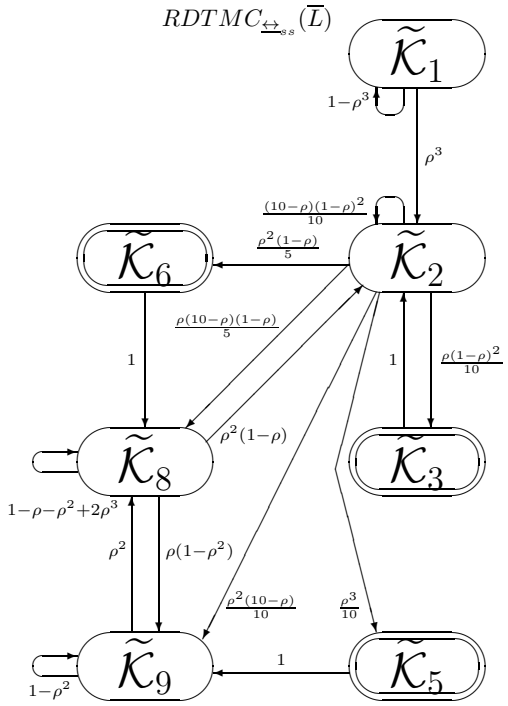

FIG. 11. The reduced quotient DTMC of the abstract generalized shared memory system with maintenance

Further, the TPM for $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is

$$\widetilde{\mathbf{P}}'^{\diamond} = \widetilde{\mathbf{F}}' + \widetilde{\mathbf{E}}'\widetilde{\mathbf{G}}'\widetilde{\mathbf{D}}' = \widetilde{\mathbf{F}}' + \widetilde{\mathbf{E}}'\mathbf{I}\widetilde{\mathbf{D}}' = \widetilde{\mathbf{F}}' + \widetilde{\mathbf{E}}'\widetilde{\mathbf{D}}' =$$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1-\rho^3 & \rho^3 & 0 & 0 \\ \frac{\rho(1-\rho)^2}{10} & \frac{\rho^3}{10} & \frac{\rho^2(1-\rho)}{5} & 0 & \frac{(10-\rho)(1-\rho)^2}{10} & \frac{\rho(10-\rho)(1-\rho)}{5} & \frac{\rho^2(10-\rho)}{10} \\ 0 & 0 & 0 & 0 & \rho^2(1-\rho) & 1-\rho-\rho^2+2\rho^3 & \rho(1-\rho^2) \\ 0 & 0 & 0 & 0 & 0 & \rho^2 & 1-\rho^2 \end{pmatrix}.$$

In Figure 11, the reduced quotient DTMC $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is presented. Then the steady-state PMF for $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is

$$\tilde{\psi}'^{\diamond} = \tfrac{1}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(\rho^3(1-\rho)^3, \rho^5(1-\rho), 2\rho^4(1-\rho)^2, 0, 10\rho^2(1-\rho), 10\rho(2-\rho), 10(1-\rho)(2+\rho)).$$

Note that $\tilde{\psi}'^{\diamond}=(\tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_3),\tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_5),\tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_6),\tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_1),\tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_2),\tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_8),\tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_9))$. By the "quotient" analogue of Proposition 5 from [42], we have

$$\begin{array}{lll}\tilde{\varphi}'(\widetilde{\mathcal{K}}_1)=0, & \tilde{\varphi}'(\widetilde{\mathcal{K}}_2)=\frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}, & \tilde{\varphi}'(\widetilde{\mathcal{K}}_3)=\frac{\rho^3(1-\rho)^3}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\ \tilde{\varphi}'(\widetilde{\mathcal{K}}_4)=0, & \tilde{\varphi}'(\widetilde{\mathcal{K}}_5)=\frac{\rho^5(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}, & \tilde{\varphi}'(\widetilde{\mathcal{K}}_6)=\frac{2\rho^4(1-\rho)^2}{20+10\rho-10\rho^2-9\rho^3-\rho^4},\\ \tilde{\varphi}'(\widetilde{\mathcal{K}}_7)=0, & \tilde{\varphi}'(\widetilde{\mathcal{K}}_8)=\frac{10\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}, & \tilde{\varphi}'(\widetilde{\mathcal{K}}_9)=\frac{10(1-\rho)(2+\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}.\end{array}$$

Thus, the steady-state PMF for $SMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is

$$\tilde{\varphi}'=\frac{1}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(0,10\rho^2(1-\rho),\rho^3(1-\rho)^3,0,\rho^5(1-\rho),2\rho^4(1-\rho)^2,\\ 0,10\rho(2-\rho),10(1-\rho)(2+\rho)).$$

This coincides with the result obtained with the use of $\tilde{\psi}'^{*}$ and $\widetilde{SJ}'$.
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 and its maintenance is not initiated, called the *average system run-through*, is $\frac{1}{\tilde{\varphi}'_2}=\frac{20+10\rho-10\rho^2-9\rho^3-\rho^4}{10\rho^2(1-\rho)}$.
- The system is not activated only in the state $\widetilde{\mathcal{K}}_1$. Then the steady-state probability that the system is activated is $1-\tilde{\varphi}'_1=1-0=1$. The common memory is available in the states $\widetilde{\mathcal{K}}_2,\widetilde{\mathcal{K}}_4,\widetilde{\mathcal{K}}_7$. Then the steady-state probability that the memory is available is $\tilde{\varphi}'_2+\tilde{\varphi}'_4+\tilde{\varphi}'_7=\frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}+0+0=\frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$. The common memory is maintained only in the states $\widetilde{\mathcal{K}}_3,\widetilde{\mathcal{K}}_5,\widetilde{\mathcal{K}}_6$. Then the steady-state probability that the memory is maintained is $\tilde{\varphi}'_3+\tilde{\varphi}'_5+\tilde{\varphi}'_6=\frac{\rho^3(1-\rho)^3}{20+10\rho-10\rho^2-9\rho^3-\rho^4}+\frac{\rho^5(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}+\frac{2\rho^4(1-\rho)^2}{20+10\rho-10\rho^2-9\rho^3-\rho^4}=\frac{\rho^3(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$. Thus, the steady-state probability that the memory is used (i.e. neither available nor maintained), the *shared memory utilization*, is $1-\frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}-\frac{\rho^3(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}=\frac{10(2+\rho-2\rho^2)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$.
- After activation of the system, we leave the state $\widetilde{\mathcal{K}}_1$ for ever, and the common memory is either requested or allocated or maintained in every remaining state, with exception of $\widetilde{\mathcal{K}}_2$. Thus, the *rate with which the necessity (also for maintenance) of shared memory emerges* coincides with the rate of leaving $\widetilde{\mathcal{K}}_2$, calculated as $\frac{\tilde{\varphi}'_2}{\widetilde{SJ}'_2}=\frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}\cdot\frac{\rho(21-12\rho+\rho^2)}{10}=\frac{\rho^3(1-\rho)(21-12\rho+\rho^2)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$.
- The parallel common memory request of two processors $\{\{r\},\{r\}\}$ is possible from the state $\widetilde{\mathcal{K}}_2$. In this state, the request probability is the sum of the execution probabilities for all multisets of multiactions containing $\{r\}$ twice. The *steady-state probability of the shared memory request from two processors* is

$\tilde{\varphi}'_2 \sum_{\{A,\widetilde{\mathcal{K}}|\{\{r\},\{r\}\}\subseteq A,\ \widetilde{\mathcal{K}}_2 \stackrel{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_2,\widetilde{\mathcal{K}}) =$
$\frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}\left(\frac{\rho^2(10-\rho)}{10}+\frac{\rho^3}{10}\right) = \frac{10\rho^4(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}.$

- The common memory request of a processor $\{r\}$ is possible from the states $\widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_8$. In each of them, the request probability is the sum of the execution probabilities for all multisets of multiactions containing $\{r\}$. The *steady-state probability of the shared memory request from a processor* is $\tilde{\varphi}'_2 \sum_{\{A,\widetilde{\mathcal{K}}|\{r\}\in A,\ \widetilde{\mathcal{K}}_2 \stackrel{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_2,\widetilde{\mathcal{K}}) +$
$\tilde{\varphi}'_8 \sum_{\{A,\widetilde{\mathcal{K}}|\{r\}\in A,\ \widetilde{\mathcal{K}}_8 \stackrel{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_8,\widetilde{\mathcal{K}}) =$
$\frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}\cdot\left(\frac{\rho(10-\rho)(1-\rho)}{5}+\frac{\rho^2(1-\rho)}{5}+\frac{\rho^2(10-\rho)}{10}+\frac{\rho^3}{10}\right)+$
$\frac{10\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}(\rho(1-\rho^2)+\rho^3) = \frac{10\rho^2(2-\rho)(1+\rho-\rho^2)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}.$

The performance indices are the same for the "complete" and the "quotient" abstract generalized shared memory systems with maintenance. The coincidence of the first, second and third performance indices obviously illustrates the results of Proposition 7 and Proposition 8 (both modified for $\mathcal{R}_{\mathcal{L}ss}(\overline{L})$). The coincidence of the fourth performance index is due to Theorem 3 (modified for $\mathcal{R}_{\mathcal{L}ss}(\overline{L})$): one should just apply its result to the derived step traces $\{\{r\},\{r\}\}$ and $\{\{r\},\{r\},\{c\}\}$ of the expression $\overline{L}$ and itself, and then sum the left and right parts of the two resulting equalities. The coincidence of the fifth performance index is due to Theorem 3 (modified for $\mathcal{R}_{\mathcal{L}ss}(\overline{L})$): one should just apply its result to the derived step traces $\{\{r\}\}$, $\{\{r\},\{c\}\}$, $\{\{r\},\{r\}\}$, $\{\{r\},\{r\},\{c\}\}$, $\{\{r\},\{m\}\}$ of the expression $\overline{L}$ and itself, and then sum the left and right parts of the five 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 with maintenance in its steady state. Remember that $\rho \in (0;1)$ is the probability of every stochastic multiaction in the specification of the system. The closer is ρ to 0, the less is the probability to execute some activities at every discrete time tick, 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 tick, hence, the system will most probably *operate*.

Since $\tilde{\varphi}'_1 = \tilde{\varphi}'_4 = \tilde{\varphi}'_7 = 0$, only $\tilde{\varphi}'_2 = \frac{10\rho^2(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$, $\tilde{\varphi}'_3 = \frac{\rho^3(1-\rho)^3}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$, $\tilde{\varphi}'_5 = \frac{\rho^5(1-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$, $\tilde{\varphi}'_6 = \frac{2\rho^4(1-\rho)^2}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$, $\tilde{\varphi}'_8 = \frac{10\rho(2-\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$, $\tilde{\varphi}'_9 = \frac{10(1-\rho)(2+\rho)}{20+10\rho-10\rho^2-9\rho^3-\rho^4}$ depend on ρ. In Figure 12, the plots of $\tilde{\varphi}'_2$, $\tilde{\varphi}'_8$, $\tilde{\varphi}'_9$ (large probability masses) as functions of ρ are depicted. In Figure 13, the plots of $\tilde{\varphi}'_3$, $\tilde{\varphi}'_5$, $\tilde{\varphi}'_6$ (small probability masses) as functions of ρ are drawn. Note that, however, we do not allow $\rho = 0$ or $\rho = 1$.

One can see that $\tilde{\varphi}'_2$, $\tilde{\varphi}'_3$, $\tilde{\varphi}'_5$, $\tilde{\varphi}'_6$, $\tilde{\varphi}'_8$ tend to 0 and $\tilde{\varphi}'_9$ tends to 1 when ρ approaches 0. Thus, when ρ is closer to 0, the probability that the memory is allocated to a processor and requested by another processor increases, hence, we have *more unsatisfied memory requests*.

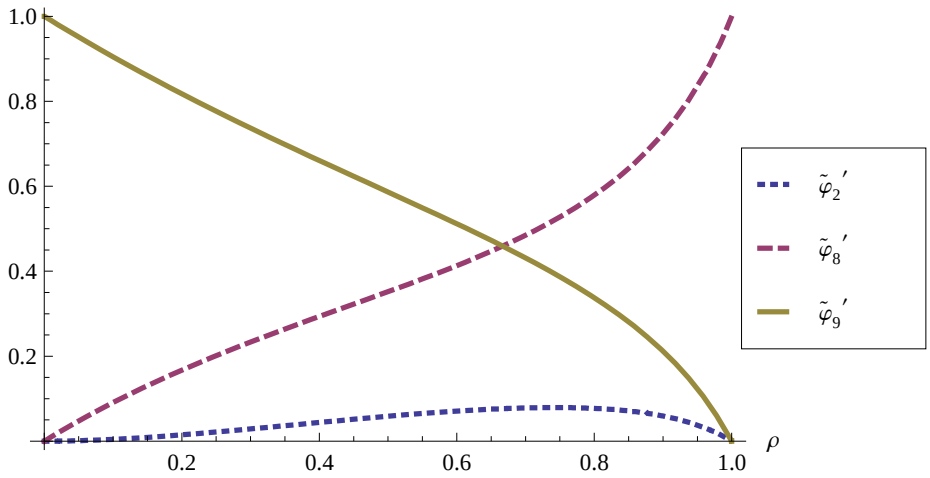

FIG. 12. Steady-state probabilities $\tilde{\varphi}'_2$, $\tilde{\varphi}'_8$, $\tilde{\varphi}'_9$ (large probability masses) as functions of the parameter ρ

Further, $\tilde{\varphi}'_2$, $\tilde{\varphi}'_3$, $\tilde{\varphi}'_5$, $\tilde{\varphi}'_6$, $\tilde{\varphi}'_9$ tend to 0 and $\tilde{\varphi}'_8$ tends to 1 when ρ approaches 1. Thus, when ρ is closer to 1, the probability that the memory is allocated to a processor (and not requested by another processor) increases, hence, we have *less unsatisfied memory requests*.

The maximal value 0.0792 of $\tilde{\varphi}'_2$ is reached when $\rho \approx 0.7427$. Then the probability that the system is activated and the memory is not requested and its maintenance is not initiated is maximal, i.e. the *maximal shared memory availability* is about 8%.

The maximal value 0.0007 of $\tilde{\varphi}'_3$ is reached when $\rho \approx 0.5158$. Then the probability that the memory maintenance is initiated is maximal, i.e. the *maximal shared memory maintenance necessity* is about 0.1%.

The maximal value 0.0044 of $\tilde{\varphi}'_5$ is reached when $\rho \approx 0.8724$. then the probability that the memory maintenance is initiated and the memory is requested by two processors is maximal, i.e. the *maximal double (parallel) demand of shared memory during its maintenance* is about 0.4%.

The maximal value 0.0023 of $\tilde{\varphi}'_6$ is reached when $\rho \approx 0.7015$. Then the probability that the memory maintenance is initiated and the memory is requested by a (single) processor is maximal, i.e. the *maximal single demand of shared memory during its maintenance* is about 0.2%.

In Figure 14, the plot of the average system run-through, calculated as $\frac{1}{\tilde{\varphi}'_2}$, as a function of ρ is depicted. One can see that the run-through tends to ∞ when ρ approaches 0 or 1. Its minimal value 12.6259 is reached when $\rho \approx 0.7427$. To speed up operation of the system, one should take the parameter ρ closer to 0.7427.

The first curve in Figure 15 represents the shared memory utilization, calculated as $1 - \tilde{\varphi}'_{2-7}$, where $\tilde{\varphi}'_{2-7} = \tilde{\varphi}'_2 + \tilde{\varphi}'_3 + \tilde{\varphi}'_4 + \tilde{\varphi}'_5 + \tilde{\varphi}'_6 + \tilde{\varphi}'_7$, as a function of ρ. One can see that the utilization tends to 1 both when ρ

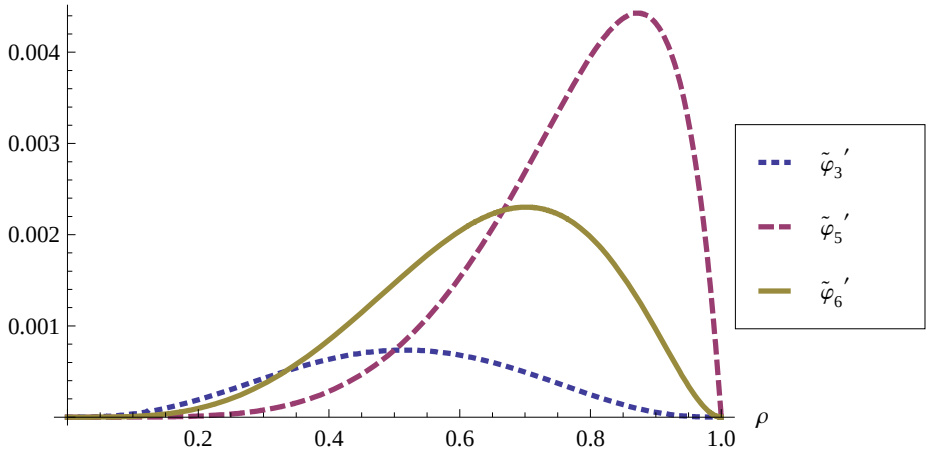

FIG. 13. Steady-state probabilities $\tilde{\varphi}_3'$, $\tilde{\varphi}_5'$, $\tilde{\varphi}_6'$ (small probability masses) as functions of the parameter ρ

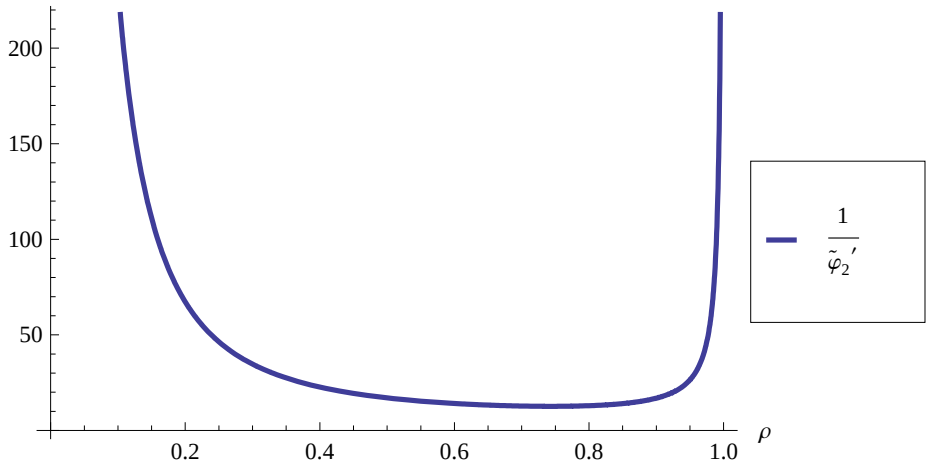

FIG. 14. Average system run-through $\frac{1}{\tilde{\varphi}_2'}$ as a function of the parameter ρ

approaches 0 and when ρ approaches 1. The minimal value 0.9149 of the utilization is reached when $\rho \approx 0.7494$. Thus, the *minimal shared memory utilization* is about 91%. To increase the utilization, one should take the parameter ρ closer to 0 or 1.

The second curve in Figure 15 represents the rate with which the necessity of shared memory emerges, calculated as $\frac{\tilde{\varphi}_2'}{SJ_2'}$, as a function of ρ. One can see that the rate tends to 0 both when ρ approaches 0 and when ρ approaches 1. The maximal value 0.0749 of the rate is reached when $\rho \approx 0.7723$. Thus, the *maximal rate with which the necessity of shared memory emerges* is about $\frac{1}{13}$. To decrease the that rate, one should take the parameter ρ closer to 0 or 1.

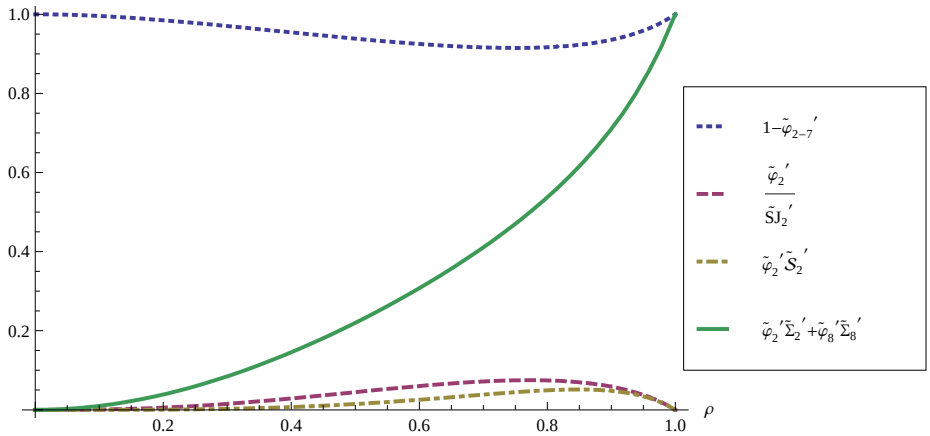

FIG. 15. Some performance indices as functions of the parameter ρ

The third curve in Figure 15 represents the steady-state probability of the shared memory request from two processors, calculated as $\tilde{\varphi}'_2\widetilde{\mathcal{S}}'_2$, where $\widetilde{\mathcal{S}}'_2 = \sum_{\{A,\widetilde{\mathcal{K}}|\{\{r\},\{r\}\}\subseteq A,\ \widetilde{\mathcal{K}}_2 \stackrel{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}})$, as function of ρ. One can see that the probability tends to 0 both when ρ approaches 0 and when ρ approaches 1. The maximal value 0.0514 of the probability is reached when $\rho \approx 0.8486$. To decrease the mentioned probability, one should take the parameter ρ closer to 0 or 1.

The fourth curve in Figure 15 represents the steady-state probability of the shared memory request from a processor, calculated as $\tilde{\varphi}'_2\widetilde{\Sigma}'_2 + \tilde{\varphi}'_8\widetilde{\Sigma}'_8$, where $\widetilde{\Sigma}'_i = \sum_{\{A,\widetilde{\mathcal{K}}|\{r\}\in A,\ \widetilde{\mathcal{K}}_i \stackrel{A}{\to} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_i, \widetilde{\mathcal{K}}),\ i \in \{2, 8\}$, as a function of ρ. One can see that the probability tends to 0 when ρ approaches 0 and it tends to 1 when ρ approaches 1. To increase the probability, one should take the parameter ρ closer to 1.

8 Conclusion

In this paper, we have considered dtsdPBC, an extension with discrete stochastic and deterministic time of Petri box calculus (PBC) [9, 11, 10, 8]. Stochastic process algebra dtsdPBC has a parallel step operational semantics, based on labeled probabilistic transition systems, and a Petri net denotational semantics in terms of dtsd-boxes, a special subclass of LDTSDPNs [41]. Step stochastic bisimulation equivalence of the process expressions is used to reduce their transition systems and Markov chains (SMCs, DTMCs and RDTMCs) with the quotienting. That equivalence guarantees identity of the steady-state probabilities, sojourn time averages and variances in the equivalence classes. Hence, the equivalence preserves the stationary performance measures and can be used for minimization of the state space.

The properties of the mentioned equivalence permit to provide dtsdPBC with a method of modeling, quotient reduction and simplified performance evaluation of concurrent stochastic systems that maintains the semantic parallelism, which stems from simultaneous executions. We have presented a case study of a generalization of the shared memory system with maintenance, by allowing for variable probabilities and weights in its specification. The generalized probability has been interpreted as a parameter of the performance index functions. The influence of the parameter value to the system's performance has been analyzed with the goal of optimization.

Future work consists in constructing a congruence relation for dtsdPBC, i.e. the equivalence that withstands application of all operations of the algebra. A possible candidate is a stronger version of the equivalence with respect to transition systems, with two extra transitions skip and redo, like in sPBC [22]. Moreover, recursion operation could be added to dtsdPBC to increase specification power of the algebra.

References

[1] W.M.P. van der Aalst, K.M. van Hee, H.A. Reijers, *Analysis of discrete-time stochastic Petri nets*, Statistica Neerlandica, **54**:2 (2000), 237–255. Zbl 0994.68091

[2] G. Balbo, *Introduction to stochastic Petri nets*, Lecture Notes in Computer Science, **2090** (2001), 84–155. Zbl 0990.68092

[3] G. Balbo, *Introduction to generalized stochastic Petri nets*, Lecture Notes in Computer Science, **4486** (2007), 83–131. Zbl 1323.68400

[4] J.A. Bergstra, J.W. Klop, *Algebra of communicating processes with abstraction*, Theoretical Computer Science, **37** (1985), 77–121.

[5] M. Bernardo, *A survey of Markovian behavioral equivalences*, Lecture Notes in Computer Science, **4486** (2007), 180–219. Zbl 1323.68402

[6] M. Bernardo, *Non-bisimulation-based Markovian behavioral equivalences*, Journal of Logic and Algebraic Programming, **72**:1 (2007), 3–49. Zbl 1121.68077

[7] M. Bernardo, R. Gorrieri, *A tutorial on EMPA: a theory of concurrent processes with nondeterminism, priorities, probabilities and time*, Theoretical Computer Science, **202**:1–2 (1998), 1–54.

[8] E. Best, R. Devillers, *Petri net primer: a compendium on the core model, analysis, and synthesis*, Computer Science Foundations and Applied Logic Series (CSFAL), Springer International Publishing / Birkhäuser, 2024.

[9] E. Best, R. Devillers, J.G. Hall, *The box calculus: a new causal algebra with multi-label communication*, Lecture Notes in Computer Science, **609** (1992), 21–69.

[10] E. Best, R. Devillers, M. Koutny, *Petri net algebra*, EATCS Monographs on Theoretical Computer Science, Springer, 2001.

[11] E. Best, M. Koutny, *A refined view of the box algebra*, Lecture Notes in Computer Science, **935** (1995), 1–20.

[12] T. Bolognesi, F. Lucidi, S. Trigila, *From timed Petri nets to timed LOTOS*, Proc. IFIP WG6.1 10^{th} Int. Symp. on Protocol Specification, Testing and Verification (PSTV) 1990 (L. Logrippo, R.L. Probert, H. Ural, eds.), Ottawa, Canada, 395–408, Elsevier Science Publishers (North-Holland), Amsterdam, The Netherlands 1990.

[13] H.M. Hanish, *Analysis of place/transition nets with timed-arcs and its application to batch process control*, Lecture Notes in Computer Science, **691** (1993), 282–299.

[14] H. Hermanns, M. Rettelbach, *Syntax, semantics, equivalences and axioms for MTIPP*, Proc. 2^{nd} Int. Workshop on Process Algebras and Performance Modelling (PAPM)

1994 (U. Herzog, M. Rettelbach, eds.), Regensberg / Erlangen, Germany, Arbeitsberichte des IMMD, **27**:4 (1994), 71–88.

[15] J. Hillston, *A compositional approach to performance modelling*, Cambridge University Press, Cambridge, UK, 1996. Zbl 1080.68003

[16] C.A.R. Hoare, *Communicating sequential processes*, Prentice-Hall, London, UK, 1985.

[17] C.-C. Jou, S.A. Smolka, *Equivalences, congruences and complete axiomatizations for probabilistic processes*, Lecture Notes in Computer Science, **458** (1990), 367–383.

[18] M. Koutny, *A compositional model of time Petri nets*, Lecture Notes in Computer Science, **1825** (2000), 303–322.

[19] V.G. Kulkarni, *Modeling and analysis of stochastic systems*, Texts in Statistical Science series, Chapman and Hall / CRC Press, 2010. Zbl 1191.60003

[20] K.G. Larsen, A. Skou, *Bisimulation through probabilistic testing*, Information and Computation, **94**:1 (1991), 1–28.

[21] H. Macià, V. Valero, D.C. Cazorla, F. Cuartero, *Introducing the iteration in sPBC*, Lecture Notes in Computer Science, **3235** (2004), 292–308. Zbl 1110.68420

[22] H. Macià, V. Valero, F. Cuartero, D. de Frutos, *A congruence relation for sPBC*, Formal Methods in System Design, **32**:2 (2008), 85–128. Zbl 1138.68040

[23] H. Macià, V. Valero, F. Cuartero, M.C. Ruiz, *sPBC: a Markovian extension of Petri box calculus with immediate multiactions*, Fundamenta Informaticae, **87**:3–4 (2008), 367–406. Zbl 1154.68092

[24] H. Macià, V. Valero, F. Cuartero, M.C. Ruiz, I.V. Tarasyuk, *Modelling a video conference system with sPBC*, Applied Mathematics and Information Sciences **10**:2 (2016), 475–493.

[25] H. Macià, V. Valero, D. de Frutos, *sPBC: a Markovian extension of finite Petri box calculus*, Proc. 9^{th} IEEE Int. Workshop on Petri Nets and Performance Models (PNPM) 2001, Aachen, Germany, 207–216, IEEE Computer Society Press, 2001.

[26] O. Marroquín, D. de Frutos, *TPBC: timed Petri box calculus*, Technical Report, Departamento de Sistemas Infofmáticos y Programación, Universidad Complutense de Madrid, Spain, 2000 (in Spanish).

[27] O. Marroquín, D. de Frutos, *Extending the Petri box calculus with time*, Lecture Notes in Computer Science, **2075** (2001), 303–322. Zbl 0986.68082

[28] M.A. Marsan, *Stochastic Petri nets: an elementary introduction*, Lecture Notes in Computer Science, **424** (1990), 1–29.

[29] M.A. Marsan, G. Balbo, G. Conte, S. Donatelli, G. Franceschinis, *Modelling with generalised stochastic Petri nets*, Wiley Series in Parallel Computing, John Wiley and Sons, 1995. Zbl 0843.68080

[30] Ph.M. Merlin, D.J. Farber, *Recoverability of communication protocols: implications of a theoretical study*, IEEE Transactions on Communications, **24**:9 (1976), 1036–1043. Zbl 0362.68096

[31] R.A.J. Milner, *Communication and concurrency*, Prentice-Hall, Upper Saddle River, NJ, USA, 1989. Zbl 0683.68008

[32] M.K. Molloy, *On the integration of the throughput and delay measures in distributed processing models*, Ph.D. thesis, Report, **CSD-810-921** (1981), University of California, Los Angeles, CA, USA.

[33] M.K. Molloy, *Discrete time stochastic Petri nets*, IEEE Transactions on Software Engineering, **11**:4 (1985), 417–423. Zbl 0558.68053

[34] A. Niaouris, *An algebra of Petri nets with arc-based time restrictions*, Lecture Notes in Computer Science, **3407** (2005), 447–462. Zbl 1109.68076

[35] A. Niaouris, M. Koutny, *An algebra of timed-arc Petri nets*, Technical Report, **CS-TR-895** (2005), School of Computer Science, Univers. of Newcastle upon Tyne, UK.

[36] Ch. Ramchandani, *Performance evaluation of asynchronous concurrent systems by timed Petri nets*, Ph.D. thesis, Department of Electrical Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, 1973.

[37] I.V. Tarasyuk, *Discrete time stochastic Petri box calculus*, Berichte aus dem Department für Informatik, **3/05**, Carl von Ossietzky Univers. Oldenburg, Germany, 2005.
[38] I.V. Tarasyuk, *Iteration in discrete time stochastic Petri box calculus*, Bulletin of the Novosibirsk Computing Center, Series Computer Science, IIS Special Issue, **24** (2006), 129–148. Zbl 1249.68132
[39] I.V. Tarasyuk, *Stochastic Petri box calculus with discrete time*, Fundamenta Informaticae, **76**:1–2 (2007), 189–218.
[40] I.V. Tarasyuk, *Equivalence relations for modular performance evaluation in dtsPBC*, Mathematical Structures in Computer Science, **24**:1 (2014), e240103.
[41] I.V. Tarasyuk, *Discrete time stochastic and deterministic Petri box calculus dtsdPBC*, Siberian Electronic Mathematical Reports, **17** (2020), 1598–1679. Zbl 1448.68352
[42] I.V. Tarasyuk, *Performance evaluation in stochastic process algebra dtsdPBC*, Siberian Electronic Mathematical Reports, **18**:2 (2021), 1105–1145. Zbl 1482.68156
[43] I.V. Tarasyuk, *Performance preserving equivalence for stochastic process algebra dtsdPBC*, Siberian Electronic Mathematical Reports, **20**:2 (2023), 646–699.
[44] I.V. Tarasyuk, H. Macià, V. Valero, *Discrete time stochastic Petri box calculus with immediate multiactions*, Technical Report, **DIAB-10-03-1**, Department of Computer Systems, High School of Computer Science Engineering, University of Castilla - La Mancha, Albacete, Spain, 2010.
[45] I.V. Tarasyuk, H. Macià, V. Valero, *Discrete time stochastic Petri box calculus with immediate multiactions dtsiPBC*, Proc. 6^{th} Int. Workshop on Practical Applications of Stochastic Modelling (PASM) 2012 and 11^{th} Int. Workshop on Parallel and Distributed Methods in Verification (PDMC) 2012 (J. Bradley, K. Heljanko, W. Knottenbelt, N. Thomas, eds.), London, UK, Electronic Notes in Theoretical Computer Science, **296** (2013), 229–252.
[46] I.V. Tarasyuk, H. Macià, V. Valero, *Performance analysis of concurrent systems in algebra dtsiPBC*, Programming and Computer Software, **40**:5 (2014), 229–249. Zbl 1339.68033
[47] I.V. Tarasyuk, H. Macià, V. Valero, *Stochastic process reduction for performance evaluation in dtsiPBC*, Siberian Electronic Mathematical Reports, **12** (2015), 513–551. Zbl 1346.60118
[48] I.V. Tarasyuk, H. Macià, V. Valero, *Stochastic equivalence for performance analysis of concurrent systems in dtsiPBC*, Siberian Electronic Mathematical Reports, **15** (2018), 1743–1812. Zbl 1414.60062
[49] R. Zijal, *Discrete time deterministic and stochastic Petri nets*, Proc. Int. Workshop on Quality of Communication-Based Systems 1994, Technical University of Berlin, Germany, 123–136, Kluwer Academic Publishers, 1995. Zbl 0817.68111
[50] R. Zijal, *Analysis of discrete time deterministic and stochastic Petri nets*, Ph.D. thesis, Technical University of Berlin, Germany, 1997.
[51] R. Zijal, R. German, *A new approach to discrete time stochastic Petri nets*, Proc. 11^{th} Int. Conf. on Analysis and Optimization of Systems, Discrete Event Systems (DES) 1994 (G. Cohen, J.-P. Quadrat, eds.), Sophia-Antipolis, France, Lecture Notes in Control and Information Sciences, **199** (1994), 198–204.

Igor Valerievich Tarasyuk
A.P. Ershov Institute of Informatics Systems,
Siberian Branch of the Russian Academy of Sciences,
Acad. Lavrentiev pr. 6,
630090 Novosibirsk, Russian Federation
E-mail address: itar@iis.nsk.su