Stochastic equivalence for performance evaluation of concurrent systems in dtsiPBC

Igor V. Tarasyuk[a,1,2,*], Hermenegilda Macià[b,1], Valentín Valero[b,1]

[a] *A.P. Ershov Institute of Informatics Systems, Siberian Branch of the Russian Academy of Sciences, 6, Acad. Lavrentiev pr., 630090 Novosibirsk, Russian Federation*

[b] *High School of Computer Science Engineering, University of Castilla - La Mancha, Avda. de España s/n, 02071 Albacete, Spain*

Abstract

We propose an extension with immediate multiactions of discrete time stochastic Petri box calculus (dtsPBC), presented by I.V. Tarasyuk. The resulting algebra dtsiPBC is a discrete time analogue of stochastic Petri box calculus (sPBC) with immediate multiactions, proposed by H. Macià, V. Valero and others within a continuous time domain. In this version of dtsiPBC, we use positive reals (instead of the previously used positive integers) as the weights of immediate multiactions to provide more flexibility in specification. The step operational semantics is constructed via labeled probabilistic transition systems. The denotational semantics is defined on the basis of a subclass of labeled discrete time stochastic Petri nets with immediate transitions. The consistency of the both semantics is demonstrated. In order to evaluate performance, the corresponding semi-Markov chains and (reduced) discrete time Markov chains are analyzed. We define step stochastic bisimulation equivalence of expressions and prove that it can be applied to reduce their transition systems and underlying semi-Markov chains while preserving the functionality and performance characteristics. We explain how this equivalence may help to simplify performance analysis of the algebraic processes. In a case study, a method of modeling, performance evaluation and behaviour preserving reduction of concurrent systems is outlined and applied to the shared memory system. We also determine the main advantages of dtsiPBC by comparing it with other well-known or similar SPAs.

Keywords: stochastic process algebra, Petri box calculus, discrete time, immediate multiaction, performance evaluation, stochastic equivalence
2000 MSC: 60J10, 60J20, 60K15, 68Q55, 68Q60, 68Q85

1. Introduction

Algebraic process calculi like CSP [53], ACP [8] and CCS [77] are a well-known formal model for the specification of computing systems and analysis of their behaviour. In such process algebras (PAs), systems and processes are specified by formulas, and verification of their properties is accomplished at a syntactic level via equivalences, axioms and inference rules. In the last decades, stochastic extensions of PAs were proposed, such as MTIPP [50], PEPA [52] and EMPA [14, 13, 9]. Stochastic process algebras (SPAs) do not just specify actions which can occur as usual process algebras (qualitative features), but they associate some quantitative parameters with actions (quantitative characteristics).

*Corresponding author. Tel.: +7 3833306360; fax: +7 3833323494.

Email addresses: itar@iis.nsk.su (Igor V. Tarasyuk), Hermenegilda.Macia@uclm.es (Hermenegilda Macià), Valentin.Valero@uclm.es (Valentín Valero)

URL: http://itar.iis.nsk.su (Igor V. Tarasyuk), http://www.dsi.uclm.es/personal/ValentinValero (Valentín Valero)

[1] Partially supported by the Spanish Ministry of Science and Innovation and the European Union FEDER Funds with the coordinated Project DArDOS entitled "Formal development and analysis of complex systems in distributed contexts: foundations, tools and applications", UCLM subproject "Formal analysis and applications of Web services and electronic contracts", under Grant TIN2015-65845-C3-2-R.

[2] Partially supported by Deutsche Forschungsgemeinschaft (DFG) under grant BE 1267/14-1.

1.1. Petri box calculus

PAs specify concurrent systems in a compositional way via an expressive formal syntax. On the other hand, Petri nets (PNs) provide a graphical representation of such systems and capture explicit asynchrony in their behaviour. To combine advantages of both models, a semantics of algebraic formulas in terms of PNs has been defined. Petri box calculus (PBC) [15, 17, 16] is a flexible and expressive process algebra developed as a tool for specification of the 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 not from single (visible or invisible) actions and variables, like in CCS, but from multisets of elementary actions and their conjugates, called multiactions (*basic formulas*). The empty multiset of actions is interpreted as the silent multiaction specifying some invisible activity. In contrast to CCS, synchronization is separated from parallelism (*concurrent constructs*). Synchronization is a unary multi-way stepwise operation based on communication of actions and their conjugates. This extends the CCS approach with conjugate matching labels. Synchronization in PBC is asynchronous, unlike that in Synchronous CCS (SCCS) [77]. Other operations are sequence and choice (*sequential constructs*). The calculus includes also restriction and relabeling (*abstraction constructs*). To specify infinite processes, refinement, recursion and iteration operations were added (*hierarchical constructs*). Thus, unlike CCS, PBC has an additional iteration construction to specify infinite behaviour when the semantic interpretation in finite PNs is possible. PBC has a step operational semantics in terms of labeled transition systems, based on the rules of structural operational semantics (SOS) [86]. 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). Note that we do not reason in terms of a big-step (natural) [57] or small-step (structural) [86] operational semantics here, and that PBC (and all its extensions to be mentioned further) have a small-step operational semantics, in that terminology. A denotational semantics of PBC was proposed via a subclass of PNs equipped with an interface and considered up to isomorphism, called Petri boxes. For more detailed comparison of PBC with other process algebras and the reasoning about importance of non-interleaving semantics see [15, 16]. In the last years, several extensions of PBC with a deterministic, a nondeterministic or a stochastic model of time were presented.

1.2. Time extensions of Petri box calculus

To specify systems with time constraints, such as real time systems, deterministic (fixed) or nondeterministic (interval) time delays are used. A time extension of PBC with a nondeterministic time model, called time Petri box calculus (tPBC), was proposed in [61]. In tPBC, timing information is added by associating time intervals (the earliest and the latest firing time) 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 [76] and called time Petri boxes (ct-boxes).

Another time enrichment of PBC, called Timed Petri box calculus (TPBC), was defined in [73], it accommodates a deterministic model of time. In contrast to tPBC, multiactions of TPBC are not instantaneous, but have time durations. Additionally, in TPBC there exist no "illegal" multiaction occurrences, unlike tPBC. The complexity of "illegal" occurrences mechanism was one of the main intentions to construct TPBC though this calculus appeared to be more complicated than tPBC. 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 [89] and called Timed Petri boxes (T-boxes). Note that tPBC and TPBC differ in ways they capture time information, and they are not in competition but complement each other.

The third time extension of PBC, called arc time Petri box calculus (atPBC), was constructed in [84], 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), where time restrictions are associated with the arcs, called arc time Petri boxes (at-boxes). Further, all the calculi tPBC, TPBC and atPBC apply the discrete time approach, but only tPBC and atPBC have immediate (multi)actions.

1.3. Stochastic extensions of Petri box calculus

The set of states for the systems with deterministic or nondeterministic delays often differs drastically from that for the timeless systems, hence, the analysis results for untimed systems may be not valid for the time ones. To

solve this problem, stochastic delays are considered, which are the random variables with a (discrete or continuous) probability distribution. If the random variables governing delays have an infinite support then the corresponding SPA can exhibit all the same behaviour as its underlying untimed PA. A stochastic extension of PBC, called stochastic Petri box calculus (sPBC), was proposed in [68]. 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. Just a finite part of PBC was initially used for the stochastic enrichment, i.e. in its former version sPBC has neither refinement nor recursion nor iteration operations. The calculus has an interleaving operational semantics defined via transition systems labeled with multiactions and their rates. Its denotational semantics was defined in terms of a subclass of labeled continuous time stochastic PNs (LCTSPNs), based on CTSPNs [74, 5] and called stochastic Petri boxes (s-boxes). In [65], the iteration operator was added to sPBC. In sPBC with iteration, performance of the processes is evaluated by analyzing their underlying continuous time Markov chains (CTMCs). In [66], a number of new equivalence relations were proposed for regular terms of sPBC with iteration to choose later a suitable candidate for a congruence. sPBC with iteration was enriched further with immediate multiactions having zero delay in [67]. 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 (LGSPNs), based on GSPNs [74, 5, 6] and called generalized stochastic Petri boxes (gs-boxes). The performance analysis in gsPBC is based on the underlying semi-Markov chains (SMCs).

PBC has a step operational semantics, whereas sPBC has an interleaving one. Remember that in step semantics, parallel executions of activities (steps) are permitted while in interleaving semantics, we can execute only single activities. Hence, a stochastic extension of PBC with a step semantics is needed to keep the concurrency degree of behavioural analysis at the same level as in PBC. As mentioned in [80, 81], in contrast to continuous time approach (used in sPBC), discrete time approach allows for constructing models of common clock systems and clocked devices. In such models, multiple transition firings (or executions of multiple activities) at time moments (ticks of the central clock) are possible, resulting in a step semantics. Moreover, employment of discrete stochastic time fills the gap between the models with deterministic (fixed) time delays and those with continuous stochastic time delays. As argued in [1], arbitrary delay distributions are much easier to handle in a discrete time domain. In [71, 72, 69], discrete stochastic time was preferred to enable simultaneous expiration of multiple delays. In [92, 94], a discrete time stochastic extension dtsPBC of finite PBC was presented. 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 [80, 81] and called discrete time stochastic Petri boxes (dts-boxes). A variety of stochastic equivalences were proposed to identify stochastic processes with similar behaviour which are differentiated by the semantic equivalence. The interrelations of all the introduced equivalences were studied. In [93, 95], we constructed an enrichment of dtsPBC with the iteration operator used to specify infinite processes. The performance evaluation in dtsPBC with iteration is accomplished via the underlying discrete time Markov chains (DTMCs) of the algebraic processes. Since dtsPBC has a discrete time semantics and geometrically distributed sojourn time in the process states, unlike sPBC with continuous time semantics and exponentially distributed delays, the calculi apply two different approaches to the stochastic extension of PBC, in spite of some similarity of their syntax and semantics inherited from PBC. The main advantage of dtsPBC is that concurrency is treated like in PBC having step semantics, whereas in sPBC parallelism is simulated by interleaving, obliging one to collect the information on causal independence of activities before constructing the semantics. In [96, 97], we presented the extension dtsiPBC of the latter calculus with immediate multiactions. 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 not relevant for performance evaluation. Thus, immediate multiactions can be considered as a kind of instantaneous dynamic state adjustment and, in many cases, they result in a simpler and more clear system representation.

1.4. Equivalence relations

A notion of equivalence is important in theory of computing systems. Equivalences are applied both to compare behaviour of systems and reduce their structure. There is a wide diversity of behavioural equivalences, and their

interrelations are well explored in the literature. The best-known and widely used one is bisimulation. Typically, the mentioned equivalences take into account only functional (qualitative) but not performance (quantitative) aspects. Additionally, the equivalences are usually interleaving ones, i.e. they interpret concurrency as sequential nondeterminism. Interleaving equivalences permit to imitate parallel execution of actions via all possible occurrence sequences (interleavings) of them. Step equivalences require instead simulating such a parallel execution by simultaneous occurrence (step) of all the involved actions. To respect quantitative features of behaviour, probabilistic equivalences have additional requirement on execution probabilities. Two equivalent processes must be able to execute the same sequences of actions, and for every such sequence, its execution probabilities within both processes should coincide. In case of probabilistic bisimulation equivalence, the states from which similar future behaviours start are grouped into equivalence classes that form elements of the aggregated state space. From every two bisimilar states, the same actions can be executed, and the subsequent states resulting from execution of an action belong to the same equivalence class. In addition, for both states, the cumulative probabilities to move to the same equivalence class by executing the same action coincide. A different kind of quantitative relations is called Markovian equivalences, which take rate (the parameter of exponential distribution that governs time delays) instead of probability. Note that the probabilistic equivalences can be seen as discrete time analogues of the Markovian ones, since the latter are defined as the continuous time relations.

Interleaving probabilistic weak trace equivalence was introduced in [32] on labeled probabilistic transition systems. Interleaving probabilistic strong bisimulation equivalence was proposed in [64] on the same model. Interleaving probabilistic equivalences were defined for probabilistic processes in [56, 42]. Interleaving Markovian strong bisimulation equivalence was constructed in [50] for MTIPP, in [52] for PEPA and in [14, 13, 9] for EMPA. Some variants of interleaving Markovian weak bisimulation equivalence were considered in [27] on Markovian process algebras, in [28] on labeled CTSPNs and in [29] on labeled GSPNs. In [10, 11], a comparison of interleaving Markovian trace, test, strong and weak bisimulation equivalences was carried out on sequential and concurrent Markovian process calculi. Nevertheless, no appropriate equivalence notion was defined for concurrent SPAs. The non-interleaving bisimulation equivalence in GSMPA [21, 20] uses ST-semantics for action particles while in Sπ [88] it is based on a sophisticated labeling.

1.5. Our contributions

In this paper, we present dtsPBC with iteration extended with immediate multiactions, called *discrete time stochastic and immediate Petri box calculus* (dtsiPBC), which is a discrete time analog of sPBC. The latter calculus has iteration and immediate multiactions within the context of a continuous time domain. In the current version of dtsiPBC, we use positive reals (instead of positive integers, used in the previous versions) as the weights of immediate multiactions, to allow for more flexibile and convenient specification of systems. The step operational semantics is constructed with the use of labeled probabilistic transition systems. The denotational semantics is defined in terms of a subclass of labeled discrete time stochastic and immediate PNs (LDTSPNs with immediate transitions, LDTSIPNs), based on the extension of DTSPNs with transition labeling and immediate transitions, called dtsi-boxes. LDTSIPNs possess some features of discrete time deterministic and stochastic PNs (DTDSPNs) [105] and discrete deterministic and stochastic PNs (DDSPNs) [104], but in LDTSIPNs simultaneous transition firings are possible while in DTDSPNs and DDSPNs only firings of single transitions are allowed. The consistency of both semantics is demonstrated. The corresponding stochastic process, the underlying SMC, is constructed and investigated, with the purpose of performance evaluation, which is the same for both semantics. In addition, the alternative solution methods are developed, based on the underlying DTMC and its reduction (RDTMC) by eliminating vanishing states. Further, we propose step stochastic bisimulation equivalence allowing one to identify algebraic processes with similar behaviour that are however differentiated by the semantics of the calculus. We examine the interrelations of the proposed notion with other equivalences of the algebra. We describe how step stochastic bisimulation equivalence can be used to reduce transition systems of expressions and their underlying SMCs while preserving the qualitative and quantitative characteristics. We prove that the mentioned equivalence guarantees identity of the stationary behaviour and residence time properties in the equivalence classes. This implies coincidence of performance indices based on steady-state probabilities of the modeled stochastic systems. The equivalences possessing the property can be used to reduce the state space of a system and thus simplify its performance evaluation, what is usually a complex problem due to the state space explosion. We present a case study of a system with two processors and a common shared memory explaining how to model concurrent systems within the calculus and analyze their performance, as well as in which way to reduce

the systems while preserving their performance indices and making simpler the performance evaluation. Finally, we consider differences and similarities between dtsiPBC and other SPAs to determine the advantages of our calculus. We discuss the SPAs approaches to the analytical solution, concurrency interpretation and application area.

The first results on this subject can be found in [96]. Concerning differences from our previous journal papers about dtsiPBC [97, 98, 99], the present text is much more detailed and many new results have been added. In particular, all the used notions (such as numbering, functions collecting executable activities, probability functions) are formally defined and completely explained with examples; the operational and denotational semantics are given in full detail (the inaction, action rules, LDTSPNs and dtsi-boxes are extensively described and discussed); compact illustrative examples (of standard and alternative solution methods) are presented; keeping properties of original Markov chains (irreducibility, positive recurrence and aperiodicity) in their embedded and state-aggregated versions is studied. The main new contribution of the paper, step stochastic bisimulation equivalence of the process expressions, is introduced and checked for stationary behaviour and sojourn time features preservation in the equivalence classes; quotienting the transition systems, SMCs and DTMCs by the equivalence, as well as the resulting simplification of performance evaluation, are considered. As an application example, the standard and generalized variants of the shared memory system, quotients of their behaviour (represented by the transition systems, SMCs and DTMCs) by the equivalence and reductions of the quotients by removing vanishing states are constructed; the generalized probabilities of the reduced quotient DTMC are treated as parameters to be adjusted for performance optimization. In the enhanced related work overview, strong points of dtsiPBC with respect to other SPAs are detected; in the extensive discussion, analytical solution, concurrency interpretation, application area and general advantages of dtsiPBC are explained.

If we compare dtsiPBC with the classical SPAs MTIPP, PEPA and EMPA, the first main difference between them comes from PBC, since dtsiPBC is based on this calculus: all algebraic operations and a notion of multiaction are inherited from PBC. The second main difference is discrete probabilities of activities induced by the discrete time approach, whereas action rates are used in the standard SPAs with continuous time. As a consequence, dtsiPBC has a non-interleaving step operational semantics. This is in contrast to the classical SPAs, where concurrency is modeled by interleaving because of the continuous probability distributions of action delays and the race condition applied when several actions can be executed in a state. The third main difference is immediate multiactions. There are no instantaneous activities in MTIPP and PEPA while the immediate actions in EMPA can have different priority levels. All immediate multiactions in dtsiPBC have the same priority level, with the intention to simplify the specification and analysis, since weights (assigned also to immediate actions in EMPA) are enough to denote preferences among immediate multiactions and to produce the conformable probabilistic behaviours. The salient point of dtsiPBC is a combination of immediate multiactions, discrete stochastic time and step semantics in an SPA.

Thus, the main contributions of the paper are the following.

- Powerful and expressive discrete time SPA with immediate activities called dtsiPBC in its final form.
- Step operational semantics of dtsiPBC in terms of labeled probabilistic transition systems.
- Petri net denotational semantics of dtsiPBC based on discrete time stochastic and immediate Petri nets.
- Performance analysis via underlying semi-Markov chains and (reduced) discrete time Markov chains.
- Stochastic equivalence used for behaviour-preserving reduction that simplifies the performance evaluation.
- Extended case study illustrating how to apply the obtained theoretical results in practice.

1.6. Structure of the paper

The paper is organized as follows. In Section 2, the syntax of the extended calculus dtsiPBC is presented. In Section 3, we construct the operational semantics of the algebra in terms of labeled probabilistic transition systems. In Section 4, we propose the denotational semantics based on a subclass of LDTSIPNs. In Section 5, the corresponding stochastic process is defined and analyzed. Step stochastic bisimulation equivalence is defined and investigated in Section 6. In Section 7, we explain how to reduce transition systems and underlying SMCs of process expressions modulo the equivalence. In Section 8, the introduced equivalence is applied to the stationary behaviour comparison to verify the performance preservation. In Section 9, a shared memory system is presented as a case study. The difference between dtsiPBC and other well-known or similar SPAs is considered in Section 10. The advantages of dtsiPBC are

explained in Section 11. Finally, Section 12 summarizes the results obtained and outlines the research perspectives. The long and complex proofs are moved to Appendix A.

2. Syntax

In this section, we propose the syntax of dtsiPBC. First, we recall a definition of multiset that is an extension of the set notion by allowing several identical elements.

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

We denote the *set of all finite multisets* over a set X by $\boldsymbol{N}^X_{fin}$. Let $M, M' \in \boldsymbol{N}^X_{fin}$. The *cardinality* of M is defined as $|M| = \sum_{x \in X} M(x)$. We write $x \in M$ if $M(x) > 0$ and $M \subseteq M'$ if $\forall x \in X,\ M(x) \leq M'(x)$. We define $(M + M')(x) = M(x) + M'(x)$ and $(M - M')(x) = \max\{0, M(x) - M'(x)\}$. When $\forall x \in X,\ M(x) \leq 1,\ M$ can be interpreted as a proper set and denoted by $M \subseteq X$. The *set of all subsets (powerset)* of X is denoted by 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)* such that $\hat{a} \neq a$ and $\hat{\hat{a}} = a$. Let $\mathcal{A} = Act \cup \widehat{Act}$ be the set of *all actions*, and $\mathcal{L} = \boldsymbol{N}^{\mathcal{A}}_{fin}$ be the set of *all multiactions*. Note that $\emptyset \in \mathcal{L}$, this corresponds to an internal move, i.e. the execution of a multiaction that contains no visible action names. The *alphabet* of $\alpha \in \mathcal{L}$ is defined as $\mathcal{A}(\alpha) = \{x \in \mathcal{A} \mid \alpha(x) > 0\}$.

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 probabilities of stochastic multiactions are required not to be equal to 1 to avoid extra model complexity, since in this case one should assign with them weights, needed to make a choice when several stochastic multiactions with probability 1 can be executed from a state. The difficulty is that when the stochastic multiactions with probability 1 occur in a step (parallel execution), all other with the less probabilities do not. In this case, some problems appear with conflicts resolving. See [80, 81] for the discussion on SPNs. This decision also allows us to avoid technical difficulties related to conditioning events with probability 0. Another reason is that not allowing probability 1 for stochastic multiactions excludes a potential source of periodicity (hence, non-ergodicity) in the underlying SMCs of the algebraic expressions. On the other hand, there is no sense to allow zero probabilities of stochastic multiactions, since they would never be performed in this case. Let $\mathcal{SL}$ be the set of *all stochastic multiactions*.

An *immediate multiaction* is a pair $(\alpha, \natural_l)$, where $\alpha \in \mathcal{L}$ and is the $l \in \boldsymbol{R}_{>0} = (0; +\infty)$ 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 immediate multiaction at the current discrete time moment. Such weights are used to calculate the probabilities to execute sets of immediate multiactions instantly. Immediate multiactions have a priority over stochastic ones. One can assume that all immediate multiactions have priority 1, whereas all stochastic ones have priority 0. This means that in a state where both kinds of multiactions can occur, immediate multiactions always occur before stochastic ones. Stochastic and immediate multiactions cannot participate together in some step (concurrent execution), i.e. the steps consisting only of immediate multiactions or those including only stochastic multiactions are allowed. Let $\mathcal{IL}$ be the set of *all immediate multiactions*.

Note that the same multiaction $\alpha \in \mathcal{L}$ may have different probabilities and weights in the same specification. An *activity* is a stochastic or an immediate multiaction. Let $\mathcal{SIL} = \mathcal{SL} \cup \mathcal{IL}$ be the set of *all activities*. The *alphabet* of a multiset of activities $\Upsilon \in \boldsymbol{N}^{\mathcal{SIL}}_{fin}$ is defined as $\mathcal{A}(\Upsilon) = \cup_{(\alpha,\kappa) \in \Upsilon} \mathcal{A}(\alpha)$. For an activity $(\alpha, \kappa) \in \mathcal{SIL}$, we define its *multiaction part* as $\mathcal{L}(\alpha, \kappa) = \alpha$ and its *probability* or *weight part* as $\Omega(\alpha, \kappa) = \kappa$ if $\kappa \in (0; 1)$; or $\Omega(\alpha, \kappa) = l$ if $\kappa = \natural_l,\ l \in \boldsymbol{R}_{>0}$. The *multiaction part* of a multiset of activities $\Upsilon \in \boldsymbol{N}^{\mathcal{SIL}}_{fin}$ is defined as $\mathcal{L}(\Upsilon) = \sum_{(\alpha,\kappa) \in \Upsilon} \alpha$.

Activities are combined into formulas (process expressions) by the following operations: *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 (choice composition) have a standard interpretation, like in other process algebras, but parallelism (parallel composition) does not include synchronization, unlike the corresponding operation in CCS [77].

Relabeling functions $f : \mathcal{A} \to \mathcal{A}$ are bijections preserving conjugates, i.e. $\forall x \in \mathcal{A},\ f(\hat{x}) = \widehat{f(x)}$. Relabeling is extended to multiactions in the usual way: for $\alpha \in \mathcal{L}$, we define $f(\alpha) = \sum_{x \in \alpha} f(x)$. Relabeling is extended to the multisets of activities as follows: for $\Upsilon \in \boldsymbol{N}_{fin}^{\mathcal{SIL}}$, 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 $\alpha \oplus_a \beta = \gamma$, where

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

In other words, we require that $\alpha \oplus_a \beta = \alpha + \beta - \{a, \hat{a}\}$, i.e. we remove one exemplar of a and one exemplar of $\hat{a}$ from the multiset sum $\alpha + \beta$, since the synchronization of a and $\hat{a}$ produces $\emptyset$. Activities are synchronized with the use of their multiaction parts, i.e. the synchronization by a of two activities, whose multiaction parts α and β possess the properties mentioned above, 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 immediate ones, since immediate multiactions have a priority over stochastic ones, hence, stochastic and immediate multiactions cannot be executed together (note also that the execution of immediate multiactions takes no time, unlike that of 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 process 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 subprocess is executed.

Static expressions specify the structure of processes. As we shall see, the expressions correspond to unmarked LDTSIPNs (note that LDTSIPNs are marked by definition).

Definition 2.2. Let $(\alpha, \kappa) \in \mathcal{SIL}$ and $a \in Act$. A *static expression* of dtsiPBC is defined as

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

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

To make the grammar above unambiguous, one can add parentheses in the productions with binary operations: $(E; E)$, $(E[]E)$, $(E \| E)$. However, we prefer the PBC approach and add them to resolve ambiguities only.

To avoid technical difficulties with the iteration operator, we should not allow any concurrency at the highest level of the second argument of iteration. This is not a severe restriction though, since we can always prefix parallel expressions by an activity with the empty multiaction part. Later on, in Example 4.2, we shall demonstrate that relaxing the restriction can result in nets which are not safe. Alternatively, we can use a different, safe, version of the iteration operator, but its net translation has six arguments. See also [16] for discussion on this subject.

Definition 2.3. Let $(\alpha, \kappa) \in \mathcal{SIL}$ and $a \in Act$. A *regular static expression* of dtsiPBC is defined as

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

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

Dynamic expressions specify the states of processes. As we shall see, the expressions correspond to LDTSIPNs (which are marked by default). 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. The *underlying static expression* of a dynamic one is obtained by removing all upper and lower bars from it.

Definition 2.4. Let $E \in StatExpr$ and $a \in Act$. A *dynamic expression* of dtsiPBC is defined as

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

Let *DynExpr* denote the set of *all dynamic expressions* of dtsiPBC.

Note that if the underlying static expression of a dynamic one is not regular, the corresponding LDTSIPN can be non-safe (though, it is 2-bounded in the worst case [16]).

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

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

3. Operational semantics

In this section, we define the step operational semantics in terms of labeled 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 these syntactic transformations is to obtain the well-structured resulting expressions called operative ones to which no inaction rules can be further applied. As we shall see, the application of an inaction rule to a dynamic expression does not lead to any discrete time tick or any transition firing in the corresponding LDTSIPN, hence, its current marking remains unchanged.

Thus, an application of every inaction rule does not require any discrete time delay, i.e. the dynamic expression transformation described by the rule is accomplished instantly.

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

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

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

Second, in Table 2, we introduce inaction rules for regular dynamic expressions in the arbitrary form. In this table, $E, F \in RegStatExpr$, $G, H, \widetilde{G}, \widetilde{H} \in RegDynExpr$ and $a \in Act$. By reason of brevity, two distinct inaction rules with the same premises are collated in some cases, resulting in the inaction rules with double conclusion.

Table 2: Inaction rules for arbitrary regular dynamic expressions.

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

Definition 3.1. 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 dtsiPBC.

Note that any dynamic expression can be always transformed into a (not necessarily unique) operative one by using the inaction rules. In the following, we consider regular expressions only and omit the word "regular".

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

3.2. Action and empty loop rules

The action rules are applied when some activities are executed. With these rules we capture the prioritization of immediate multiactions with respect to stochastic ones. We also have the empty loop rule which is used to capture a delay of one discrete time unit in the same state when no immediate multiactions are executable. In this case, the empty multiset of activities is executed. The action and empty loop 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 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 (or immediate, otherwise) multiactions describe dynamic expression transformations in the form of $G \stackrel{\Gamma}{\rightarrow} \widetilde{G}$ (or $G \stackrel{I}{\rightarrow} \widetilde{G}$) due to execution of non-empty multisets Γ of stochastic (or I of immediate) multiactions. The rules represent possible state changes of the specified processes when some non-empty multisets of stochastic (or immediate) multiactions are executed. As we shall see, the application of an action rule with stochastic (or immediate) multiactions to a dynamic expression leads in the corresponding LDTSIPN to a discrete time tick at which some stochastic transitions fire (or to the instantaneous firing of some immediate transitions) and possible change of the current marking. The current marking remains unchanged only if there is a self-loop produced by the iterative execution of a non-empty multiset, which must be one-element, i.e. the single stochastic (or immediate) multiaction. The reason is the regularity requirement that allows no concurrency at the highest level of the second argument of iteration.

The empty loop rule (applicable only when no immediate multiactions can be executed from the current state) describes dynamic expression transformations in the form of $G \stackrel{\emptyset}{\rightarrow} G$ due to execution of the empty multiset of activities at a discrete time tick. The rule reflects a non-zero probability to stay in the current state at the next time moment, which is an essential feature of discrete time stochastic processes. As we shall see, the application of the empty loop rule to a dynamic expression leads to a discrete time tick in the corresponding LDTSIPN at which no transitions fire and the current marking is not changed. This is a new rule that has no prototype among inaction rules of PBC, since it represents a time delay, but no notion of time exists in PBC. The PBC rule $G \stackrel{\emptyset}{\rightarrow} G$ from [17, 16] in our setting would correspond to the rule $G \Rightarrow G$ that describes staying in the current state when no time elapses. Since we do not need the latter rule to transform dynamic expressions into operative ones and it can even destroy the definition of operative expressions, we do not introduce it in dtsiPBC.

Thus, an application of every action rule with stochastic multiactions or the empty loop rule requires one discrete time unit delay, i.e. the execution of a (possibly empty) multiset of stochastic 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 of time.

Note that expressions of dtsiPBC can contain identical activities. To avoid technical difficulties, such as the proper calculation of the state change probabilities for multiple transitions, we can always enumerate coinciding activities from left to right in the syntax of expressions. 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 3.3. The *numbering* of expressions is defined as $\iota ::= n \mid (\iota)(\iota)$, where $n \in \boldsymbol{N}$.

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

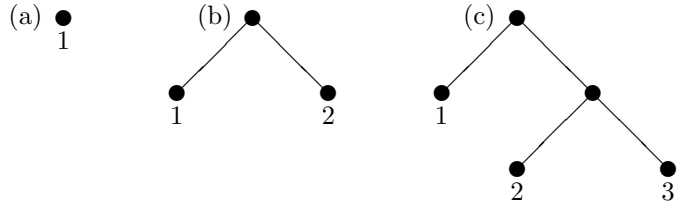

Figure 1: The binary trees encoded with the numberings 1, (1)(2) and (1)((2)(3)).

Example 3.1. *The numbering* 1 *encodes the binary tree depicted in Figure 1(a) with the root labeled by* 1*. The numbering* (1)(2) *corresponds to the binary tree depicted in Figure 1(b) without internal nodes and with two leaves labeled by* 1 *and* 2*. The numbering* (1)((2)(3)) *represents the binary tree depicted in Figure 1(c) with one internal node, which is the root for the subtree* (2)(3)*, and three leaves labeled by* 1, 2 *and* 3*.*

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 be able to identify the mentioned instances.

The *content* of a numbering $\iota \in Num$ is

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

After the enumeration, the multisets of activities from the expressions will become the proper sets. Suppose that the identical activities are enumerated when needed to avoid ambiguity. This enumeration is considered to be implicit.

Let X be some set. We denote the Cartesian product $X \times X$ by X^2. Let $\mathcal{E} \subseteq X^2$ be an equivalence relation on X. Then the *equivalence class* (with respect to $\mathcal{E}$) of an element $x \in X$ is defined by $[x]_{\mathcal{E}} = \{y \in X \mid (x, y) \in \mathcal{E}\}$. The equivalence $\mathcal{E}$ partitions X into the *set of equivalence classes* $X/_{\mathcal{E}} = \{[x]_{\mathcal{E}} \mid x \in X\}$.

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. G is an *initial* dynamic expression, denoted by $init(G)$, if $\exists E \in RegStatExpr,\ G \in [\overline{E}]_{\approx}$. G is a *final* dynamic expression, denoted by $final(G)$, if $\exists E \in RegStatExpr,\ G \in [\underline{E}]_{\approx}$.

Definition 3.4. Let $G \in OpRegDynExpr$. We define the *set of all non-empty sets of activities which can be potentially executed from* G, denoted by $Can(G)$. Let $(\alpha, \kappa) \in \mathcal{SIL},\ E, F \in RegStatExpr,\ H \in OpRegDynExpr$ and $a \in Act$.

1. If $final(G)$ then $Can(G) = \emptyset$.
2. If $G = \overline{(\alpha, \kappa)}$ then $Can(G) = \{\{(\alpha, \kappa)\}\}$.
3. 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 $a, \hat{a} \notin \mathcal{A}(\Upsilon)$), $\Upsilon \in Can(G\ \mathsf{sy}\ a),\ \Upsilon \in Can([G * E * F])$, $\Upsilon \in Can([E * G * F]),\ \Upsilon \in Can([E * F * G])$.
4. If $\Upsilon \in Can(G)$ and $\Xi \in Can(H)$ then $\Upsilon + \Xi \in Can(G \| H)$.
5. If $\Upsilon \in Can(G\ \mathsf{sy}\ a)$ and $(\alpha, \kappa), (\beta, \lambda) \in \Upsilon$ are different activities such that $a \in \alpha,\ \hat{a} \in \beta$, 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) $\Upsilon - \{(\alpha, \kappa), (\beta, \lambda)\} + \{(\alpha \oplus_a \beta, \natural_{l+m})\} \in Can(G\ \mathsf{sy}\ a)$ if $\kappa = \natural_l,\ \lambda = \natural_m,\ l, m \in \boldsymbol{R}_{>0}$.

   When we synchronize the same set of activities in different orders, we obtain several activities with the same multiaction and probability or weight parts, but with different numberings having the same content. Then we only consider a single one of the resulting activities to avoid introducing redundant ones.

   For example, the synchronization of stochastic multiactions $(\alpha, \rho)_1$ and $(\beta, \chi)_2$ in different orders generates the activities $(\alpha \oplus_a \beta, \rho \cdot \chi)_{(1)(2)}$ and $(\beta \oplus_a \alpha, \chi \cdot \rho)_{(2)(1)}$. Similarly, the synchronization of immediate multiactions $(\alpha, \natural_l)_1$ and $(\beta, \natural_m)_2$ in different orders generates the activities $(\alpha \oplus_a \beta, \natural_{l+m})_{(1)(2)}$ and $(\beta \oplus_a \alpha, \natural_{m+l})_{(2)(1)}$. Since $Cont((1)(2)) = \{1, 2\} = Cont((2)(1))$, in both cases, only the first activity (or, symmetrically, the second one) resulting from synchronization will appear in a set from $Can(G\ \mathsf{sy}\ a)$.

Note that if $\Upsilon \in Can(G)$ then by definition of $Can(G)$, $\forall \Xi \subseteq \Upsilon,\ \Xi \neq \emptyset$, we have $\Xi \in Can(G)$.

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

Definition 3.5. Let $G \in OpRegDynExpr$. The *set of all non-empty sets of activities which can be executed from G* is

$$Now(G) = \begin{cases} Can(G), & (Can(G) \subseteq \boldsymbol{N}^{\mathcal{SL}}_{fin} \setminus \{\emptyset\}) \vee (Can(G) \subseteq \boldsymbol{N}^{\mathcal{IL}}_{fin} \setminus \{\emptyset\}); \\ Can(G) \cap \boldsymbol{N}^{\mathcal{IL}}_{fin}, & \text{otherwise.} \end{cases}$$

An operative dynamic expression expression $G \in OpRegDynExpr$ is *tangible*, denoted by $tang(G)$, if $Now(G) \subseteq \boldsymbol{N}^{\mathcal{SL}}_{fin} \setminus \{\emptyset\}$. In particular, we have $tang(G)$, if $Now(G) = \emptyset$. Otherwise, the expression G is *vanishing*, denoted by $vanish(G)$, and in this case $\emptyset \neq Now(G) \subseteq \boldsymbol{N}^{\mathcal{IL}}_{fin} \setminus \{\emptyset\}$. Note that the operative dynamic expressions from $[G]_{\approx}$ may have different types in general. The following example demonstrates two operative dynamic expressions H and H' with $H \approx H'$, such that $vanish(H)$, but $tang(H')$.

Example 3.2. *Let* $G = (\overline{(\{a\}, \natural_1)}[](\{b\}, \natural_2))\|\overline{(\{c\}, \frac{1}{2})}$ *and* $G' = ((\{a\}, \natural_1)[]\overline{(\{b\}, \natural_2)})\|\overline{(\{c\}, \frac{1}{2})}$. *Then* $G \approx G'$, *since* $G \Leftarrow G'' \Rightarrow G'$ *for* $G'' = \overline{((\{a\}, \natural_1)[](\{b\}, \natural_2))\|(\{c\}, \frac{1}{2})}$, *but* $Can(G) = \{\{(\{a\}, \natural_1)\}, \{(\{c\}, \frac{1}{2})\}, \{(\{a\}, \natural_1), (\{c\}, \frac{1}{2})\}\}$, $Can(G') = \{\{(\{b\}, \natural_2)\}, \{(\{c\}, \frac{1}{2})\}, \{(\{b\}, \natural_2), (\{c\}, \frac{1}{2})\}\}$ *and* $Now(G) = \{\{(\{a\}, \natural_1)\}\}$, $Now(G') = \{\{(\{b\}, \natural_2)\}\}$. *Clearly, we have* $vanish(G)$ *and* $vanish(G')$. *The executions like that of* $\{(\{c\}, \frac{1}{2})\}$ *(and all sets including it) from* G *and* G' *must be disabled using preconditions in the action rules, since immediate multiactions have a priority over stochastic ones, hence, the former are always executed first.*

Let $H = \overline{(\{a\}, \natural_1)}[](\{b\}, \frac{1}{2})$ *and* $H' = (\{a\}, \natural_1)[]\overline{(\{b\}, \frac{1}{2})}$. *Then* $H \approx H'$, *since* $H \Leftarrow H'' \Rightarrow H'$ *for* $H'' = \overline{(\{a\}, \natural_1)[](\{b\}, \frac{1}{2})}$, *but* $Can(H) = Now(H) = \{\{(\{a\}, \natural_1)\}\}$ *and* $Can(H') = Now(H') = \{\{(\{b\}, \frac{1}{2})\}\}$. *We have* $vanish(H)$, *but* $tang(H')$. *To make the action rules correct under structural equivalence, the executions like that of* $\{(\{b\}, \frac{1}{2})\}$ *from* H' *must be disabled using preconditions in the action rules, since immediate multiactions have a priority over stochastic ones, hence, the choices between them are always resolved in favour of the former.*

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

Now, in Table 3, we define the action and empty loop rules. In this table, $(\alpha, \rho), (\beta, \chi) \in \mathcal{SL}$, $(\alpha, \natural_l), (\beta, \natural_m) \in \mathcal{IL}$ and $(\alpha, \kappa) \in \mathcal{SIL}$. Further, $E, F \in RegStatExpr$, $G, H \in OpRegDynExpr$, $\widetilde{G}, \widetilde{H} \in RegDynExpr$ and $a \in Act$. Moreover, $\Gamma, \Delta \in \boldsymbol{N}^{\mathcal{SL}}_{fin} \setminus \{\emptyset\}$, $\Gamma' \in \boldsymbol{N}^{\mathcal{SL}}_{fin}$, $I, J \in \boldsymbol{N}^{\mathcal{IL}}_{fin} \setminus \{\emptyset\}$, $I' \in \boldsymbol{N}^{\mathcal{IL}}_{fin}$ and $\Upsilon \in \boldsymbol{N}^{\mathcal{SIL}}_{fin} \setminus \{\emptyset\}$.

We use the following abbreviations in the names of the rules from the table: "**El**" for "**E**mpty **l**oop", "**B**" for "**B**asis case", "**S**" for "**S**equence", "**C**" for "**C**hoice", "**P**" for "**P**arallel", "**L**" for "re**L**abeling", "**Rs**" for "**Res**triction", "**I**" for "**I**teraton" and "**Sy**" for "**Sy**nchronization". The first rule in the table is the empty loop rule **El**. The other rules are the action rules, describing transformations of dynamic expressions, which are built using particular algebraic operations. If we cannot merge a rule with stochastic multiactions and a rule with immediate multiactions for some operation then we get the coupled action rules. In such cases, the names of the action rules with immediate multiactions have a suffix '**i**'. To make presentation more compact, the action rules with double conclusion are combined from two distinct action rules with the same premises.

Almost all the rules in Table 3 (excepting **El**, **P2**, **P2i**, **Sy2** and **Sy2i**) resemble those of gsPBC [67], but the former correspond to execution of sets of activities, not of single activities, as in the latter, and our rules have simpler preconditions (if any), since all immediate multiactions in dtsiPBC have the same priority level, unlike those of gsPBC. The preconditions in rules **El**, **C**, **P1**, **I2** and **I3** are needed to ensure that (possibly empty) sets of stochastic multiactions are executed only from *tangible* operative dynamic expressions, such that all operative dynamic expressions structurally equivalent to them are tangible as well. For example, assuming that $tang([G]_{\approx})$ in rule **C**, if $init(G)$ then $G \approx \overline{F}$ for some static expression F and $G[]E \approx \overline{F}[]E \approx \overline{F[]E} \approx F[]\overline{E}$. Hence, it should be guaranteed that $tang([F[]\overline{E}]_{\approx})$,

Table 3: Action and empty loop rules.

El $\dfrac{tang([G]_{\approx})}{G \xrightarrow{\emptyset} G}$

B $\overline{(\alpha,\kappa)} \xrightarrow{\{(\alpha,\kappa)\}} \underline{(\alpha,\kappa)}$

S $\dfrac{G \xrightarrow{\Upsilon} \widetilde{G}}{G; E \xrightarrow{\Upsilon} \widetilde{G}; E,\ E; G \xrightarrow{\Upsilon} E; \widetilde{G}}$

C $\dfrac{G \xrightarrow{\Gamma} \widetilde{G},\ \neg init(G) \vee (init(G) \wedge tang([\overline{E}]_{\approx}))}{G[]E \xrightarrow{\Gamma} \widetilde{G}[]E,\ E[]G \xrightarrow{\Gamma} E[]\widetilde{G}}$

Ci $\dfrac{G \xrightarrow{I} \widetilde{G}}{G[]E \xrightarrow{I} \widetilde{G}[]E,\ E[]G \xrightarrow{I} E[]\widetilde{G}}$

P1 $\dfrac{G \xrightarrow{\Gamma} \widetilde{G},\ tang([H]_{\approx})}{G\|H \xrightarrow{\Gamma} \widetilde{G}\|H,\ H\|G \xrightarrow{\Gamma} H\|\widetilde{G}}$

P1i $\dfrac{G \xrightarrow{I} \widetilde{G}}{G\|H \xrightarrow{I} \widetilde{G}\|H,\ H\|G \xrightarrow{I} H\|\widetilde{G}}$

P2 $\dfrac{G \xrightarrow{\Gamma} \widetilde{G},\ H \xrightarrow{\Delta} \widetilde{H}}{G\|H \xrightarrow{\Gamma+\Delta} \widetilde{G}\|\widetilde{H}}$

P2i $\dfrac{G \xrightarrow{I} \widetilde{G},\ H \xrightarrow{J} \widetilde{H}}{G\|H \xrightarrow{I+J} \widetilde{G}\|\widetilde{H}}$

L $\dfrac{G \xrightarrow{\Upsilon} \widetilde{G}}{G[f] \xrightarrow{f(\Upsilon)} \widetilde{G}[f]}$

Rs $\dfrac{G \xrightarrow{\Upsilon} \widetilde{G},\ a, \hat{a} \notin \mathcal{A}(\Upsilon)}{G \text{ rs } a \xrightarrow{\Upsilon} \widetilde{G} \text{ rs } a}$

I1 $\dfrac{G \xrightarrow{\Upsilon} \widetilde{G}}{[G * E * F] \xrightarrow{\Upsilon} [\widetilde{G} * E * F]}$

I2 $\dfrac{G \xrightarrow{\Gamma} \widetilde{G},\ \neg init(G) \vee (init(G) \wedge tang([\overline{F}]_{\approx}))}{[E * G * F] \xrightarrow{\Gamma} [E * \widetilde{G} * F]}$

I2i $\dfrac{G \xrightarrow{I} \widetilde{G}}{[E * G * F] \xrightarrow{I} [E * \widetilde{G} * F]}$

I3 $\dfrac{G \xrightarrow{\Gamma} \widetilde{G},\ \neg init(G) \vee (init(G) \wedge tang([\overline{F}]_{\approx}))}{[E * F * G] \xrightarrow{\Gamma} [E * F * \widetilde{G}]}$

I3i $\dfrac{G \xrightarrow{I} \widetilde{G}}{[E * F * G] \xrightarrow{I} [E * F * \widetilde{G}]}$

Sy1 $\dfrac{G \xrightarrow{\Upsilon} \widetilde{G}}{G \text{ sy } a \xrightarrow{\Upsilon} \widetilde{G} \text{ sy } a}$

Sy2 $\dfrac{G \text{ sy } a \xrightarrow{\Gamma'+\{(\alpha,\rho)\}+\{(\beta,\chi)\}} \widetilde{G} \text{ sy } a,\ a \in \alpha,\ \hat{a} \in \beta}{G \text{ sy } a \xrightarrow{\Gamma'+\{(\alpha \oplus_a \beta, \rho \cdot \chi)\}} \widetilde{G} \text{ sy } a}$

Sy2i $\dfrac{G \text{ sy } a \xrightarrow{I'+\{(\alpha,\natural_l)\}+\{(\beta,\natural_m)\}} \widetilde{G} \text{ sy } a,\ a \in \alpha,\ \hat{a} \in \beta}{G \text{ sy } a \xrightarrow{I'+\{(\alpha \oplus_a \beta, \natural_{l+m})\}} \widetilde{G} \text{ sy } a}$

which holds iff $tang([\overline{E}]_{\approx})$. The case $E[]G$ is treated similarly. Further, assuming that $tang([G]_{\approx})$ in rule **P1**, it should be guaranteed that $tang([G\|H]_{\approx})$ and $tang([H\|G]_{\approx})$, which holds iff $tang([H]_{\approx})$. The preconditions in rules **I2** and **I3** are analogous to that in rule **C**.

Rule **El** corresponds to one discrete time unit delay while executing no activities and therefore it has no analogues among the rules of gsPBC that adapts the continuous time model.

Rules **P2** and **P2i** have no similar rules in gsPBC, since interleaving semantics of the algebra allows no simultaneous execution of activities. On the other hand, **P2** and **P2i** have in PBC the analogous rule **PAR** that is used to construct step semantics of the calculus, but the former two rules correspond to execution of sets of activities, unlike that of multisets of multiactions in the latter rule. Rules **P2** and **P2i** cannot be merged, since otherwise simultaneous execution of stochastic and immediate multiactions would be allowed.

Rules **Sy2** and **Sy2i** differ from the corresponding synchronization rules in gsPBC, since the probability or the weight of synchronization in the former rules and the rate or the weight of synchronization in the latter rules are calculated in two distinct ways.

Rule **Sy2** establishes that the synchronization of two stochastic multiactions is made by taking the product of their probabilities, since we are considering that both must occur for the synchronization to happen, so this corresponds, in some sense, to the probability of the independent event intersection, but the real situation is more complex, since these stochastic multiactions can also be executed in parallel. Nevertheless, when scoping (the combined operation consisting of synchronization followed by restriction over the same action [16]) is applied over a parallel execution, we get as final result just the simple product of the probabilities, since no normalization is needed there. Multiplication is an associative and commutative binary operation that is distributive over addition, i.e. it fulfills all practical conditions imposed on the synchronization operator in [51]. Further, if both arguments of multiplication are from $(0; 1)$ then the result belongs to the same interval, hence, multiplication naturally maintains probabilistic compositionality in our model. Our approach is similar to the multiplication of rates of the synchronized actions in MTIPP [50] in the case when the rates are less than 1. Moreover, for the probabilities ρ and χ of two stochastic multiactions to be synchronized

we have $\rho \cdot \chi < \min\{\rho, \chi\}$, i.e. multiplication meets the performance requirement stating that the probability of the resulting synchronized stochastic multiaction should be less than the probabilities of the two ones to be synchronized. While performance evaluation, it is usually supposed that the execution of two components together require more system resources and time than the execution of each single one. This resembles the *bounded capacity* assumption from [51]. Thus, multiplication is easy to handle with and it satisfies the algebraic, probabilistic, time and performance requirements. Therefore, we have chosen the product of the probabilities for the synchronization. See also [24, 23] for a discussion about binary operations producing the rates of synchronization in the continuous time setting.

In rule **Sy2i**, we sum the weights of two synchronized immediate multiactions, since the weights can be interpreted as the rewards [91], thus, we collect the rewards. Next, we express that the synchronized execution of immediate multiactions has more importance than that of every single one. The weights of immediate multiactions can also be seen as bonus rewards associated with transitions [12]. The rewards are summed during synchronized execution of immediate multiactions, since in this case all the synchronized activities can be seen as participated in the execution. We prefer to collect more rewards, thus, the transitions providing greater rewards will have a preference and they will be executed with a greater probability. Since execution of immediate multiactions takes no time, we prefer to execute in a step as many synchronized immediate multiactions as possible to get more significant progress in behaviour. Under behavioural progress we understand an advance in executing activities, which does not always imply a progress in time, as in the case when the activities are immediate multiactions. This aspect will be used later, while evaluating performance via analysis of the embedded discrete time Markov chains (EDTMCs) of expressions. Since every state change in EDTMC takes one unit of (local) time, greater advance in operation of the EDTMC allows one to calculate quicker performance indices.

Example 3.3. *In the following cases, the weights of immediate multiactions are interpreted as bonus rewards to be summed while synchronous or parallel execution of the immediate multiactions specifying instantaneous probabilistic choice.*

- *A customer in a shop considers which products to purchase. He will get a bonus (pay points) l when he decides $(\{a\}, \natural_l)$ to buy the product A and, for the deciding $(\{\hat{a}\}, \natural_m)$ to buy the product B, he will have the bonus m. Thus, on every decision to buy both products A and B (first A, and next time B; or first B, then A; or on the decision $\{(\{a\}, \natural_l), \{\hat{a}\}, \natural_m)\}$ to buy A and B together, in one visit to the shop, i.e. in parallel; or on the decision $(\emptyset, \natural_{l+m})$ to buy a kit with A and B, which corresponds to their synchronized composition), the customer will get a bonus $l + m$, this is a standard and well-accepted practice.*

- *A cook in a fast-food restaurant plans his everyday work (cooking a two-component dinner dish of vegetables and meat), that consists in the decision $(\{a\}, \natural_l)$ to perform work A (boil vegetables), for which he will get a payment l, and the decision $(\{\hat{a}\}, \natural_m)$ to perform work B (fry meat), with the payment m. The works A and B are independent, and they can be even done together, since there are several (at least, two) free rings on the electric cooker in the kitchen. Then, on every decision to perform both works A and B (first A, then B; or first B, then A; or on the decision $\{(\{a\}, \natural_l), \{\hat{a}\}, \natural_m)\}$ to perform A and B in parallel; or on the decision $(\emptyset, \natural_{l+m})$ to do a work including both A and B, for example, to warm up a frozen combined (two-in-one) product (consisting of vegetables and meat), prepared by the cook ahead of time, which corresponds to the synchronized composition of works A and B), the cook will get a payment $l + m$, this is logical and fair.*

In the both situations above, more successful customer or cook spends less resources (power, electricity, water, etc.) to get his bonus or payment $l+m$. Thus, the preferred and encouraged way of doing (the ideal behaviour or work) consists in the parallel or the synchronized executing of actions. Since we prefer to collect more bonus rewards, clearly, the decisions providing more rewards must have a preference and should be executed with a greater probability.

The standard approach while system modeling within dtsiPBC is to split the system operations into the probabilistic decision, specified by an immediate multiaction, and the time-consuming work followed, that is specified by one or more stochastic multiactions. It is more natural to interpret weights of immediate multiactions as bonus rewards, subsequently used to determine the decision probabilities, since probabilities of stochastic multiactions are intended to calculate the duration of work.

Table 4: Comparison of inaction, action and empty loop rules.

Rules	State change	Time progress	Activities execution
Inaction rules	−	−	−
Action rules (stochastic multiactions)	±	+	+
Action rules (immediate multiactions)	±	−	+
Empty loop rule	−	+	−

Observe also that we do not have self-synchronization, i.e. synchronization of an activity with itself, since all the (enumerated) activities executed together are considered to be different. This allows us to avoid rather cumbersome and unexpected behaviour, as well as many technical difficulties [16].

In Table 4, inaction rules, action rules (with stochastic or immediate multiactions) and empty loop rule are compared according to the three questions about their application: whether it changes the current state, whether it leads to a time progress, and whether it results in execution of some activities. Positive answers to the questions are denoted by the plus sign while negative ones are specified by the minus sign. If both positive and negative answers can be given to some of the questions in different cases then the plus-minus sign is written. The process states are considered up to structural equivalence of the corresponding expressions, and time progress is not regarded as a state change.

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. The set of *all sets of activities executable in* a state $s = [G]_{\approx}$ is defined as $Exec(s) = \{\Upsilon \mid \exists H \in s,\ \exists \widetilde{H},\ H \xrightarrow{\Upsilon} \widetilde{H}\}$.

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 in general, since the preconditions in the action rules disable executions of the activities with the lower-priority types from every $H \in s$, as the next example shows.

Example 3.4. *Let H, H' be from Example 3.2 and $s = [H]_{\approx} = [H']_{\approx}$. We have $Now(H) = \{\{(\{a\}, \natural_1)\}\}$ and $Now(H') = \{\{(\{b\}, \frac{1}{2})\}\}$. Since only rules* **Ci** *and* **B** *can be applied to H while no action rule can be applied to H', we get $Exec(s) = \{\{(\{a\}, \natural_1)\}\}$. Then, for $H' \in s$ and $\Upsilon = \{(\{b\}, \frac{1}{2})\} \in Now(H')$, we obtain $\Upsilon \notin Exec(s)$.*

The state s is *tangible*, if $Exec(s) \subseteq \boldsymbol{N}^{S\mathcal{L}}_{fin}$. For tangible states we always have $\emptyset \in Exec(s)$, and we may have $Exec(s) = \{\emptyset\}$. Otherwise, the state s is *vanishing*, and in this case $Exec(s) \subseteq \boldsymbol{N}^{\mathcal{I}\mathcal{L}}_{fin} \setminus \{\emptyset\}$.

Since for every $H \in s$, $Now(H)$ containing the multisets of activities with the lower-priority types is not included into $Exec(s)$, and the types of states are determined from the highest-priority types of the executable activities, the state type definitions based on $Now(H)$, $H \in s$, and on $Exec(s)$ are consistent.

Note that if $\Upsilon \in Exec(s)$ then by rules **P2**, **P2i**, **Sy2**, **Sy2i** and definition of $Exec(s)$, $\forall \Xi \subseteq \Upsilon,\ \Xi \neq \emptyset$, we have $\Xi \in Exec(s)$, i.e. $2^{\Upsilon} \setminus \{\emptyset\} \subseteq Exec(s)$.

Since the inaction rules only distribute and move upper and lower bars along the syntax of dynamic expressions, all $H \in s$ have the same underlying static expression F. Process expressions always have a finite length, hence, the number of all (enumerated) activities and the number of all operations in the syntax of F are finite as well. The action rules **Sy2** and **Sy2i** are the only ones that generate new activities. They result from the handshake synchronization of actions and their conjugates belonging to the multiaction parts of the first and second constituent activity, respectively. Since we have a finite number of operators $\mathtt{sy}$ in F and all the multiaction parts of the activities are finite multisets, the number of the new synchronized activities is also finite. The action rules contribute to $Exec(s)$ (in addition to the empty set, if rule **El** is applicable) only the sets consisting both of activities from F and the new activities, produced by **Sy2** and **Sy2i**. Since we have a finite number n of all such activities, we get $|Exec(s)| \leq 2^n < \infty$. Thus, summation and

multiplication by elements from the finite set $Exec(s)$ are well-defined. Similar reasoning can be used to demonstrate that for all dynamic expressions H (not just for those from s), $Now(H)$ is a finite set.

Definition 3.6. 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 $\exists \Upsilon,\ H \xrightarrow{\Upsilon} \widetilde{H}$ then $[\widetilde{H}]_{\approx} \in DR(G)$.

The set of *all tangible states from* $DR(G)$ is denoted by $DR_T(G)$, and the set of *all vanishing states from* $DR(G)$ is denoted by $DR_V(G)$. Clearly, $DR(G) = DR_T(G) \uplus DR_V(G)$ ($\uplus$ denotes disjoint union).

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 set of stochastic multiactions* Υ *is ready for execution in* s or the *weight of the set of immediate multiactions* Υ *which is ready for execution in* s is

$$PF(\Upsilon, s) = \begin{cases} \prod\limits_{(\alpha,\rho)\in\Upsilon} \rho \cdot \prod\limits_{\{(\beta,\chi)\}\in Exec(s)|(\beta,\chi)\notin\Upsilon\}} (1-\chi), & s \in DR_T(G); \\ \sum\limits_{(\alpha,\natural_l)\in\Upsilon} l, & s \in DR_V(G). \end{cases}$$

In the case $\Upsilon = \emptyset$ and $s \in DR_T(G)$ we define

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

If $s \in DR_T(G)$ and $Exec(s) \neq \{\emptyset\}$ then $PF(\Upsilon, s)$ can be interpreted as a *joint* probability of independent events (in a probability sense, i.e. the probability of intersection of these events is equal to the product of their probabilities). Each such an event consists in the positive or negative decision to be executed of a particular stochastic multiaction. Every executable stochastic multiaction decides probabilistically (using its probabilistic part) and independently (from others), if it wants to be executed in s. If Υ is a set of all executable stochastic multiactions which have decided to be executed in s and $\Upsilon \in Exec(s)$ then Υ is ready for execution in s. The multiplication in the definition is used because it reflects the probability of the independent event intersection. Alternatively, when $\Upsilon \neq \emptyset$, $PF(\Upsilon, s)$ can be interpreted as the probability to execute *exclusively* the set of stochastic multiactions Υ in s, i.e. the probability of *intersection* of two events calculated using the conditional probability formula in the form $\mathsf{P}(X \cap Y) = \mathsf{P}(X|Y)\mathsf{P}(Y)$. The event X consists in the execution of Υ in s. The event Y consists in the non-execution in s of all the executable stochastic multiactions not belonging to Υ. Since the mentioned non-executions are obviously independent events, the probability of Y is a product of the probabilities of the non-executions: $\mathsf{P}(Y) = \prod_{\{(\beta,\chi)\}\in Exec(s)|(\beta,\chi)\notin\Upsilon\}}(1-\chi)$. The conditioning of X by Y makes the executions of the stochastic multiactions from Υ independent, since all of them can be executed in parallel in s by definition of $Exec(s)$. Hence, the probability to execute Υ *under condition* that no executable stochastic multiactions not belonging to Υ are executed in s is a product of probabilities of these stochastic multiactions: $\mathsf{P}(X|Y) = \prod_{(\alpha,\rho)\in\Upsilon} \rho$. Thus, the probability that Υ is executed *and* no executable stochastic multiactions not belonging to Υ are executed in s is the probability of X conditioned by Y multiplied by the probability of Y: $\mathsf{P}(X \cap Y) = \mathsf{P}(X|Y)\mathsf{P}(Y) = \prod_{(\alpha,\rho)\in\Upsilon} \rho \cdot \prod_{\{(\beta,\chi)\}\in Exec(s)|(\beta,\chi)\notin\Upsilon\}}(1-\chi)$. When $\Upsilon = \emptyset$, $PF(\Upsilon, s)$ can be interpreted as the probability not to execute in s any executable stochastic multiactions, thus, $PF(\emptyset, s) = \prod_{\{(\beta,\chi)\}\in Exec(s)}(1-\chi)$. When only the empty set of activities can be executed in s, i.e. $Exec(s) = \{\emptyset\}$, we take $PF(\emptyset, s) = 1$, since we stay in s in this case. Note that for $s \in DR_T(G)$ we have $PF(\emptyset, s) \in (0; 1]$, hence, we can stay in s at the next time moment with a certain positive probability.

If $s \in DR_V(G)$ then $PF(\Upsilon, s)$ can be interpreted as the *overall (cumulative)* weight of the immediate multiactions from Υ, i.e. the sum of all their weights. The summation here is used since the weights can be seen as the rewards which are collected [91]. In addition, this means that concurrent execution of the immediate multiactions has more importance than that of every single one. The weights of immediate multiactions can also be interpreted as bonus rewards of transitions [12]. The rewards are summed when immediate multiactions are executed in parallel, because all of them participated in the execution. Since execution of immediate multiactions takes no time, we prefer to

execute in a step as many parallel immediate multiactions as possible to get more progress in behaviour. This aspect will be used later, while evaluating performance on the basis of the EDTMCs of expressions. Note that this reasoning is the same as that used to define the weight of synchronized immediate multiactions in the rule **Sy2i**.

Note that the definition of $PF(\Upsilon, s)$ (as well as the definitions of other probability functions which we shall present) is based on the enumeration of activities which is considered implicit.

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

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

If $s \in DR_T(G)$ then $PT(\Upsilon, s)$ can be interpreted as the *conditional* probability to execute Υ in s calculated using the conditional probability formula in the form $\mathsf{P}(Z|W) = \frac{\mathsf{P}(Z \cap W)}{\mathsf{P}(W)}$. The event Z consists in the exclusive execution of Υ in s, hence, $\mathsf{P}(Z) = PF(\Upsilon, s)$. The event W consists in the exclusive execution of any set (including the empty one) $\Xi \in Exec(s)$ in s. Thus, $W = \cup_j Z_j$, where $\forall j,\ Z_j$ are mutually exclusive events (in a probability sense, i.e. intersection of these events is the empty event) and $\exists i,\ Z = Z_i$. We have $\mathsf{P}(W) = \sum_j \mathsf{P}(Z_j) = \sum_{\Xi \in Exec(s)} PF(\Xi, s)$, because summation reflects the probability of the mutually exclusive event union. Since $Z \cap W = Z_i \cap (\cup_j Z_j) = Z_i = Z$, we have $\mathsf{P}(Z|W) = \frac{\mathsf{P}(Z)}{\mathsf{P}(W)} = \frac{PF(\Upsilon, s)}{\sum_{\Xi \in Exec(s)} PF(\Xi, s)}$. One can also treat $PT(\Upsilon, s)$ and $PF(\Upsilon, s)$ as the *actual* and *potential* probabilities to execute Υ in s, respectively, since we have $PT(\Upsilon, s) = PF(\Upsilon, s)$ only when *all* sets (including the empty one) consisting of the executable stochastic multiactions can be executed in s. In this case, all the mentioned stochastic multiactions can be executed in parallel in s and we have $\sum_{\Xi \in Exec(s)} PF(\Xi, s) = 1$, since this sum collects the products of *all* combinations of the probability parts of the stochastic multiactions and the negations of these parts. But in general, for example, for two stochastic multiactions (α, ρ) and (β, χ) executable in s, it may happen that they cannot be executed in s together, in parallel, i.e. $\emptyset, \{(\alpha, \rho)\}, \{(\beta, \chi)\} \in Exec(s)$, but $\{(\alpha, \rho), (\beta, \chi)\} \notin Exec(s)$. Note that for $s \in DR_T(G)$ we have $PT(\emptyset, s) \in (0; 1]$, hence, there is a non-zero probability to stay in the state s at the next time moment. Then the residence time in s is at least 1 discrete time unit, being 1 when s is left with the next time tick.

If $s \in DR_V(G)$ then $PT(\Upsilon, s)$ can be interpreted as the weight of the set of immediate multiactions Υ which is ready for execution in s *normalized* by the weights of *all* the sets executable in s. This approach is analogous to that used in the EMPA definition of the probabilities of immediate actions executable from the same process state [14] (inspired by way in which the probabilities of conflicting immediate transitions in GSPNs are calculated [6]). The only difference is that we have a step semantics and, for every set of immediate multiactions executed in parallel, we use its cumulative weight. To get the analogy with EMPA possessing interleaving semantics, we should interpret the weights of immediate actions of EMPA as the cumulative weights of the sets of immediate multiactions of dtsiPBC.

The advantage of our two-stage approach to definition of the probability to execute a set of activities is that the resulting probability formula $PT(\Upsilon, s)$ is valid both for (sets of) stochastic and immediate multiactions. It allows one to unify the notation used later while constructing the operational semantics and analyzing performance.

Note that the sum of outgoing probabilities for the expressions belonging to the derivations of G is equal to 1. More formally, $\forall s \in DR(G),\ \sum_{\Upsilon \in Exec(s)} PT(\Upsilon, s) = 1$. This, obviously, follows from the definition of $PT(\Upsilon, s)$, and guarantees that it always defines a probability distribution.

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

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

The summation in the definition above reflects the probability of the mutually exclusive event union, since $\sum_{\{\Upsilon | \exists H \in s,\ \exists \widetilde{H} \in \tilde{s},\ H \stackrel{\Upsilon}{\to} \widetilde{H}\}} PT(\Upsilon, s) = \frac{1}{\sum_{\Xi \in Exec(s)} PF(\Xi, s)} \cdot \sum_{\{\Upsilon | \exists H \in s,\ \exists \widetilde{H} \in \tilde{s},\ H \stackrel{\Upsilon}{\to} \widetilde{H}\}} PF(\Upsilon, s)$, where for each Υ, $PF(\Upsilon, s)$ is the probability of the exclusive execution of Υ in s. Note that $\forall s \in DR(G),\ \sum_{\{\tilde{s} | \exists H \in s,\ \exists \widetilde{H} \in \tilde{s},\ \exists \Upsilon,\ H \stackrel{\Upsilon}{\to} \widetilde{H}\}} PM(s, \tilde{s}) = \sum_{\{\tilde{s} | \exists H \in s,\ \exists \widetilde{H} \in \tilde{s},\ \exists \Upsilon,\ H \stackrel{\Upsilon}{\to} \widetilde{H}\}} \sum_{\{\Upsilon | \exists H \in s,\ \exists \widetilde{H} \in \tilde{s},\ H \stackrel{\Upsilon}{\to} \widetilde{H}\}} PT(\Upsilon, s) = \sum_{\Upsilon \in Exec(s)} PT(\Upsilon, s) = 1$.

Example 3.5. *Let* $E = (\{a\}, \rho)[](\{a\}, \chi)$*, where* $\rho, \chi \in (0; 1)$*.* $DR(\overline{E})$ *consists of the equivalence classes* $s_1 = [\overline{E}]_\approx$ *and* $s_2 = [\underline{E}]_\approx$*. We have* $DR_T(\overline{E}) = \{s_1, s_2\}$*. The execution probabilities are calculated as follows. Since* $Exec(s_1) =$

Table 5: Calculation of the probability functions $PF,\ PT,\ PM$ for $s_1 \in DR(\overline{E})$ and $E = (\{a\},\rho)[](\{a\},\chi)$

$s_1 \backslash \Upsilon$	$\emptyset$	$\{(\{a\},\rho)\}$	$\{(\{a\},\chi)\}$	Σ
PF	$(1-\rho)(1-\chi)$	$\rho(1-\chi)$	$\chi(1-\rho)$	$1-\rho\chi$
PT	$\frac{(1-\rho)(1-\chi)}{1-\rho\chi}$	$\frac{\rho(1-\chi)}{1-\rho\chi}$	$\frac{\chi(1-\rho)}{1-\rho\chi}$	1
PM	$\frac{(1-\rho)(1-\chi)}{1-\rho\chi}$ (s_1)	$\frac{\rho+\chi-2\rho\chi}{1-\rho\chi}$ (s_2)		1

Table 6: Calculation of the probability functions $PF,\ PT,\ PM$ for $s'_1 \in DR(\overline{E'})$ and $E' = (\{a\},\natural_l)[](\{a\},\natural_m)$

$s'_1 \backslash \Upsilon$	$\{(\{a\},\natural_l)\}$	$\{(\{a\},\natural_m)\}$	Σ
PF	l	m	$l+m$
PT	$\frac{l}{l+m}$	$\frac{m}{l+m}$	1
PM	1 (s'_2)		1

$\{\emptyset, \{(\{a\},\rho)\}, \{(\{a\},\chi)\}\}$, *we get* $PF(\{(\{a\},\rho)\}, s_1) = \rho(1-\chi)$, $PF(\{(\{a\},\chi)\}, s_1) = \chi(1-\rho)$ *and* $PF(\emptyset, s_1) = (1-\rho)(1-\chi)$. *Then* $\sum_{\Xi \in Exec(s_1)} PF(\Xi, s_1) = \rho(1-\chi) + \chi(1-\rho) + (1-\rho)(1-\chi) = 1-\rho\chi$. *Thus,* $PT(\{(\{a\},\rho)\}, s_1) = \frac{\rho(1-\chi)}{1-\rho\chi}$, $PT(\{(\{a\},\chi)\}, s_1) = \frac{\chi(1-\rho)}{1-\rho\chi}$ *and* $PT(\emptyset, s_1) = PM(s_1, s_1) = \frac{(1-\rho)(1-\chi)}{1-\rho\chi}$. *Further,* $Exec(s_2) = \{\emptyset\}$, *hence,* $\sum_{\Xi \in Exec(s_2)} PF(\Xi, s_2) = PF(\emptyset, s_2) = 1$ *and* $PT(\emptyset, s_2) = PM(s_2, s_2) = \frac{1}{1} = 1$. *Finally,* $PM(s_1, s_2) = PT(\{(\{a\},\rho)\}, s_1) + PT(\{(\{a\},\chi)\}, s_1) = \frac{\rho(1-\chi)}{1-\rho\chi} + \frac{\chi(1-\rho)}{1-\rho\chi} = \frac{\rho+\chi-2\rho\chi}{1-\rho\chi}$. *In Table 5, the calculation of the probability functions* $PF(\Upsilon, s_1)$, $PT(\Upsilon, s_1)$, $PM(s_1, s)$ *is explained, where* $\Upsilon \in Exec(s_1)$, $s \in \{s_1, s_2\}$ *(the value of* s *is depicted in the parentheses near the value of* $PM(s_1, s)$*) and* $\Sigma = \sum_{\Xi \in Exec(s_1)} PX(\Xi, s_1)$, $PX \in \{PF, PT, PM\}$.

Let $E' = (\{a\},\natural_l)[](\{a\},\natural_m)$, *where* $l, m \in \boldsymbol{R}_{>0}$. $DR(\overline{E'})$ *consists of the equivalence classes* $s'_1 = [\overline{E'}]_\approx$ *and* $s'_2 = [\underline{E'}]_\approx$. *We have* $DR_T(\overline{E'}) = \{s'_2\}$ *and* $DR_V(\overline{E'}) = \{s'_1\}$. *The execution probabilities are calculated as follows. Since* $Exec(s'_1) = \{\{(\{a\},\natural_l)\}, \{(\{a\},\natural_m)\}\}$, *we get* $PF(\{(\{a\},\natural_l)\}, s'_1) = l$ *and* $PF(\{(\{a\},\natural_m)\}, s'_1) = m$. *Then* $\sum_{\Xi \in Exec(s'_1)} PF(\Xi, s'_1) = l + m$. *Thus,* $PT(\{(\{a\},\natural_l)\}, s'_1) = \frac{l}{l+m}$ *and* $PT(\{(\{a\},\natural_m)\}, s'_1) = \frac{m}{l+m}$. *Further,* $Exec(s'_2) = \{\emptyset\}$, *hence,* $\sum_{\Xi \in Exec(s'_2)} PF(\Xi, s'_2) = PF(\emptyset, s'_2) = 1$ *and* $PT(\emptyset, s'_2) = PM(s'_2, s'_2) = \frac{1}{1} = 1$. *Finally,* $PM(s'_1, s'_2) = PT(\{(\{a\},\natural_l)\}, s'_1) + PT(\{(\{a\},\natural_m)\}, s'_1) = \frac{l}{l+m} + \frac{m}{l+m} = 1$. *In Table 6, the calculation of the probability functions* $PF(\Upsilon, s'_1)$, $PT(\Upsilon, s'_1)$, $PM(s'_1, s')$ *is explained, where* $\Upsilon \in Exec(s'_1)$, $s' \in \{s'_2\}$ *(the value of* s' *is depicted in the parentheses near the value of* $PM(s'_1, s')$*) and* $\Sigma = \sum_{\Xi \in Exec(s'_1)} PX(\Xi, s'_1)$, $PX \in \{PF, PT, PM\}$.

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

- the set of *states* is $S_G = DR(G)$;
- the set of *labels* is $L_G = 2^{SI\mathcal{L}} \times (0; 1]$;
- 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 definition of $TS(G)$ is correct, i.e. for every state, the sum of the probabilities of all the transitions starting from it is 1. This is guaranteed by the note after the definition of $PT(\Upsilon, s)$. Thus, we have defined a *generative* model of probabilistic processes [42]. The reason is that the sum of the probabilities of the transitions with all possible labels should be equal to 1, not only of those with the same labels (up to enumeration of activities they include) as in the *reactive* models, and we do not have a nested probabilistic choice as in the *stratified* models.

The transition system $TS(G)$ associated with a dynamic expression G describes all the steps (concurrent executions) that occur at discrete time moments with some (one-step) probability and consist of sets of activities. Every step consisting of stochastic multiactions or the empty step (i.e. that consisting of the empty set 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. 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 $s \stackrel{\Upsilon}{\rightarrow}_{\mathcal{P}} \tilde{s}$. It is interpreted as follows: the probability to change from s to $\tilde{s}$ as a result of executing Υ is $\mathcal{P}$.

Note that for tangible states, Υ can be the empty set, and its execution does not change the current state (i.e. the equivalence class), since we have a loop transition $s \stackrel{\emptyset}{\rightarrow}_{\mathcal{P}} s$ from a tangible state s to itself. This corresponds to the application of the empty loop rule to expressions from the equivalence class. We have to keep track of such executions, called *empty loops*, because they have non-zero probabilities. This follows from the definition of $PF(\emptyset, s)$ and the fact that multiaction probabilities cannot be equal to 1 as they belong to the interval $(0; 1)$. For vanishing states Υ cannot be the empty set, since we must execute some immediate multiactions from them at the current moment.

The step probabilities belong to the interval $(0; 1]$, being 1 in the case when we cannot leave a tangible state s and the only transition leaving it is the empty loop one $s \stackrel{\emptyset}{\rightarrow}_1 s$, or if there is just a single transition from a vanishing state to any other one.

We write $s \stackrel{\Upsilon}{\rightarrow} \tilde{s}$ if $\exists \mathcal{P},\ s \stackrel{\Upsilon}{\rightarrow}_{\mathcal{P}} \tilde{s}$ and $s \rightarrow \tilde{s}$ if $\exists \Upsilon,\ s \stackrel{\Upsilon}{\rightarrow} \tilde{s}$.

The first equivalence we are going to introduce is isomorphism which is a coincidence of systems up to renaming of their components or states.

Definition 3.8. Let $TS(G) = (S_G, L_G, \mathcal{T}_G, s_G)$ and $TS(G') = (S_{G'}, L_{G'}, \mathcal{T}_{G'}, s_{G'})$ be the transition systems of dynamic expressions G and G', respectively. A mapping $\beta : S_G \rightarrow S_{G'}$ 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. $\forall s, \tilde{s} \in S_G,\ \forall \Upsilon,\ s \stackrel{\Upsilon}{\rightarrow}_{\mathcal{P}} \tilde{s} \ \Leftrightarrow\ \beta(s) \stackrel{\Upsilon}{\rightarrow}_{\mathcal{P}} \beta(\tilde{s})$.

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

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

Definition 3.9. Two dynamic expressions G and G' are *equivalent with respect to transition systems*, denoted by $G =_{ts} G'$, if $TS(G) \simeq TS(G')$.

Example 3.6. *Consider the expression* $\mathsf{Stop} = (\{g\}, \frac{1}{2})\ \mathsf{rs}\ g$ *specifying the special process that is only able to perform empty loops with probability* 1 *and never terminates. We could actually use any arbitrary action from* $\mathcal{A}$ *and any probability belonging to the interval* $(0; 1)$ *in the definition of* Stop. *Note that* Stop *is analogous to the one used in the examples of [66]. Then, for* $\rho, \chi, \theta, \phi \in (0; 1)$ *and* $l, m \in \boldsymbol{R}_{>0}$, *let*

$$E = [(\{a\}, \rho) * ((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta))[]((\{e\}, \natural_m); (\{f\}, \phi)))) * \mathsf{Stop}].$$

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

$$s_1 = [[\overline{(\{a\}, \rho)} * ((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta))[]((\{e\}, \natural_m); (\{f\}, \phi)))) * \mathsf{Stop}]]_{\approx},$$
$$s_2 = [[(\{a\}, \rho) * (\overline{(\{b\}, \chi)}; ((((\{c\}, \natural_l); (\{d\}, \theta))[]((\{e\}, \natural_m); (\{f\}, \phi)))) * \mathsf{Stop}]]_{\approx},$$
$$s_3 = [[(\{a\}, \rho) * ((\{b\}, \chi); \overline{((((\{c\}, \natural_l); (\{d\}, \theta))[]((\{e\}, \natural_m); (\{f\}, \phi))))} * \mathsf{Stop}]]_{\approx},$$
$$s_4 = [[(\{a\}, \rho) * ((\{b\}, \chi); ((((\{c\}, \natural_l); \overline{(\{d\}, \theta)})[]((\{e\}, \natural_m); (\{f\}, \phi)))) * \mathsf{Stop}]]_{\approx},$$
$$s_5 = [[(\{a\}, \rho) * ((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta))[]((\{e\}, \natural_m); \overline{(\{f\}, \phi)}))) * \mathsf{Stop}]]_{\approx}.$$

We have $DR_T(\overline{E}) = \{s_1, s_2, s_4, s_5\}$ *and* $DR_V(\overline{E}) = \{s_3\}$.

In Figure 2, the transition system $TS(\overline{E})$ *is presented. The tangible states are depicted in ovals and the vanishing ones are depicted in boxes. For simplicity of the graphical representation, the singleton sets of activities are written without outer braces.*

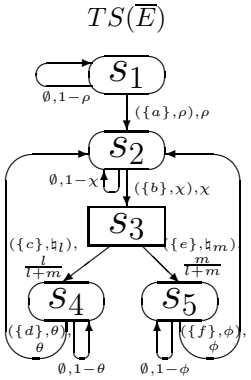

Figure 2: The transition system of $\overline{E}$ for $E = [(\{a\},\rho) * ((\{b\},\chi);((\{c\},\natural_l);(\{d\},\theta))[](\{e\},\natural_m);(\{f\},\phi)))) * \mathsf{Stop}]$.

4. Denotational semantics

In this section, we construct the denotational semantics in terms of a subclass of labeled discrete time stochastic and immediate PNs (LDTSIPNs), called discrete time stochastic and immediate Petri boxes (dtsi-boxes).

4.1. Labeled DTSIPNs

Let us introduce a class of labeled discrete time stochastic and immediate Petri nets (LDTSIPNs), a subclass of DTSPNs [80, 81] (we do not allow the transition probabilities to be equal to 1) extended with transition labeling and immediate transitions. LDTSIPNs resemble in part discrete time deterministic and stochastic PNs (DTDSPNs) [105], as well as discrete deterministic and stochastic PNs (DDSPNs) [104]. DTDSPNs and DDSPNs are the extensions of DTSPNs with deterministic transitions (having fixed delay that can be zero), inhibitor arcs, priorities and guards. In addition, while stochastic transitions of DTDSPNs, like those of DTSPNs, have geometrically distributed delays, stochastic transitions of DDSPNs have discrete time phase-type distributed delays. At the same time, LDTSIPNs are not subsumed by DTDSPNs or DDSPNs, since LDTSIPNs have a step semantics while DTDSPNs and DDSPNs have interleaving one. LDTSIPNs are somewhat similar to labeled weighted DTSPNs (LWDTSPNs) from [30], but in LWDTSPNs there are no immediate transitions, all (stochastic) transitions have weights, the transition probabilities may be equal to 1 and only maximal fireable subsets of the enabled transitions are fired.

Stochastic preemptive time Petri nets (spTPNs) [25] is a discrete time model with a maximal step semantics, where both time ticks and instantaneous parallel firings of maximal transition sets are possible, but the transition steps in LDTSIPNs are not obliged to be maximal. The transition delays in spTPNs are governed by static general discrete distributions, associated with the transitions, while the transitions of LDTSIPNs are only associated with probabilities (or weights), used later to calculate the step probabilities after one unit (from tangible markings) or zero (from vanishing markings) delay. Further, LDTSIPNs have just geometrically distributed or deterministic zero delays in the markings. Moreover, the discrete time tick and concurrent transition firing are treated in spTPNs as different events while firing every (possibly empty) set of stochastic transitions in LDTSIPNs requires one unit time delay. spTPNs are essentially a modification and extension of unlabeled LWDTSPNs with additional facilities, such as inhibitor arcs, priorities, resources, preemptions, schedulers etc. However, the price of such an expressiveness of spTPNs is that the model is rather intricate and difficult to analyze.

Note also that guards in DTDSPNs and DDSPNs, inhibitor arcs and priorities in DTDSPNs, DDSPNs and spTPNs, as well as the maximal step semantics of LWDTSPNs and spTPNs make all these models Turing powerful, resulting in undecidability of many important behavioural properties.

First, we present a formal definition of LDTSIPNs.

Definition 4.1. A *labeled discrete time stochastic and immediate Petri net (LDTSIPN)* is a tuple $N = (P_N, T_N, W_N, \Omega_N, \mathcal{L}_N, M_N)$, where

- P_N and $T_N = Ts_N \uplus Ti_N$ are finite sets of *places* and *stochastic and immediate transitions*, respectively, such that $P_N \cup T_N \neq \emptyset$ and $P_N \cap T_N = \emptyset$;

- $W_N : (P_N \times T_N) \cup (T_N \times P_N) \to \boldsymbol{N}$ is a function providing the *weights of arcs* between places and transitions;
- Ω_N is the *transition probability and weight* function such that
  - $\Omega_N|_{Ts_N} : Ts_N \to (0; 1)$ (it associates stochastic transitions with probabilities);
  - $\Omega_N|_{Ti_N} : Ti_N \to \boldsymbol{R}_{>0}$ (it associates immediate transitions with weights);
- $\mathcal{L}_N : T_N \to \mathcal{L}$ is the *transition labeling* function assigning multiactions to transitions;
- $M_N \in \boldsymbol{N}_{fin}^{P_N}$ is the *initial marking*.

The graphical representation of LDTSIPNs is like that for standard labeled PNs, but with probabilities or weights written near the corresponding transitions. Square boxes of normal thickness depict stochastic transitions, and those with thick borders represent immediate transitions. In the case the probabilities or the weights are not given in the picture, they are considered to be of no importance in the corresponding examples, such as those used to describe the stationary behaviour. The weights of arcs are depicted with them. The names of places and transitions are depicted near them when needed.

We now consider the semantics of LDTSIPNs.

Let N be an LDTSIPN and $t \in T_N$, $U \in \boldsymbol{N}_{fin}^{T_N}$. The *precondition* ${}^\bullet t$ and the *postcondition* $t^\bullet$ of t are the multisets of places defined as $({}^\bullet t)(p) = W_N(p, t)$ and $(t^\bullet)(p) = W_N(t, p)$. The *precondition* ${}^\bullet U$ and the *postcondition* $U^\bullet$ of U are the multisets of places defined as ${}^\bullet U = \sum_{t \in U} {}^\bullet t$ and $U^\bullet = \sum_{t \in U} t^\bullet$. Note that for $U = \emptyset$ we have ${}^\bullet\emptyset = \emptyset = \emptyset^\bullet$.

Let N be an LDTSIPN and $M, \widetilde{M} \in \boldsymbol{N}_{fin}^{P_N}$.

Immediate transitions have a priority over stochastic ones, thus, immediate transitions always fire first, if they can. Suppose that all stochastic transitions have priority 0 and all immediate ones have priority 1.

A transition $t \in T_N$ is *enabled* at M if ${}^\bullet t \subseteq M$. In other words, a transition is enabled in a marking if it has enough tokens in its input places (in the places from its precondition). Let $Ena(M)$ be the set of *all transitions enabled at* M.

Firings of transitions are atomic operations, and transitions may fire concurrently in steps. We assume that all transitions participating in a step should differ, hence, only the sets (not multisets) of transitions may fire. Thus, we do not allow self-concurrency, i.e. firing of transitions in parallel to themselves. This restriction is introduced to avoid some technical difficulties while calculating probabilities for multisets of transitions as we shall see after the following formal definitions. Moreover, we do not need to consider self-concurrency, since denotational semantics of expressions will be defined via dtsi-boxes which are safe LDTSIPNs (hence, no self-concurrency is possible).

The following definition of fireability respects the prioritization among different types of transitions. A set of transitions $U \subseteq Ena(M)$ is *fireable* at a marking M, if ${}^\bullet U \subseteq M$ and one of the following holds:

1. $\emptyset \neq U \subseteq Ti_N$; or
2. $U \subseteq Ts_N$ and $Ena(M) \subseteq Ts_N$.

In other words, a set of transitions U is fireable at a marking M, if it has enough tokens in its input places at M and the following holds. If U consists of *immediate* transitions then it is enabled, since no additional condition is needed for its fireability. If U is empty or it consists of *stochastic* transitions then there exist no immediate transitions enabled at M. Let $Fire(M)$ be the set of *all transition sets fireable at* M.

By the definition of fireability, it follows that $Fire(M) \subseteq 2^{Ti_N} \setminus \{\emptyset\}$ or $Fire(M) \subseteq 2^{Ts_N}$. The marking M is *tangible*, denoted by $tang(M)$, if $Fire(M) \subseteq 2^{Ts_N}$. For a tangible marking M we always have $\emptyset \in Fire(M)$ by the definition of fireability (item 2), hence, we may have $Fire(M) = \{\emptyset\}$. Otherwise, the marking M is *vanishing*, denoted by $vanish(M)$, and in this case $Fire(M) \subseteq 2^{Ti_N} \setminus \{\emptyset\}$. A transition $t \in Ena(M)$ is *fireable* at a marking M, denoted by $t \in Fire(M)$, if $\{t\} \in Fire(M)$. If $stang(M)$ then a stochastic transition $t \in Fire(M)$ fires with probability $\Omega_N(t)$ when no different stochastic transition is fireable in Q, i.e. $Fire(Q) = \{\emptyset, \{t\}\}$. By the definition of fireability, $\forall U \in Fire(Q)\, 2^U \setminus \{\emptyset\} \subseteq Fire(Q)$.

Let $U \in Fire(M)$ and $U \neq \emptyset$. The *probability that the set of stochastic transitions U is ready for firing at M* or the *weight of the set of immediate transitions U which is ready for firing at M* is

$$PF(U, M) = \begin{cases} \displaystyle\prod_{t \in U} \Omega_N(t) \cdot \prod_{\{u \in Fire(M) | u \notin U\}} (1 - \Omega_N(u)), & tang(M); \\ \displaystyle\sum_{t \in U} \Omega_N(t), & vanish(M). \end{cases}$$

In the case $U = \emptyset$ and $tang(M)$ we define

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

Let $U \in Fire(Q)$. Besides U, some other sets of transitions may be ready for firing at M, hence, a kind of conditioning or normalization is needed to calculate the firing probability. The concurrent firing of the transitions from U changes the marking M to $\widetilde{M} = M - {}^\bullet U + U^\bullet$, denoted by $M \stackrel{U}{\rightarrow_{\mathcal{P}}} \widetilde{M}$, where $\mathcal{P} = PT(U, M)$ is the *probability that the set of transitions U fires at M* defined as

$$PT(U, M) = \frac{PF(U, M)}{\sum_{V \in Fire(M)} PF(V, M)}.$$

Note that in the case $U = \emptyset$ and $tang(M)$ we have $M = \widetilde{M}$.

The advantage of our two-stage approach to definition of the probability that a set of transitions fires is that the resulting probability formula $PT(U, M)$ is valid both for (sets of) stochastic and immediate transitions. It allows one to unify the notation used later while constructing the denotational semantics and analyzing performance.

Note that for all markings of an LDTSIPN N, the sum of outgoing probabilities is equal to 1. More formally, $\forall M \in \mathbf{N}_{fin}^{P_N},\ \sum_{U \in Fire(M)} PT(U, M) = 1$. This obviously follows from the definition of $PT(U, M)$ and guarantees that it defines a probability distribution.

We write $M \stackrel{U}{\rightarrow} \widetilde{M}$ if $\exists \mathcal{P},\ M \stackrel{U}{\rightarrow_{\mathcal{P}}} \widetilde{M}$ and $M \rightarrow \widetilde{M}$ if $\exists U,\ M \stackrel{U}{\rightarrow} \widetilde{M}$.

The *probability to move from M to $\widetilde{M}$ by firing any set of transitions* is

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

Since $PM(M, \widetilde{M})$ is the probability for *any* (including the empty one) transition set to change from marking M to $\widetilde{M}$, we use summation in the definition. Note that $\forall M \in \mathbf{N}_{fin}^{P_N},\ \sum_{\{\widetilde{M} | M \rightarrow \widetilde{M}\}} PM(M, \widetilde{M}) = \sum_{\{\widetilde{M} | M \rightarrow \widetilde{M}\}} \sum_{\{U | M \stackrel{U}{\rightarrow} \widetilde{M}\}} PT(U, M) = \sum_{U \in Fire(M)} PT(U, M) = 1$.

Definition 4.2. Let N be an LDTSIPN. The *reachability set* of N, denoted by $RS(N)$, is the minimal set of markings such that

- $M_N \in RS(N)$;
- if $M \in RS(N)$ and $M \rightarrow \widetilde{M}$ then $\widetilde{M} \in RS(N)$.

Definition 4.3. Let N be an LDTSIPN. The *reachability graph* of N is a (labeled probabilistic) transition system $RG(N) = (S_N, L_N, \mathcal{T}_N, s_N)$, where

- the set of *states* is $S_N = RS(N)$;
- the set of *labels* is $L_N = 2^{T_N} \times (0; 1]$;
- the set of *transitions* is $\mathcal{T}_N = \{(M, (U, \mathcal{P}), \widetilde{M}) \mid M, \widetilde{M} \in RS(N),\ M \stackrel{U}{\rightarrow_{\mathcal{P}}} \widetilde{M}\}$;
- the *initial state* is $s_N = M_N$.

The set of *all tangible markings from* $RS(N)$ is denoted by $RS_T(N)$, and the set of *all vanishing markings from* $RS(N)$ is denoted by $RS_V(N)$. Obviously, $RS(N) = RS_T(N) \uplus RS_V(N)$.

4.2. Algebra of dtsi-boxes

We now introduce discrete time stochastic and immediate Petri boxes and the algebraic operations to define a net representation of dtsiPBC expressions.

Definition 4.4. A *discrete time stochastic and immediate Petri box (dtsi-box)* is a tuple $N = (P_N, T_N, W_N, \Lambda_N)$, where

- P_N and T_N are finite sets of *places* and *transitions*, respectively, such that $P_N \cup T_N \neq \emptyset$ and $P_N \cap T_N = \emptyset$;
- $W_N : (P_N \times T_N) \cup (T_N \times P_N) \to \boldsymbol{N}$ is a function providing the *weights of arcs* between places and transitions;
- Λ_N is the *place and transition labeling* function such that
  - $\Lambda_N|_{P_N} : P_N \to \{\mathsf{e}, \mathsf{i}, \mathsf{x}\}$ (it specifies *entry, internal* and *exit* places, respectively);
  - $\Lambda_N|_{T_N} : T_N \to \{\varrho \mid \varrho \subseteq 2^{\mathcal{SIL}} \times \mathcal{SIL}\}$ (it associates transitions with the *relabeling relations* on activities).

Moreover, $\forall t \in T_N,\ {}^\bullet t \neq \emptyset \neq t^\bullet$. In addition, for the set of *entry* places of N, defined as ${}^\circ N = \{p \in P_N \mid \Lambda_N(p) = \mathsf{e}\}$, and for the set of *exit* places of N, defined as $N^\circ = \{p \in P_N \mid \Lambda_N(p) = \mathsf{x}\}$, the following condition holds: ${}^\circ N \neq \emptyset \neq N^\circ,\ {}^\bullet({}^\circ N) = \emptyset = ({N^\circ})^\bullet$.

A dtsi-box is *plain* if $\forall t \in T_N,\ \exists (\alpha, \kappa) \in \mathcal{SIL},\ \Lambda_N(t) = \varrho_{(\alpha,\kappa)}$, where $\varrho_{(\alpha,\kappa)} = \{(\emptyset, (\alpha, \kappa))\}$ is a *constant relabeling* that can be identified with the activity (α, κ). A *marked plain dtsi-box* is a pair (N, M_N), where N is a plain dtsi-box and $M_N \in \boldsymbol{N}^{P_N}_{fin}$ is its marking. We shall use the following notation: $\overline{N} = (N, {}^\circ N)$ and $\underline{N} = (N, N^\circ)$. Note that a marked plain dtsi-box $(P_N, T_N, W_N, \Lambda_N, M_N)$ could be interpreted as the LDTSIPN $(P_N, T_N, W_N, \Omega_N, \mathcal{L}_N, M_N)$, where functions Ω_N and $\mathcal{L}_N$ are defined as follows: $\forall t \in T_N\ \Omega_N(t) = \kappa$ if $\kappa \in (0; 1)$; or $\Omega_N(t) = l$ if $\kappa = \natural_l,\ l \in \boldsymbol{R}_{>0}$; and $\mathcal{L}_N(t) = \alpha$, where $\Lambda_N(t) = \varrho_{(\alpha,\kappa)}$. Behaviour of the marked dtsi-boxes follows from the firing rule of LDTSIPNs. A plain dtsi-box N is *n-bounded* ($n \in \boldsymbol{N}$) if $\overline{N}$ is so, i.e. $\forall M \in RS(\overline{N}),\ \forall p \in P_N,\ M(p) \leq n$, and it is *safe* if it is 1-bounded. A plain dtsi-box N is *clean* if $\forall M \in RS(\overline{N}),\ {}^\circ N \subseteq M \Rightarrow M = {}^\circ N$ and $N^\circ \subseteq M \Rightarrow M = N^\circ$, i.e. if there are tokens in all its entry (exit) places then no other places have tokens.

The structure of the plain dtsi-box corresponding to a static expression is constructed like in PBC [17, 16], i.e. we use simultaneous refinement and relabeling meta-operator (net refinement) in addition to the *operator dtsi-boxes* corresponding to the algebraic operations of dtsiPBC and featuring transformational transition relabelings. Operator dtsi-boxes specify n-ary functions from plain dtsi-boxes to plain dtsi-boxes (we have $1 \leq n \leq 3$ in dtsiPBC). Thus, as we shall see in Theorem 4.1, the resulting plain dtsi-boxes are safe and clean. In the definition of the denotational semantics, we shall apply standard constructions used for PBC. Let Θ denote *operator box* and u denote *transition name* from the PBC setting.

The relabeling relations $\varrho \subseteq 2^{\mathcal{SIL}} \times \mathcal{SIL}$ are defined as follows:

- $\varrho_{id} = \{(\{(\alpha, \kappa)\}, (\alpha, \kappa)) \mid (\alpha, \kappa) \in \mathcal{SIL}\}$ is the *identity relabeling* keeping the interface as it is;
- $\varrho_{(\alpha,\kappa)} = \{(\emptyset, (\alpha, \kappa))\}$ is the *constant relabeling* that can be identified with $(\alpha, \kappa) \in \mathcal{SIL}$ itself;
- $\varrho_{[f]} = \{(\{(\alpha, \kappa)\}, (f(\alpha), \kappa)) \mid (\alpha, \kappa) \in \mathcal{SIL}\}$;
- $\varrho_{\mathsf{rs}\ a} = \{(\{(\alpha, \kappa)\}, (\alpha, \kappa)) \mid (\alpha, \kappa) \in \mathcal{SIL},\ a, \hat{a} \notin \alpha\}$;
- $\varrho_{\mathsf{sy}\ a}$ is the least relabeling relation containing ϱ_{id} such that if $(\Upsilon, (\alpha, \kappa)), (\Xi, (\beta, \lambda)) \in \varrho_{\mathsf{sy}\ a},\ a \in \alpha,\ \hat{a} \in \beta$ then
  - $(\Upsilon + \Xi, (\alpha \oplus_a \beta, \kappa \cdot \lambda)) \in \varrho_{\mathsf{sy}\ a}$, if $\kappa, \lambda \in (0; 1)$;
  - $(\Upsilon + \Xi, (\alpha \oplus_a \beta, \natural_{l+m})) \in \varrho_{\mathsf{sy}\ a}$ if $\kappa = \natural_l,\ \lambda = \natural_m,\ l, m \in \boldsymbol{R}_{>0}$.

The plain dtsi-boxes $N_{(\alpha,\rho)_t},\ N_{(\alpha,\natural_l)_t}$, where $\rho \in (0; 1)$ and $l \in \boldsymbol{R}_{>0}$, and operator dtsi-boxes are presented in Figure 3. The label i of internal places is usually omitted.

In the case of the iteration, a decision that we must take is the selection of the operator box that we shall use for it, since we have two proposals in plain PBC for that purpose [16]. One of them provides us with a safe version with six transitions in the operator box, but there is also a simpler version, which has only three transitions. In general, in PBC,

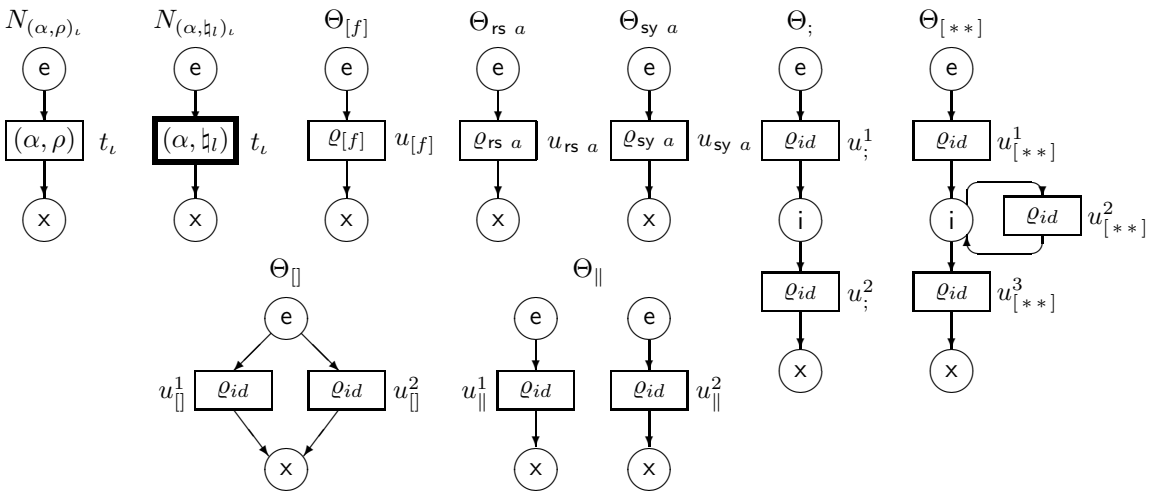

Figure 3: The plain and operator dtsi-boxes.

with the latter version we may generate 2-bounded nets, which only occurs when a parallel behavior appears at the highest level of the body of the iteration. Nevertheless, in our case, and due to the syntactical restriction introduced for regular terms, this particular situation cannot occur, so that the net obtained will be always safe.

To construct the semantic function that associates a plain dtsi-box with every static expression of dtsiPBC, we introduce the *enumeration* function $Enu : T \to Num$, which associates the numberings with transitions of a plain dtsi-box $N = (P, T, W, \Lambda)$ in accordance with those of activities. In the case of synchronization, the function associates with the resulting new transition the concatenation of the parenthesized numberings of the transitions it comes from.

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

- $Box_{dtsi}((\alpha,\kappa)_\iota) = N_{(\alpha,\kappa)_\iota}$. Since a single transition t_ι corresponds to the activity $(\alpha,\kappa)_\iota \in \mathcal{SIL}$, their numberings coincide:

$$Enu(t_\iota) = \iota.$$

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

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

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

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

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

$$Enu(t) = Enu_E(t),\ t \in T_E,\ a, \hat{a} \notin \alpha,\ \Lambda_E(t) = \varrho_{(\alpha,\kappa)}.$$

- $Box_{dtsi}(E \text{ sy } a) = \Theta_{\text{sy } a}(Box_{dtsi}(E))$. Note that $\forall v, w \in T_E$, such that $\Lambda_E(v) = \varrho_{(\alpha,\kappa)}$, $\Lambda_E(w) = \varrho_{(\beta,\lambda)}$ and $a \in \alpha$, $\hat{a} \in \beta$, the new transition t resulting from synchronization of v and w has the label $\Lambda(t) = \varrho_{(\alpha \oplus_a \beta, \kappa \cdot \lambda)}$ if t is a stochastic transition ($\kappa, \lambda \in (0; 1)$); or $\Lambda(t) = \varrho_{(\alpha \oplus_a \beta, \natural_{l+m})}$ if t is an immediate one ($\kappa = \natural_l$, $\lambda = \natural_m$, $l, m \in \boldsymbol{R}_{>0}$); and the numbering $Enu(t) = (Enu_E(v))(Enu_E(w))$. Thus, the enumeration function is defined as

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

  According to the definition of $\varrho_{\text{sy } a}$, the synchronization is only possible when all the transitions in the set are stochastic or when all of them are immediate. If we synchronize the same set of transitions in different orders, we obtain several resulting transitions with the same label and probability or weight, but with the different numberings having the same content. Then, we only consider a single transition from the resulting ones in the plain dtsi-box to avoid introducing redundant transitions.

  For example, if the transitions t and u are generated by synchronizing v and w in different orders, we have $\Lambda(t) = \varrho_{(\alpha \oplus_a \beta, \kappa \cdot \lambda)} = \Lambda(u)$ for stochastic transitions ($\kappa, \lambda \in (0; 1)$) or $\Lambda(t) = \varrho_{(\alpha \oplus_a \beta, \natural_{l+m})} = \Lambda(u)$ for immediate ones ($\kappa = \natural_l$, $\lambda = \natural_m$, $l, m \in \boldsymbol{R}_{>0}$), but $Enu(t) = (Enu_E(v))(Enu_E(w)) \neq (Enu_E(w))(Enu_E(v)) = Enu(u)$, whereas $Cont(Enu(t)) = Cont(Enu(v)) \cup Cont(Enu(w)) = Cont(Enu(u))$. Then only one transition t (or, symmetrically, u) will appear in $Box_{dtsi}(E \text{ sy } a)$.

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

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

We now can formally define the denotational semantics as a homomorphism.

Definition 4.5. Let $(\alpha, \kappa) \in \mathcal{SIL}$, $a \in Act$ and $E, F, K \in RegStatExpr$. The *denotational semantics* of dtsiPBC is a mapping Box_{dtsi} from $RegStatExpr$ into the domain of plain dtsi-boxes defined as follows:

1. $Box_{dtsi}((\alpha, \kappa)_\iota) = N_{(\alpha,\kappa)_\iota}$;
2. $Box_{dtsi}(E \circ F) = \Theta_\circ(Box_{dtsi}(E), Box_{dtsi}(F))$, $\circ \in \{;, [], \|\}$;
3. $Box_{dtsi}(E[f]) = \Theta_{[f]}(Box_{dtsi}(E))$;
4. $Box_{dtsi}(E \circ a) = \Theta_{\circ a}(Box_{dtsi}(E))$, $\circ \in \{\text{rs}, \text{sy}\}$;
5. $Box_{dtsi}([E * F * K]) = \Theta_{[**]}(Box_{dtsi}(E), Box_{dtsi}(F), Box_{dtsi}(K))$.

The dtsi-boxes of dynamic expressions can be defined as well. For $E \in RegStatExpr$, let $Box_{dtsi}(\overline{E}) = \overline{Box_{dtsi}(E)}$ and $Box_{dtsi}(\underline{E}) = \underline{Box_{dtsi}(E)}$.

Note that this definition is compositional in the sense that, for any arbitrary dynamic expression, we may decompose it in some inner dynamic and static expressions, for which we may apply the definition, thus obtaining the corresponding plain dtsi-boxes, which can be joined according to the term structure (by definition of Box_{dtsi}), the resulting plain box being marked in the places that were marked in the argument nets.

Theorem 4.1. *For any static expression E, $Box_{dtsi}(\overline{E})$ is safe and clean.*

Proof. The structure of the net is obtained as in PBC [17, 16], combining both refinement and relabeling. Consequently, the dtsi-boxes thus obtained will be safe and clean. □

Let $\simeq$ denote isomorphism between transition systems and reachability graphs that binds their initial states. Note that the names of transitions of the dtsi-box corresponding to a static expression could be identified with the enumerated activities of the latter.

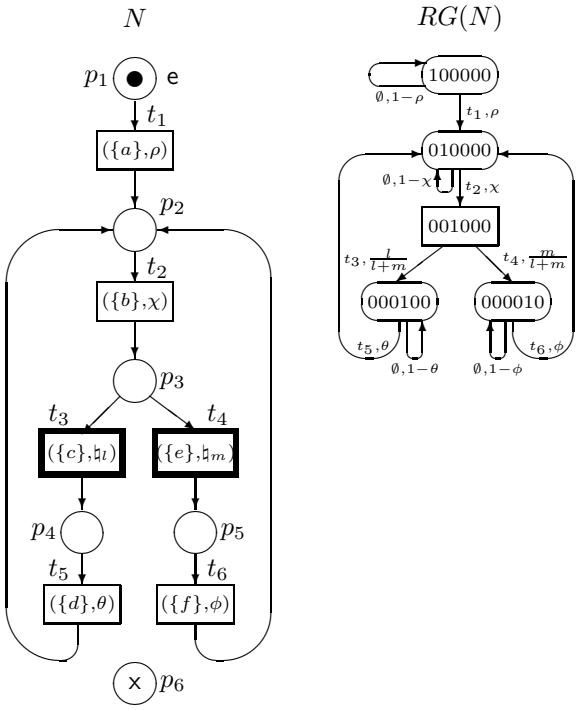

Figure 4: The marked dtsi-box $N = Box_{dtsi}(\overline{E})$ for $E = [(\{a\},\rho) * ((\{b\},\chi); ((\{c\},\natural_l); (\{d\},\theta))[](\{e\},\natural_m); (\{f\},\phi)))) * \mathsf{Stop}]$ and its reachability graph.

Theorem 4.2. *For any static expression E,*

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

Proof. See Appendix A.1. □

Example 4.1. *Let E be from Example 3.6. In Figure 4, the marked dtsi-box $N = Box_{dtsi}(\overline{E})$ and its reachability graph RG(N) are presented. It is easy to see that $TS(\overline{E})$ and RG(N) are isomorphic.*

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

Example 4.2. *Let $E = [((\{a\},\frac{1}{2}) * ((\{b\},\frac{1}{2}) || (\{c\},\frac{1}{2})) * (\{d\},\frac{1}{2})]$. In Figure 5, the marked dtsi-box $N = Box_{dtsi}(\overline{E})$ and its reachability graph RG(N) are presented. In the marking $(0,1,1,2,0,0)$ there are 2 tokens in the place p_4. Symmetrically, at the marking $(0,1,1,0,2,0)$ there are 2 tokens in the place p_5. Thus, allowing concurrency in the second argument of iteration in the expression $\overline{E}$ can lead to non-safeness of the corresponding marked dtsi-box N, though, it is 2-bounded in the worst case [16]. The origin of the problem is that N has a self-loop with two subnets which can function independently. Therefore, we have decided to consider regular expressions only, since the alternative, which is a safe version of the iteration operator with six arguments in the corresponding dtsi-box, like that from [16], is rather cumbersome and has too intricate Petri net interpretation. Our motivation was to keep the algebraic and Petri net specifications as simple as possible.*

5. Performance evaluation

In this section we demonstrate how Markov chains corresponding to the expressions and dtsi-boxes can be constructed and then used for performance evaluation.

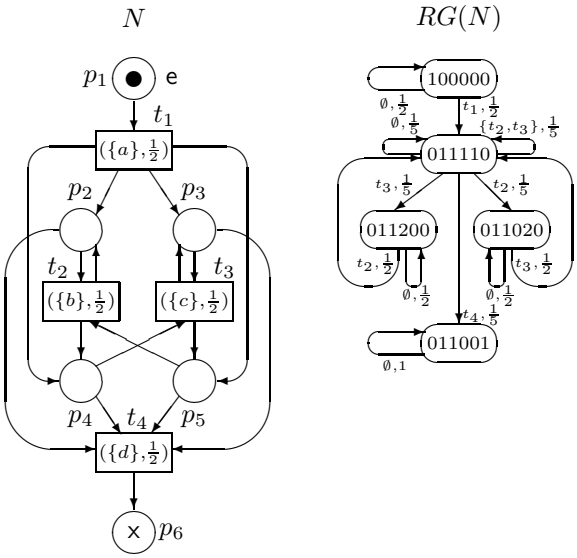

Figure 5: The marked dtsi-box $N = Box_{dtsi}(\overline{E})$ for $E = [((\{a\}, \frac{1}{2}) * ((\{b\}, \frac{1}{2}) \| (\{c\}, \frac{1}{2})) * (\{d\}, \frac{1}{2})]$ and its reachability graph.

5.1. Analysis of the underlying SMC

For a dynamic expression G, a discrete random variable $\xi(s)$ is associated with every tangible state $s \in DR_T(G)$. The variable captures the residence (sojourn) time in the state. One can interpret staying in a state at the next discrete time moment as a failure and leaving it as a success in some trial series. It is easy to see that $\xi(s)$ is geometrically distributed with the parameter $1 - PM(s, s)$, since the probability to stay in s for $k - 1$ time moments and leave it at the moment $k \geq 1$, called the probability mass function (PMF) of the residence time in s, is $p_{\xi(s)}(k) = \mathsf{P}(\xi(s) = k) = PM(s, s)^{k-1}(1 - PM(s, s))$ $(k \in \mathbf{N}_{\geq 1})$ (the residence time in s is k in this case). Hence, the probability distribution function (PDF) of the residence time in s is $F_{\xi(s)}(k) = \mathsf{P}(\xi(s) < k) = 1 - PM(s, s)^{k-1}$ $(k \in \mathbf{N}_{\geq 1})$ (the probability that the residence time in s is less than k).

Note that the residence time ∞ in an absorbing tangible state s can be interpreted as a random variable $\xi(s)$ that is geometrically distributed with the parameter $0 = 1 - PM(s, s)$. In that case, $PM(s, s) = 1$ and there exists no finite residence time value with a positive probability. Hence, $p_{\xi(s)}(k) = PM(s, s)^{k-1}(1 - PM(s, s)) = 1^{k-1} \cdot 0 = 0$ $(k \in \mathbf{N}_{\geq 1})$, i.e. the probability that the residence time is k equals 0 for every $k \geq 1$. Then we cannot leave s for a different state after any number of time ticks and we stay in s for infinite time.

The mean value formula for the geometrical distribution allows us to calculate the average sojourn time in $s \in DR_T(G)$ as $SJ(s) = \frac{1}{1-PM(s,s)}$. Clearly, 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 $VAR(s)$, $s \in DR(G)$.

To evaluate performance of the system specified by a dynamic expression G, we should investigate the stochastic process associated with it. The process is the underlying SMC [91, 62], denoted by $SMC(G)$, which can be analyzed by extracting from it the embedded (absorbing) discrete time Markov chain (EDTMC) corresponding to G, denoted

by $EDTMC(G)$. The construction of the latter is analogous to that applied in the context of generalized stochastic PNs (GSPNs) in [74, 5, 6], and also in the framework of discrete time deterministic and stochastic PNs (DTDSPNs) in [105], as well as within discrete deterministic and stochastic PNs (DDSPNs) [104]. $EDTMC(G)$ only describes the state changes of $SMC(G)$ while ignoring its time characteristics. Thus, to construct the EDTMC, we should abstract from all time aspects of behaviour of the SMC, i.e. from the sojourn time in its states. The (local) sojourn time in every state of the EDTMC is equal to one discrete time unit. It is well-known that every SMC is fully described by the EDTMC and the state sojourn time distributions (the latter can be specified by the vector of PDFs of residence time in the states) [48].

Let G be a dynamic expression and $s, \tilde{s} \in DR(G)$. The transition system $TS(G)$ can have self-loops going from a state to itself which have a non-zero probability. Obviously, the current state remains unchanged in this case.
Let $s \to s$. The *probability to stay in s due to k ($k \geq 1$) self-loops* is

$$PM(s, s)^k.$$

The *self-loops abstraction factor in the state s* is

$$SL(s) = \begin{cases} \frac{1}{1-PM(s,s)}, & s \to s; \\ 1, & \text{otherwise.} \end{cases}$$

The *self-loops abstraction vector* of G, denoted by SL, has the elements $SL(s),\ s \in DR(G)$.
Let $s \to \tilde{s}$ and $s \neq \tilde{s}$, i.e. $PM(s, s) < 1$. The *probability to move from s to $\tilde{s}$ by executing any set of activities after possible self-loops* is

$$PM^*(s, \tilde{s}) = \left\{ \begin{array}{ll} PM(s, \tilde{s}) \sum_{k=0}^{\infty} PM(s, s)^k = \frac{PM(s,\tilde{s})}{1-PM(s,s)}, & s \to s; \\ PM(s, \tilde{s}), & \text{otherwise;} \end{array} \right\} = SL(s)PM(s, \tilde{s}).$$

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

Note that $\forall s \in DR_T(G),\ SL(s) = \frac{1}{1-PM(s,s)} = SJ(s)$, hence, $\forall s \in DR_T(G)$ with $PM(s, s) < 1$, it holds $PM^*(s, \tilde{s}) = SJ(s)PM(s, \tilde{s})$, since we always have the empty loop (which is a self-loop) $s \xrightarrow{\emptyset} s$ from every tangible state s. Empty loops are not possible from vanishing states, hence, $\forall s \in DR_V(G)$ with $PM(s, s) < 1$, it holds $PM^*(s, \tilde{s}) = \frac{PM(s,\tilde{s})}{1-PM(s,s)}$, when there are non-empty self-loops (produced by iteration) from s, or $PM^*(s, \tilde{s}) = PM(s, \tilde{s})$, when there are no self-loops from s. Further, we suppose that all (if any) loops among vanishing states are "transient" rather than "absorbing", as in [75, 6]. Then for each s with $PM(s, s) = 1$ (absorbing state) we have $s \in DR_T(G)$, since there exist no absorbing vanishing states, hence, $\forall s \in DR_V(G)\ PM(s, s) < 1$.

Note that after abstraction from the probabilities of transitions which do not change the states, the remaining transition probabilities are normalized. In order to calculate transition probabilities $PT(\Upsilon, s)$, we had to normalize $PF(\Upsilon, s)$. Then, to obtain transition probabilities of the state-changing steps $PM^*(s, \tilde{s})$, we now have to normalize $PM(s, \tilde{s})$. Thus, we have a two-stage normalization as a result.

Notice that $PM^*(s, \tilde{s})$ defines a probability distribution, since $\forall s \in DR(G)$, such that s is not an absorbing state (i.e. $PM(s, s) < 1$, hence, there are transitions to different states after possible self-loops from it) we have
$\sum_{\{\tilde{s} | s \to \tilde{s},\ s \neq \tilde{s}\}} PM^*(s, \tilde{s}) = \frac{1}{1-PM(s,s)} \sum_{\{\tilde{s} | s \to \tilde{s},\ s \neq \tilde{s}\}} PM(s, \tilde{s}) = \frac{1}{1-PM(s,s)}(1 - PM(s, s)) = 1$.

We decided to consider self-loops followed only by a state-changing step just for convenience. Alternatively, we could take a state-changing step followed by self-loops or a state-changing step preceded and followed by self-loops. In all these three cases our sequence begins or/and ends with the loops which do not change states. At the same time, the overall probabilities of the evolutions can differ, since self-loops have positive probabilities. To avoid inconsistency of definitions and too complex description, we consider sequences ending with a state-changing step. It resembles in some sense a construction of branching bisimulation [41] taking self-loops instead of silent transitions. Further, we shall not abstract from self-loops with probability 1 while constructing EDTMCs, to maintain a probability distribution among transitions (actually, a single transition to the same state) from every state with such a self-loop.

Definition 5.1. Let G be a dynamic expression. The *embedded (absorbing) discrete time Markov chain (EDTMC)* of G, denoted by $EDTMC(G)$, has the state space $DR(G)$, the initial state $[G]_{\approx}$ and the transitions $s \twoheadrightarrow_{\mathcal{P}} \tilde{s}$, if $s \to \tilde{s}$ and $s \neq \tilde{s}$, where $\mathcal{P} = PM^*(s, \tilde{s})$; or $s \twoheadrightarrow_1 s$, if $PM(s, s) = 1$.

The *underlying SMC* of G, denoted by $SMC(G)$, has the EDTMC $EDTMC(G)$ and the sojourn time in every $s \in DR_T(G)$ is geometrically distributed with the parameter $1 - PM(s, s)$ while the sojourn time in every $s \in DR_V(G)$ is equal to 0.

EDTMCs and underlying SMCs of static expressions can be defined as well. For $E \in RegStatExpr$, let $EDTMC(E) = EDTMC(\overline{E})$ and $SMC(E) = SMC(\overline{E})$.

Let G be a dynamic expression. The elements $\mathcal{P}^*_{ij}$ $(1 \le i, j \le n = |DR(G)|)$ of the (one-step) transition probability matrix (TPM) $\mathbf{P}^*$ for $EDTMC(G)$ are defined as

$$\mathcal{P}^*_{ij} = \begin{cases} PM^*(s_i, s_j), & s_i \to s_j,\ i \ne j; \\ 1, & PM(s_i, s_i) = 1,\ i = j; \\ 0, & \text{otherwise.} \end{cases}$$

The transient (k-step, $k \in \boldsymbol{N}$) PMF $\psi^*[k] = (\psi^*[k](s_1), \ldots, \psi^*[k](s_n))$ for $EDTMC(G)$ is calculated as

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

where $\psi^*[0] = (\psi^*[0](s_1), \ldots, \psi^*[0](s_n))$ is the initial PMF defined as

$$\psi^*[0](s_i) = \begin{cases} 1, & s_i = [G]_\approx; \\ 0, & \text{otherwise.} \end{cases}$$

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

The steady-state PMF $\psi^* = (\psi^*(s_1), \ldots, \psi^*(s_n))$ for $EDTMC(G)$ is a solution of the equation system

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

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

Note that the vector ψ^* exists and is unique, if $EDTMC(G)$ is ergodic. Then $EDTMC(G)$ has a single steady state, and we have $\psi^* = \lim_{k\to\infty} \psi^*[k]$. We shall consider only Markov chains with at most one steady state.

The steady-state PMF for the underlying semi-Markov chain $SMC(G)$ is calculated via multiplication of every $\psi^*(s_i)$ $(1 \le i \le n)$ by the average sojourn time $SJ(s_i)$ in the state s_i, after which we normalize the resulting values. Note that for each tangible state $s \in DR_T(G)$ we have $SJ(s) > 1$, since $SJ(s) = \frac{1}{1-PM(s,s)}$, where $PM(s, s) \ge PM(s, \emptyset)$, and by definition of $PM(s, \emptyset)$, it holds $PM(s, \emptyset) > 0$ because the probabilities of stochastic multiactions are always less than 1. Remember that for each vanishing state $s \in DR_V(G)$ we have $SJ(s) = 0$.

Thus, the steady-state PMF $\varphi = (\varphi(s_1), \ldots, \varphi(s_n))$ for $SMC(G)$ is

$$\varphi(s_i) = \begin{cases} \dfrac{\psi^*(s_i)SJ(s_i)}{\sum_{j=1}^{n} \psi^*(s_j)SJ(s_j)}, & s_i \in DR_T(G); \\ 0, & s_i \in DR_V(G). \end{cases}$$

Thus, to calculate φ, we apply abstraction from self-loops with probability less than 1 to get $\mathbf{P}^*$ and then ψ^*, followed by weighting by SJ and normalization. $EDTMC(G)$ has no self-loops with probability less than 1, unlike $SMC(G)$, hence, the behaviour of $EDTMC(G)$ may stabilize quicker than that of $SMC(G)$ (if each of them has a single steady state), since $\mathbf{P}^*$ has only zero (excepting the states having self-loops with probability 1) elements at the main diagonal.

Example 5.1. *Let E be from Example 3.6. In Figure 6, the underlying SMC $SMC(\overline{E})$ is presented. The average sojourn times in the states of the underlying SMC are written next to them in bold font.*

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

$$SJ = \left(\frac{1}{\rho}, \frac{1}{\chi}, 0, \frac{1}{\theta}, \frac{1}{\phi}\right).$$

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

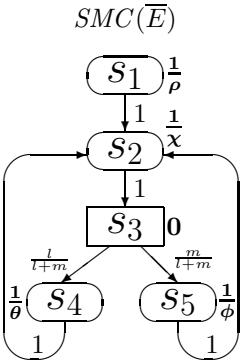

Figure 6: The underlying SMC of $\overline{E}$ for $E = [(\{a\}, \rho) * ((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta))[](((\{e\}, \natural_m); (\{f\}, \phi)))) * \mathsf{Stop}]$.

$$VAR = \left(\frac{1-\rho}{\rho^2}, \frac{1-\chi}{\chi^2}, 0, \frac{1-\theta}{\theta^2}, \frac{1-\phi}{\phi^2}\right).$$

The TPM for EDTMC($\overline{E}$) is

$$\mathbf{P}^* = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

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

$$\psi^* = \left(0, \frac{1}{3}, \frac{1}{3}, \frac{l}{3(l+m)}, \frac{m}{3(l+m)}\right).$$

The steady-state PMF ψ^* weighted by SJ is

$$\left(0, \frac{1}{3\chi}, 0, \frac{l}{3\theta(l+m)}, \frac{m}{3\phi(l+m)}\right).$$

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

$$\psi^* SJ^T = \frac{\theta\phi(l+m) + \chi(\phi l + \theta m)}{3\chi\theta\phi(l+m)}.$$

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

$$\varphi = \frac{1}{\theta\phi(l+m) + \chi(\phi l + \theta m)}(0, \theta\phi(l+m), 0, \chi\phi l, \chi\theta m).$$

In the case $l = m$ and $\theta = \phi$ we have

$$\varphi = \frac{1}{2(\chi+\theta)}(0, 2\theta, 0, \chi, \chi).$$

Let G be a dynamic expression and $s, \tilde{s} \in DR(G)$, $S, \widetilde{S} \subseteq DR(G)$. The following standard *performance indices (measures)* can be calculated based on the steady-state PMF φ for $SMC(G)$ and the average sojourn time vector SJ of G [83, 33, 58].

- The *average recurrence (return) time in the state s* (the number of discrete time units required for this) is $\frac{1}{\varphi(s)}$.

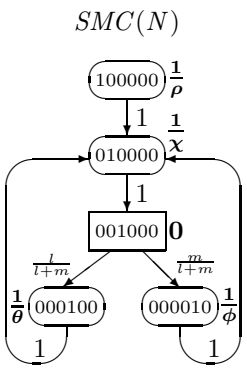

Figure 7: The underlying SMC of $N = Box_{dtsi}(\overline{E})$ for $E = [(\{a\}, \rho) * ((\{b\}, \chi); ((\{c\}, \natural_l); (\{d\}, \theta))[](\{e\}, \natural_m); (\{f\}, \phi)))) * \mathsf{Stop}]$.

- The *fraction of residence time in the state* s is $\varphi(s)$.
- The *fraction of residence time in the set of states* S or the *probability of the event determined by a condition that is true for all states from* S is $\sum_{s\in S} \varphi(s)$.
- The *relative fraction of residence time in the set of states* S *with respect to that in* $\widetilde{S}$ is $\frac{\sum_{s\in S} \varphi(s)}{\sum_{\tilde{s}\in\widetilde{S}} \varphi(\tilde{s})}$.
- The *rate of leaving the state* s is $\frac{\varphi(s)}{SJ(s)}$.
- The *steady-state probability to perform a step with a set of activities* Ξ is $\sum_{s\in DR(G)} \varphi(s) \sum_{\{\Upsilon|\Xi\subseteq\Upsilon\}} PT(\Upsilon, s)$.
- The *probability of the event determined by a reward function* r *on the states* is $\sum_{s\in DR(G)} \varphi(s) r(s)$, where $\forall s \in DR(G),\ 0 \le r(s) \le 1$.

Let $N = (P_N, T_N, W_N, \Omega_N, \mathcal{L}_N, M_N)$ be a LDTSIPN and $M, \widetilde{M} \in \boldsymbol{N}^{P_N}_{fin}$. Then the average sojourn time $SJ(M)$, the sojourn time variance $VAR(M)$, the probabilities $PM^*(M, \widetilde{M})$, the transition relation $M \twoheadrightarrow_{\mathcal{P}} \widetilde{M}$, 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.

As we have mentioned earlier, every marked plain dtsi-box could be interpreted as the LDTSIPN. Therefore, we can evaluate performance with the LDTSIPNs corresponding to dtsi-boxes and then transfer the results to the latter.

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.

Proposition 5.1. *For any static expression* E,

$$SMC(\overline{E}) \simeq SMC(Box_{dtsi}(\overline{E})).$$

Proof. By Theorem 4.2 and definitions of underlying SMCs for dynamic expressions and LDTSIPNs taking into account the following. First, for the associated SMCs, the average sojourn time in the states is the same, since it is defined via the analogous probability functions. Second, the transition probabilities of the associated SMCs are the sums of those belonging to transition systems or reachability graphs. □

Example 5.2. *Let* E *be from Example 3.6 and* $N = Box_{dtsi}(\overline{E})$. *In Figure 7, the underlying SMC* $SMC(N)$ *is presented. Clearly,* $SMC(\overline{E})$ *and* $SMC(N)$ *are isomorphic. Thus, both the transient and steady-state PMFs for* $SMC(N)$ *and* $SMC(\overline{E})$ *coincide.*

5.2. Analysis of the DTMC

Let us consider an alternative solution method, studying the DTMCs of expressions based on the state change probabilities $PM(s, \tilde{s})$.

Definition 5.2. Let G be a dynamic expression. The *discrete time Markov chain (DTMC)* of G, denoted by $DTMC(G)$, has the state space $DR(G)$, the initial state $[G]_{\approx}$ and the transitions $s \to_{\mathcal{P}} \tilde{s}$, where $\mathcal{P} = PM(s, \tilde{s})$.

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

One can see that $EDTMC(G)$ is constructed from $DTMC(G)$ as follows. For each state of $DTMC(G)$, we remove a possible self-loop with probability less than 1 associated with it and then normalize the probabilities of the remaining transitions from the state. Thus, $EDTMC(G)$ and $DTMC(G)$ differ only by existence of self-loops with probability less than 1 and magnitudes of the probabilities of the remaining transitions. Hence, $EDTMC(G)$ and $DTMC(G)$ have the same communication classes of states and $EDTMC(G)$ is irreducible iff $DTMC(G)$ is so. Since both $EDTMC(G)$ and $DTMC(G)$ are finite, they are positive recurrent. Thus, in case of irreducibility, each of them has a single stationary PMF. Note that $EDTMC(G)$ and/or $DTMC(G)$ may be periodic, thus having a unique stationary distribution, but no steady-state (limiting) one. For example, it may happen that $EDTMC(G)$ is periodic while $DTMC(G)$ is aperiodic due to self-loops associated with some states of the latter. The states of $SMC(G)$ are classified using $EDTMC(G)$, hence, $SMC(G)$ is irreducible (positive recurrent, aperiodic) iff $EDTMC(G)$ is so.

Let G be a dynamic expression. The elements $\mathcal{P}_{ij}$ $(1 \le i, j \le n = |DR(G)|)$ of (one-step) transition probability matrix (TPM) $\mathbf{P}$ for $DTMC(G)$ are defined as

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

The steady-state PMF ψ for $DTMC(G)$ is defined like the corresponding notion ψ^* for $EDTMC(G)$.

Let us determine a relationship between steady-state PMFs for $DTMC(G)$ and $EDTMC(G)$. The following theorem proposes the equation that relates the mentioned steady-state PMFs.

First, we introduce some helpful notation. For a vector $v = (v_1, \ldots, v_n)$, let $Diag(v)$ be a diagonal matrix of order n with the elements $Diag_{ij}(v)$ $(1 \le i, j \le n)$ defined as

$$Diag_{ij}(v) = \begin{cases} v_i, & i = j; \\ 0, & \text{otherwise.} \end{cases}$$

Theorem 5.1. *Let G be a dynamic expression and SL be its self-loops abstraction vector. Then the steady-state PMFs ψ for $DTMC(G)$ and ψ^* for $EDTMC(G)$ are related as follows: $\forall s \in DR(G)$,*

$$\psi(s) = \frac{\psi^*(s)SL(s)}{\sum\limits_{\tilde{s} \in DR(G)} \psi^*(\tilde{s})SL(\tilde{s})}.$$

Proof. Let there is an absorbing state $s_i \in DR(G)$ $(1 \le i \le n)$, i.e. $PM(s_i, s_i) = 1$. Then $\mathcal{P}_{ii} = 1$ in the TPM $\mathbf{P}$ of $DTMC(G)$ and $\mathcal{P}^*_{ii} = 1$ in the TPM $\mathbf{P}^*$ of $EDTMC(G)$, by definitions of those TPMs. We have earlier supposed that there exist no absorbing vanishing states, hence, $s_i \in DR_T(G)$ and $SL(s_i) = \infty = SJ(s_i)$. We have also supposed at most one single steady state in the considered Markov chains, hence, $\{s_i\}$ is a single communication (and ergodic) class of states in both $DTMC(G)$ and $EDTMC(G)$. Then $\psi(s_i) = 1 = \psi^*(s_i)$, whereas $\forall s \in DR(G) \setminus \{s_i\}$ $SL(s) < \infty$, $SJ(s) < \infty$ and $\psi(s) = 0 = \psi^*(s)$. We thus get for s_i

$$\frac{\psi^*(s_i)SL(s_i)}{\sum_{\tilde{s} \in DR(G)} \psi^*(\tilde{s})SL(\tilde{s})} = \frac{1 \cdot SL(s_i)}{1 \cdot SL(s_i)} = \frac{1}{1} = 1 = \psi(s_i),$$

whereas $\forall s \in DR(G) \setminus \{s_i\}$

$$\frac{\psi^*(s)SL(s)}{\sum_{\tilde{s} \in DR(G)} \psi^*(\tilde{s})SL(\tilde{s})} = \frac{0 \cdot SL(s)}{1 \cdot SL(s_i)} = \frac{0}{\infty} = 0 = \psi(s).$$

Let there are no absorbing states, i.e. $\forall s \in DR(G)\ PM(s,s) < 1$. Let PSL be a vector with the elements

$$PSL(s) = \left\{ \begin{array}{ll} PM(s,s), & s \to s; \\ 0, & \text{otherwise} \end{array} \right\} = 1 - \frac{1}{SL(s)}.$$

By definition of $PM^*(s,\tilde{s})$, we have $\mathbf{P}^* = Diag(SL)(\mathbf{P} - Diag(PSL)) = Diag(SL)(\mathbf{P} - \mathbf{I} + Diag(SL)^{-1}) = Diag(SL)(\mathbf{P} - \mathbf{I}) + \mathbf{I}$. Hence,

$$\mathbf{P}^* - \mathbf{I} = Diag(SL)(\mathbf{P} - \mathbf{I}).$$

Then $\psi^*(\mathbf{P}^* - \mathbf{I}) = \mathbf{0}$ implies

$$\psi^* Diag(SL)(\mathbf{P} - \mathbf{I}) = \mathbf{0}.$$

For $v = \psi^* Diag(SL)$, we get

$$v(\mathbf{P} - \mathbf{I}) = \mathbf{0}.$$

In order to calculate ψ on the basis of v, we must normalize it by dividing its elements by their sum, since we should have $\psi \mathbf{1}^T = 1$ as a result:

$$\psi = \frac{1}{v\mathbf{1}^T} v = \frac{1}{\psi^* Diag(SL)\mathbf{1}^T} \psi^* Diag(SL).$$

Thus, the elements of ψ are calculated as follows: $\forall s \in DR(G)$,

$$\psi(s) = \frac{\psi^*(s)SL(s)}{\sum_{\tilde{s} \in DR(G)} \psi^*(\tilde{s})SL(\tilde{s})}.$$

It is easy to check that ψ is a solution of the equation system

$$\left\{ \begin{array}{l} \psi(\mathbf{P} - \mathbf{I}) = \mathbf{0} \\ \psi \mathbf{1}^T = 1 \end{array} \right. ,$$

hence, it is indeed the steady-state PMF for $DTMC(G)$. □

The following proposition relates the steady-state PMFs for $SMC(G)$ and $DTMC(G)$.

Proposition 5.2. *Let G be a dynamic expression, φ be the steady-state PMF for $SMC(G)$ and ψ be the steady-state PMF for $DTMC(G)$. Then $\forall s \in DR(G)$,*

$$\varphi(s) = \left\{ \begin{array}{ll} \dfrac{\psi(s)}{\sum\limits_{\tilde{s} \in DR_T(G)} \psi(\tilde{s})}, & s \in DR_T(G); \\ 0, & s \in DR_V(G). \end{array} \right.$$

Proof. Let $s \in DR_T(G)$. Remember that $\forall s \in DR_T(G)$, $SL(s) = SJ(s)$ and $\forall s \in DR_V(G)$, $SJ(s) = 0$. Then, by Theorem 5.1, we have

$$\frac{\psi(s)}{\sum_{\tilde{s} \in DR_T(G)} \psi(\tilde{s})} = \frac{\frac{\psi^*(s)SL(s)}{\sum_{\tilde{s} \in DR(G)} \psi^*(\tilde{s})SL(\tilde{s})}}{\sum_{\tilde{s} \in DR_T(G)} \left(\frac{\psi^*(\tilde{s})SL(\tilde{s})}{\sum_{\check{s} \in DR(G)} \psi^*(\check{s})SL(\check{s})} \right)} = \frac{\psi^*(s)SL(s)}{\sum_{\tilde{s} \in DR(G)} \psi^*(\tilde{s})SL(\tilde{s})} \cdot \frac{\sum_{\check{s} \in DR(G)} \psi^*(\check{s})SL(\check{s})}{\sum_{\tilde{s} \in DR_T(G)} \psi^*(\tilde{s})SL(\tilde{s})} =$$

$$\frac{\psi^*(s)SL(s)}{\sum_{\tilde{s} \in DR_T(G)} \psi^*(\tilde{s})SL(\tilde{s})} = \frac{\psi^*(s)SJ(s)}{\sum_{\tilde{s} \in DR_T(G)} \psi^*(\tilde{s})SJ(\tilde{s})} = \frac{\psi^*(s)SJ(s)}{\sum_{\tilde{s} \in DR(G)} \psi^*(\tilde{s})SJ(\tilde{s})} = \varphi(s).$$

□

Thus, to calculate φ, one can only apply normalization to some elements of ψ (corresponding to the tangible states), instead of abstracting from self-loops with probability less than 1 to get $\mathbf{P}^*$ and then ψ^*, followed by weighting by *SJ* and normalization. Hence, using *DTMC*(G) instead of *EDTMC*(G) allows one to avoid multistage analysis, but the payment for it is more time-consuming numerical and more complex analytical calculation of ψ with respect to ψ^*. The reason is that *DTMC*(G) may self-loops with probability less that 1, unlike *EDTMC*(G), hence, the behaviour of *DTMC*(G) may stabilize slower than that of *EDTMC*(G) (if each of them has a single steady state) and $\mathbf{P}$ is potentially more dense matrix than $\mathbf{P}^*$, since $\mathbf{P}$ may have additional non-zero elements at the main diagonal. Nevertheless, Proposition 5.2 is very important, since the relationship between φ and ψ it discovers will be used in Proposition 5.3 to relate the steady-state PMFs for *SMC*(G) and the reduced *DTMC*(G), as well as in Section 8 to prove preservation of the stationary behaviour by a stochastic equivalence.

Example 5.3. *Let E be from Example 3.6. In Figure 8, the DTMC DTMC($\overline{E}$) is presented.*
The TPM for DTMC($\overline{E}$) is

$$\mathbf{P} = \begin{pmatrix} 1-\rho & \rho & 0 & 0 & 0 \\ 0 & 1-\chi & \chi & 0 & 0 \\ 0 & 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & \theta & 0 & 1-\theta & 0 \\ 0 & \phi & 0 & 0 & 1-\phi \end{pmatrix}.$$

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

$$\psi = \frac{1}{\theta\phi(1+\chi)(l+m)+\chi(\phi l+\theta m)}(0, \theta\phi(l+m), \chi\theta\phi(l+m), \chi\phi l, \chi\theta m).$$

Remember that $DR_T(\overline{E}) = \{s_1, s_2, s_4, s_5\}$ and $DR_V(\overline{E}) = \{s_3\}$. Hence,

$$\sum_{\tilde{s}\in DR_T(\overline{E})} \psi(\tilde{s}) = \psi(s_1)+\psi(s_2)+\psi(s_4)+\psi(s_5) = \frac{\theta\phi(l+m)+\chi(\phi l+\theta m)}{\theta\phi(1+\chi)(l+m)+\chi(\phi l+\theta m)}.$$

By Proposition 5.2, we have

$$\begin{aligned}
&\varphi(s_1) = 0 \cdot \tfrac{\theta\phi(1+\chi)(l+m)+\chi(\phi l+\theta m)}{\theta\phi(l+m)+\chi(\phi l+\theta m)} = 0,\\
&\varphi(s_2) = \tfrac{\theta\phi(l+m)}{\theta\phi(1+\chi)(l+m)+\chi(\phi l+\theta m)} \cdot \tfrac{\theta\phi(1+\chi)(l+m)+\chi(\phi l+\theta m)}{\theta\phi(l+m)+\chi(\phi l+\theta m)} = \tfrac{\theta\phi(l+m)}{\theta\phi(l+m)+\chi(\phi l+\theta m)},\\
&\varphi(s_3) = 0,\\
&\varphi(s_4) = \tfrac{\chi\phi l}{\theta\phi(1+\chi)(l+m)+\chi(\phi l+\theta m)} \cdot \tfrac{\theta\phi(1+\chi)(l+m)+\chi(\phi l+\theta m)}{\theta\phi(l+m)+\chi(\phi l+\theta m)} = \tfrac{\chi\phi l}{\theta\phi(l+m)+\chi(\phi l+\theta m)},\\
&\varphi(s_5) = \tfrac{\chi\theta m}{\theta\phi(1+\chi)(l+m)+\chi(\phi l+\theta m)} \cdot \tfrac{\theta\phi(1+\chi)(l+m)+\chi(\phi l+\theta m)}{\theta\phi(l+m)+\chi(\phi l+\theta m)} = \tfrac{\chi\theta m}{\theta\phi(l+m)+\chi(\phi l+\theta m)}.
\end{aligned}$$

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

$$\varphi = \frac{1}{\theta\phi(l+m)+\chi(\phi l+\theta m)}(0, \theta\phi(l+m), 0, \chi\phi l, \chi\theta m).$$

This coincides with the result obtained in Example 5.1 with the use of ψ^* and SJ.

5.3. Analysis of the reduced DTMC

Let us now consider the method from [33, 75, 5, 7, 6] that eliminates vanishing states from the EMC (EDTMC, in our terminology) corresponding to the underlying SMC of every GSPN N. The TPM for the resulting *reduced* EDTMC (REDTMC) has smaller size than that for the EDTMC. The method demonstrates that there exists a transformation of the underlying SMC of N into a CTMC, whose states are the tangible markings of N. This CTMC, which is essentially the *reduced* underlying SMC (RSMC) of N, is constructed on the basis of the REDTMC. The CTMC can then be directly solved to get both the transient and the steady-state PMFs over the tangible markings of N. In [33], the program and computational complexities of such an *elimination* method, based on the REDTMC, were evaluated and

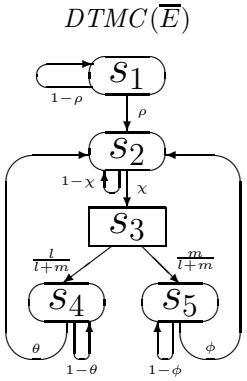

Figure 8: The DTMC of $\overline{E}$ for $E = [(\{a\}, \rho) * ((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta))[](\{\{e\}, \natural_m); (\{f\}, \phi)))) * \mathsf{Stop}]$.

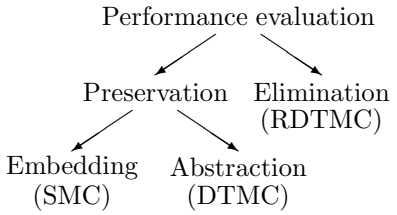

Figure 9: Performance evaluation methods in dtsiPBC

compared with those of the *preservation* method that does not eliminate vanishing states and based on the EDTMC. The preservation method for GSPNs corresponds in dtsiPBC to the analysis of the underlying SMCs of expressions (*embedding*). Note that the performance evaluation method based on the complete DTMCs of the dtsiPBC expressions (*abstraction*) has no prototype in the GSPN model, hence, it is a novel (alternative to embedding) variant of the preservation method. In Figure 9, a classification of the performance analysis techniques within dtsiPBC is presented.

The elimination method for GSPNs can be easily transferred to dtsiPBC, hence, for every dynamic expression G, we can find a DTMC (since the sojourn time in the tangible states from $DR(G)$ is discrete and geometrically distributed) with the states from $DR_T(G)$, which can be directly solved to find the transient and the steady-state PMFs over the tangible states. We shall demonstrate that such a *reduced* DTMC (RDTMC) of G, denoted by $RDTMC(G)$, can be constructed from $DTMC(G)$, using the method analogous to that designed in [75, 5, 7, 6] in the framework of GSPNs to transform EDTMC into REDTMC. Since the sojourn time in the vanishing states is zero, the state changes of $RDTMC(G)$ occur in the moments of the global discrete time associated with $SMC(G)$, unlike those of $EDTMC(G)$, which happen only when the current state changes to some *different* one, irrespective of the global time. Therefore, in our case, we can skip the stages of constructing the REDTMC of G, denoted by $REDTMC(G)$, from $EDTMC(G)$, and recovering RSMC of G, denoted by $RSMC(G)$ (which is the sought-for DTMC), from $REDTMC(G)$, since we shall have $RSMC(G) = RDTMC(G)$.

Let G be a dynamic expression and $\mathbf{P}$ be the TPM for $DTMC(G)$. We reorder the states from $DR(G)$ such that the first rows and columns of $\mathbf{P}$ will correspond to the states from $DR_V(G)$ and the last ones will correspond to the states from $DR_T(G)$. Let $|DR(G)| = n$ and $|DR_T(G)| = m$. The resulting matrix can be decomposed as follows:

$$\mathbf{P} = \begin{pmatrix} \mathbf{C} & \mathbf{D} \\ \mathbf{E} & \mathbf{F} \end{pmatrix}.$$

The elements of the $(n-m)\times(n-m)$ submatrix $\mathbf{C}$ are the probabilities to move from vanishing to vanishing states, and those of the $(n-m)\times m$ submatrix $\mathbf{D}$ are the probabilities to move from vanishing to tangible states. The elements of the $m\times(n-m)$ submatrix $\mathbf{E}$ are the probabilities to move from tangible to vanishing states, and those of the $m\times m$ submatrix $\mathbf{F}$ are the probabilities to move from tangible to tangible states.

The TPM $\mathbf{P}^\circ$ for $RDTMC(G)$ is the $m\times m$ matrix, calculated as

$$\mathbf{P}^\diamond = \mathbf{F} + \mathbf{EGD},$$

where the elements of the matrix $\mathbf{G}$ are the probabilities to move from vanishing to vanishing states in any number of state changes, without traversal of tangible states.

If there are no loops among vanishing states then for any vanishing state there exists a value $l \in \boldsymbol{N}$ such that every sequence of state changes that starts in a vanishing state and is longer than l should reach a tangible state. Thus, $\exists l \in \boldsymbol{N}\ \forall k > l\ \mathbf{C}^k = \mathbf{0}$ and $\sum_{k=0}^{\infty} \mathbf{C}^k = \sum_{k=0}^{l} \mathbf{C}^k$. If there are loops among vanishing states then all such loops are supposed to be of "transient" rather than "absorbing" type, since the latter is treated as a specification error to be corrected, like in [75, 6]. We have earlier required that $SMC(G)$ has a single closed communication (which is also ergodic) class of states. Remember that a communication class of states is their equivalence class w.r.t. communication relation, i.e. a maximal subset of communicating states. A communication class of states is closed if only the states belonging to it are accessible from every its state.

The ergodic class cannot consist of vanishing states only, to avoid "absorbing" loops among them, hence, it contains tangible states as well. Thus, any sequence of vanishing state changes that starts in the ergodic class will reach a tangible state at some time moment. All the states that do not belong to the ergodic class should be transient. Hence, any sequence of vanishing state changes that starts in a transient vanishing state will some time reach either a transient tangible state or a state from the ergodic class [62]. In the latter case, a tangible state will be reached as well, as argued above. Thus, every sequence of vanishing state changes in $SMC(G)$ that starts in a vanishing state will exit the set of all vanishing states in the future. This implies that the probabilities to move from vanishing to vanishing states in $k \in \boldsymbol{N}$ state changes, without traversal of tangible states, will lead to 0 when k tends to ∞. Then we have $\lim_{k\to\infty} \mathbf{C}^k = \lim_{k\to\infty}(\mathbf{I} - (\mathbf{I} - \mathbf{C}))^k = \mathbf{0}$, hence, $\mathbf{I} - \mathbf{C}$ is a non-singular matrix, i.e. its determinant is not equal to zero. Thus, the inverse matrix of $\mathbf{I} - \mathbf{C}$ exists and may be expressed by a Neumann series as $\sum_{k=0}^{\infty}(\mathbf{I} - (\mathbf{I} - \mathbf{C}))^k = \sum_{k=0}^{\infty} \mathbf{C}^k = (\mathbf{I} - \mathbf{C})^{-1}$. Therefore,

$$\mathbf{G} = \sum_{k=0}^{\infty} \mathbf{C}^k = \begin{cases} \sum_{k=0}^{l} \mathbf{C}^k, & \exists l \in \boldsymbol{N},\ \forall k > l,\ \mathbf{C}^k = \mathbf{0}, \quad \text{no loops among vanishing states;} \\ (\mathbf{I} - \mathbf{C})^{-1}, & \lim_{k\to\infty} \mathbf{C}^k = \mathbf{0}, \quad \text{loops among vanishing states;} \end{cases}$$

where $\mathbf{0}$ is the square matrix consisting only of zeros and $\mathbf{I}$ is the identity matrix, both of order $n - m$.

For $1 \le i, j \le m$ and $1 \le k, l \le n - m$, let $\mathcal{F}_{ij}$ be the elements of the matrix $\mathbf{F}$, $\mathcal{E}_{ik}$ be those of $\mathbf{E}$, $\mathcal{G}_{kl}$ be those of $\mathbf{G}$ and $\mathcal{D}_{lj}$ be those of $\mathbf{D}$. By definition, the elements $\mathcal{P}^\diamond_{ij}$ of the matrix $\mathbf{P}^\diamond$ are calculated as

$$\mathcal{P}^\diamond_{ij} = \mathcal{F}_{ij} + \sum_{k=1}^{n-m}\sum_{l=1}^{n-m} \mathcal{E}_{ik}\mathcal{G}_{kl}\mathcal{D}_{lj} = \mathcal{F}_{ij} + \sum_{k=1}^{n-m} \mathcal{E}_{ik} \sum_{l=1}^{n-m} \mathcal{G}_{kl}\mathcal{D}_{lj} = \mathcal{F}_{ij} + \sum_{l=1}^{n-m} \mathcal{D}_{lj} \sum_{k=1}^{n-m} \mathcal{E}_{ik}\mathcal{G}_{kl},$$

i.e. $\mathcal{P}^\diamond_{ij}$ $(1 \le i, j \le m)$ is the total probability to move from the tangible state s_i to the tangible state s_j in any number of steps, without traversal of tangible states, but possibly going through vanishing states.

Let $s, \tilde{s} \in DR_T(G)$ such that $s = s_i$, $\tilde{s} = s_j$. The *probability to move from s to $\tilde{s}$ in any number of steps, without traversal of tangible states* (if such a movement is possible, i.e. its probability is positive) is

$$PM^\diamond(s, \tilde{s}) = \mathcal{P}^\diamond_{ij}.$$

Definition 5.3. Let G be a dynamic expression and $[G]_\approx \in DR_T(G)$. The *reduced discrete time Markov chain (RDTMC)* of G, denoted by $RDTMC(G)$, has the state space $DR_T(G)$, the initial state $[G]_\approx$ and the transitions $s \hookrightarrow_{\mathcal{P}} \tilde{s}$, where $\mathcal{P} = PM^\diamond(s, \tilde{s})$.

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

Let us now try to define $RSMC(G)$ as a "restriction" of $SMC(G)$ to its tangible states. Since the sojourn time in the tangible states of $SMC(G)$ is discrete and geometrically distributed, we can see that $RSMC(G)$ is a DTMC with the state space $DR_T(G)$, the initial state $[G]_\approx$ and the transitions whose probabilities collect all those in $SMC(G)$ to move from the tangible to the tangible states, directly or indirectly, namely, by going through its vanishing states only. Thus, $RSMC(G)$ should have the transitions $s \hookrightarrow_{\mathcal{P}} \tilde{s}$, where $\mathcal{P} = PM^\diamond(s, \tilde{s})$, resulting in $RSMC(G) = RDTMC(G)$.

One can see that $RDTMC(G)$ is constructed from $DTMC(G)$ as follows. All vanishing states and all transitions to, from and between them are removed. All transitions between tangible states are preserved. The probabilities of

transitions between tangible states may become greater and new transitions between tangible states may be added, both iff there exist moves between these tangible states in any number of steps, going through vanishing states only. Thus, for each sequence of transitions between two tangible states in $DTMC(G)$ there exists a (possibly shorter, since the eventual passed through vanishing states are removed) sequence between the same states in $RDTMC(G)$ and vice versa. If $DTMC(G)$ is irreducible then all its states (including tangible ones) communicate, hence, all states of $RDTMC(G)$ communicate as well and it is irreducible. Since both $DTMC(G)$ and $RDTMC(G)$ are finite, they are positive recurrent. Thus, in case of irreducibility of $DTMC(G)$, each of them has a single stationary PMF. Note that $DTMC(G)$ and/or $RDTMC(G)$ may be periodic, thus having a unique stationary distribution, but no steady-state (limiting) one. For example, it may happen that $DTMC(G)$ is aperiodic while $RDTMC(G)$ is periodic due to removing vanishing states from the former.

Let $DR_T(G) = \{s_1, \ldots, s_m\}$ and $[G]_{\approx} \in DR_T(G)$. Then the transient ($k$-step, $k \in \boldsymbol{N}$) PMF $\psi^{\diamond}[k] = (\psi^{\diamond}[k](s_1), \ldots, \psi^{\diamond}[k](s_m))$ for $RDTMC(G)$ is calculated as

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

where $\psi^{\diamond}[0] = (\psi^{\diamond}[0](s_1), \ldots, \psi^{\diamond}[0](s_m))$ is the initial PMF defined as

$$\psi^{\diamond}[0](s_i) = \begin{cases} 1, & s_i = [G]_{\approx}; \\ 0, & \text{otherwise.} \end{cases}$$

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

The steady-state PMF $\psi^{\diamond} = (\psi^{\diamond}(s_1), \ldots, \psi^{\diamond}(s_m))$ for $RDTMC(G)$ is a solution of the equation system

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

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

Note that the vector $\psi^{\diamond}$ exists and is unique, if $RDTMC(G)$ is ergodic. Then $RDTMC(G)$ has a single steady state, and we have $\psi^{\diamond} = \lim_{k\to\infty} \psi^{\diamond}[k]$.

The zero sojourn times in the vanishing states guarantee that the state changes of $RDTMC(G)$ occur in the moments of the global discrete time associated with $SMC(G)$, i.e. every such state change occurs after one time unit delay. Hence, the sojourn time in the tangible states is the same for $RDTMC(G)$ and $SMC(G)$. The state change probabilities of $RDTMC(G)$ are those to move from tangible to tangible states in any number of steps, without traversal of tangible states. Therefore, $RDTMC(G)$ and $SMC(G)$ have the same transient behaviour over the tangible states, thus, the transient analysis of $SMC(G)$ is possible to accomplish using $RDTMC(G)$.

The following proposition relates the steady-state PMFs for $SMC(G)$ and $RDTMC(G)$. It proves that the steady-state probabilities of the tangible states coincide for them.

Proposition 5.3. *Let G be a dynamic expression, φ be the steady-state PMF for $SMC(G)$ and $\psi^{\diamond}$ be the steady-state PMF for $RDTMC(G)$. Then $\forall s \in DR(G)$,*

$$\varphi(s) = \begin{cases} \psi^{\diamond}(s), & s \in DR_T(G); \\ 0, & s \in DR_V(G). \end{cases}$$

Proof. To make the proof more clear, we use the following unified notation. $\mathbf{I}$ denotes the identity matrices of any size. $\mathbf{0}$ denotes square matrices and row vectors of any size and length of values 0. $\mathbf{1}$ denotes square matrices and row vectors of any size and length of values 1.

Let $\mathbf{P}$ be the reordered TPM for $DTMC(G)$ and ψ be the steady-state PMF for $DTMC(G)$, i.e. ψ is a solution of the equation system

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

Let $|DR(G)| = n$ and $|DR_T(G)| = m$. The decomposed $\mathbf{P}$, $\mathbf{P} - \mathbf{I}$ and ψ are

$$\mathbf{P} = \begin{pmatrix} \mathbf{C} & \mathbf{D} \\ \mathbf{E} & \mathbf{F} \end{pmatrix},\ \mathbf{P} - \mathbf{I} = \begin{pmatrix} \mathbf{C} - \mathbf{I} & \mathbf{D} \\ \mathbf{E} & \mathbf{F} - \mathbf{I} \end{pmatrix} \text{ and } \psi = (\psi_V, \psi_T),$$

where $\psi_V = (\psi_1, \ldots, \psi_{n-m})$ is the subvector of ψ with the steady-state probabilities of vanishing states and $\psi_T = (\psi_{n-m+1}, \ldots, \psi_n)$ is that with the steady-state probabilities of tangible states.

Then the equation system for ψ is decomposed as follows:

$$\begin{cases} \psi_V(\mathbf{C} - \mathbf{I}) + \psi_T\mathbf{E} = \mathbf{0} \\ \psi_V\mathbf{D} + \psi_T(\mathbf{F} - \mathbf{I}) = \mathbf{0} \\ \psi_V\mathbf{1}^T + \psi_T\mathbf{1}^T = 1 \end{cases}.$$

Further, let $\mathbf{P}^\diamond$ be the TPM for $RDTMC(G)$. Then $\psi^\diamond$ is a solution of the equation system

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

We have

$$\mathbf{P}^\diamond = \mathbf{F} + \mathbf{EGD},$$

where the matrix $\mathbf{G}$ can have two different forms, depending on whether the loops among vanishing states exist, hence, we consider the two following cases.

1. There exist *no loops among vanishing states*. We have $\exists l \in \mathbb{N},\ \forall k > l,\ \mathbf{C}^k = \mathbf{0}$ and $\mathbf{G} = \sum_{k=0}^{l} \mathbf{C}^k$.
   Let us right-multiply the first equation of the decomposed equation system for ψ by $\mathbf{G}$:

   $$\psi_V(\mathbf{CG} - \mathbf{G}) + \psi_T\mathbf{EG} = \mathbf{0}.$$

   Taking into account that $\mathbf{G} = \sum_{k=0}^{l} \mathbf{C}^k$, we get

   $$\psi_V\left(\sum_{k=1}^{l} \mathbf{C}^k + \mathbf{C}^{l+1} - \mathbf{C}^0 - \sum_{k=1}^{l} \mathbf{C}^k\right) + \psi_T\mathbf{EG} = \mathbf{0}.$$

   Since $\mathbf{C}^{l+1} = \mathbf{0}$ and $\mathbf{C}^0 = \mathbf{I}$, we obtain

   $$-\psi_V + \psi_T\mathbf{EG} = \mathbf{0} \text{ and } \psi_V = \psi_T\mathbf{EG}.$$

   Let us substitute ψ_V with $\psi_T\mathbf{EG}$ in the second equation of the decomposed equation system for ψ:

   $$\psi_T\mathbf{EGD} + \psi_T(\mathbf{F} - \mathbf{I}) = \mathbf{0} \text{ and } \psi_T(\mathbf{F} + \mathbf{EGD} - \mathbf{I}) = \mathbf{0}.$$

   Since $\mathbf{F} + \mathbf{EGD} = \mathbf{P}^\diamond$, we have

   $$\psi_T(\mathbf{P}^\diamond - \mathbf{I}) = \mathbf{0}.$$

2. There exist *loops among vanishing states*. We have $\lim_{k\to\infty} \mathbf{C}^k = \mathbf{0}$ and $\mathbf{G} = (\mathbf{I} - \mathbf{C})^{-1}$.
   Let us right-multiply the first equation of the decomposed equation system for ψ by $\mathbf{G}$:

   $$-\psi_V(\mathbf{I} - \mathbf{C})\mathbf{G} + \psi_T\mathbf{EG} = \mathbf{0}.$$

   Taking into account that $\mathbf{G} = (\mathbf{I} - \mathbf{C})^{-1}$, we get

   $$-\psi_V + \psi_T\mathbf{EG} = \mathbf{0} \text{ and } \psi_V = \psi_T\mathbf{EG}.$$

   Let us substitute ψ_V with $\psi_T\mathbf{EG}$ in the second equation of the decomposed equation system for ψ:

   $$\psi_T\mathbf{EGD} + \psi_T(\mathbf{F} - \mathbf{I}) = \mathbf{0} \text{ and } \psi_T(\mathbf{F} + \mathbf{EGD} - \mathbf{I}) = \mathbf{0}.$$

   Since $\mathbf{F} + \mathbf{EGD} = \mathbf{P}^\diamond$, we have

   $$\psi_T(\mathbf{P}^\diamond - \mathbf{I}) = \mathbf{0}.$$

The third equation $\psi_V \mathbf{1}^T + \psi_T \mathbf{1}^T = 1$ of the decomposed equation system for ψ implies that if ψ_V has non-zero elements then the sum of the elements of ψ_T is less than one. We normalize ψ_T by dividing its elements by their sum:

$$v = \frac{1}{\psi_T \mathbf{1}^T} \psi_T.$$

It is easy to check that v is a solution of the equation system

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

hence, it is the steady-state PMF for $RDTMC(G)$ and we have

$$\psi^\diamond = v = \frac{1}{\psi_T \mathbf{1}^T} \psi_T.$$

Note that $\forall s \in DR_T(G),\ \psi_T(s) = \psi(s)$. Then the elements of $\psi^\diamond$ are calculated as follows: $\forall s \in DR_T(G)$,

$$\psi^\diamond(s) = \frac{\psi_T(s)}{\sum_{\tilde{s} \in DR_T(G)} \psi_T(\tilde{s})} = \frac{\psi(s)}{\sum_{\tilde{s} \in DR_T(G)} \psi(\tilde{s})}.$$

By Proposition 5.2, $\forall s \in DR_T(G),\ \varphi(s) = \frac{\psi(s)}{\sum_{\tilde{s} \in DR_T(G)} \psi(\tilde{s})}$.
Therefore, $\forall s \in DR_T(G)$,

$$\varphi(s) = \frac{\psi(s)}{\sum_{\tilde{s} \in DR_T(G)} \psi(\tilde{s})} = \psi^\diamond(s).$$

□

Thus, to calculate φ, one can just take all the elements of $\psi^\diamond$ as the steady-state probabilities of the tangible states, instead of abstracting from self-loops with probability less than 1 to get $\mathbf{P}^*$ and then ψ^*, followed by weighting by SJ and normalization. Hence, using $RDTMC(G)$ instead of $EDTMC(G)$ allows one to avoid such a multistage analysis, but constructing $\mathbf{P}^\diamond$ also requires some efforts, including calculating matrix powers or inverse matrices. Note that $RDTMC(G)$ may have self-loops with probability less than 1, unlike $EDTMC(G)$, hence, the behaviour of $RDTMC(G)$ may stabilize slower than that of $EDTMC(G)$ (if each of them has a single steady state). On the other hand, $\mathbf{P}^\diamond$ is generally smaller and denser matrix than $\mathbf{P}^*$, since $\mathbf{P}^\diamond$ may have additional non-zero elements not only at the main diagonal, but also many of them outside it. Therefore, in most cases, we have less time-consuming numerical calculation of $\psi^\diamond$ with respect to ψ^*. At the same time, the complexity of the analytical calculation of $\psi^\diamond$ with respect to ψ^* depends on the model structure, such as the number of vanishing states and loops among them, but usually it is lower, since the matrix size reduction plays an important role in many cases. Hence, for the system models with many immediate activities we normally have a significant simplification of the solution. At the abstraction level of SMCs, the elimination of vanishing states decreases their impact to the solution complexity while allowing immediate activities to specify a comprehensible logical structure of systems at the higher level of transition systems.

Example 5.4. *Let E be from Example 3.6. Remember that $DR_T(\overline{E}) = \{s_1, s_2, s_4, s_5\}$ and $DR_V(\overline{E}) = \{s_3\}$. We reorder the states from $DR(\overline{E})$, by moving the vanishing states to the first positions: s_3, s_1, s_2, s_4, s_5.*

The reordered TPM for $DTMC(\overline{E})$ is

$$\mathbf{P}_r = \begin{pmatrix} 0 & 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & 1-\rho & \rho & 0 & 0 \\ \chi & 0 & 1-\chi & 0 & 0 \\ 0 & 0 & \theta & 1-\theta & 0 \\ 0 & 0 & \phi & 0 & 1-\phi \end{pmatrix}.$$

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

$RDTMC(\overline{E})$

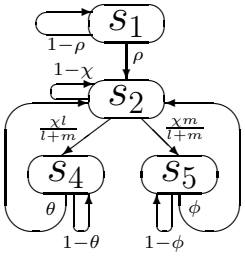

Figure 10: The reduced DTMC of $\overline{E}$ for $E = [(\{a\}, \rho) * ((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta))[](\{e\}, \natural_m)); (\{f\}, \phi)))) * \mathsf{Stop}]$.

$$\mathbf{C} = 0,\ \mathbf{D} = \left(0, 0, \frac{l}{l+m}, \frac{m}{l+m}\right),\ \mathbf{E} = \begin{pmatrix} 0 \\ \chi \\ 0 \\ 0 \end{pmatrix},\ \mathbf{F} = \begin{pmatrix} 1-\rho & \rho & 0 & 0 \\ 0 & 1-\chi & 0 & 0 \\ 0 & \theta & 1-\theta & 0 \\ 0 & \phi & 0 & 1-\phi \end{pmatrix}.$$

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

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

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

$$\mathbf{P}^\diamond = \mathbf{F} + \mathbf{EGD} = \mathbf{F} + \mathbf{EID} = \mathbf{F} + \mathbf{ED} = \begin{pmatrix} 1-\rho & \rho & 0 & 0 \\ 0 & 1-\chi & \frac{\chi l}{l+m} & \frac{\chi m}{l+m} \\ 0 & \theta & 1-\theta & 0 \\ 0 & \phi & 0 & 1-\phi \end{pmatrix}.$$

In Figure 10, the reduced DTMC $RDTMC(\overline{E})$ *is presented. The steady-state PMF for* $RDTMC(\overline{E})$ *is*

$$\psi^\diamond = \frac{1}{\theta\phi(l+m) + \chi(\phi l + \theta m)}(0, \theta\phi(l+m), \chi\phi l, \chi\theta m).$$

Note that $\psi^\diamond = (\psi^\diamond(s_1), \psi^\diamond(s_2), \psi^\diamond(s_4), \psi^\diamond(s_5))$*. By Proposition 5.3, we have*

$$\begin{aligned} \varphi(s_1) &= 0, \\ \varphi(s_2) &= \tfrac{\theta\phi(l+m)}{\theta\phi(l+m)+\chi(\phi l+\theta m)}, \\ \varphi(s_3) &= 0, \\ \varphi(s_4) &= \tfrac{\chi\phi l}{\theta\phi(l+m)+\chi(\phi l+\theta m)}, \\ \varphi(s_5) &= \tfrac{\chi\theta m}{\theta\phi(l+m)+\chi(\phi l+\theta m)}. \end{aligned}$$

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

$$\varphi = \frac{1}{\theta\phi(l+m) + \chi(\phi l + \theta m)}(0, \theta\phi(l+m), 0, \chi\phi l, \chi\theta m).$$

This coincides with the result obtained in Example 5.1 with the use of ψ^* *and SJ.*

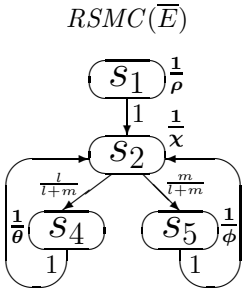

Figure 11: The reduced SMC of $\overline{E}$ for $E = [(\{a\}, \rho) * (((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta))[](((\{e\}, \natural_m); (\{f\}, \phi)))) * \mathsf{Stop}]$.

Example 5.5. *Let E be from Example 3.6. In Figure 11, the reduced underlying SMC RSMC($\overline{E}$) is depicted. The average sojourn times in the states of the reduced underlying SMC are written next to them in bold font. In spite of the equality RSMC($\overline{E}$) = RDTMC($\overline{E}$), the graphical representation of RSMC($\overline{E}$) differs from that of RDTMC($\overline{E}$), since the former is based on the REDTMC($\overline{E}$), where each state is decorated with the* positive *average sojourn time of RSMC($\overline{E}$) in it. REDTMC($\overline{E}$) is constructed from EDTMC($\overline{E}$) in the similar way as RDTMC($\overline{E}$) is obtained from DTMC($\overline{E}$). By construction, the residence time in each state of RSMC($\overline{E}$) is geometrically distributed. Hence, the associated parameter of geometrical distribution is uniquely recovered from the average sojourn time in the state.*

Let us now formally prove that RSMC coincides with RDTMC. Although this assertion is very intuitive, its proof is rather involved. The relation between *DTMC* and *RDTMC* is obtained using the transition function $PM^\circ(s, \tilde{s})$, based on $PM(s, \tilde{s})$. The relation between *RDTMC* and the embedded *RDTMC* (*ERDTMC*) is obtained using the transition function $(PM^\circ)^*(s, \tilde{s})$, based on $PM^\circ(s, \tilde{s})$. The relation between *EDTMC* and the reduced *EDTMC* (*REDTMC*) is obtained using the transition function $(PM^*)^\circ(s, \tilde{s})$, based on $PM^*(s, \tilde{s})$. Let G be a dynamic expression. We shall prove that the TPM $(\mathbf{P}^\circ)^*$ for the embedded *RDTMC*(G) (*ERDTMC*(G)), (forwardly) constructed by reduction (eliminating vanishing states) of *DTMC*(G), followed by embedding *ERDTMC*(G) into *RDTMC*(G), *coincides with* the (finally) embedded TPM $((\mathbf{P}^*)^\circ)^*$, (reversely) constructed by embedding *EDTMC*(G) into *SMC*(G), followed by reduction *REDTMC*(G) of *EDTMC*(G), and final embedding *EREDTMC*(G) into *RSMC*(G). The final embedding in the reverse construction is needed, since new self-loops may arise after reducing *EDTMC*(G), i.e. *REDTMC*(G) may become not an EDTMC, but a DTMC featuring self-loops with probability less than 1. Note that for $s, \tilde{s} \in DR_T(G)$, we have $(PM^\circ)^*(s, \tilde{s}) = SL^\circ(s)PM^\circ(s, \tilde{s})$ in *ERDTMC*(G). Here $SL^\circ(s)$ is the self-loops abstraction factor in s in *RDTMC*(G). This corresponds to a different expression $(PM^*)^\circ(s, \tilde{s}) = (SL \cdot PM)^\circ(s, \tilde{s})$ in *REDTMC*(G). In particular, $SL^\circ(s) > SL(s)$ when $PM^\circ(s, s) > PM(s, s)$, which is the reason for a new self-loop associated with s in *RDTMC*(G). As we shall see, in that case $(PM^\circ)^*(s, \tilde{s}) > (PM^*)^\circ(s, \tilde{s})$. The following theorem relates those finally embedded reduced embedded TPM $((\mathbf{P}^*)^\circ)^*$ (i.e. the TPM for *EREDTMC*(G)) and embedded reduced TPM $(\mathbf{P}^\circ)^*$ (the TPM for *ERDTMC*(G)).

Theorem 5.2. *Let G be a dynamic expression, $(\mathbf{P}^\circ)^*$ results from embedding the TPM $\mathbf{P}^\circ$ for RDTMC(G), and $((\mathbf{P}^*)^\circ)^*$ results from reduction and final embedding the TPM $\mathbf{P}^*$ for EDTMC(G). Then*

$$((\mathbf{P}^*)^\circ)^* = (\mathbf{P}^\circ)^*.$$

Proof. See Appendix Appendix A.2. □

Thus, reduction before embedding is more optimal computationally for DTMCs of the process expressions, since only one embedding is needed in that case. This is especially important when the DTMCs have many loops from tangible states via (one or more) vanishing states only. Such loops remain after the first embedding, and they become self-loops after the subsequent reduction, which are removed just after the second embedding.

In the continuous time setting, the embedding of the EDTMC into an SMC with only exponential and zero sojourn times is not hard to construct. Such SMCs resemble CTMCs, in that their behaviour is not changed at presence of self-loops (the average sojourn times are kept). The EDTMC transition probabilities are obtained by normalizing the rates of the transitions from each state to the different ones. Only the embedding of the reduced EDTMC into the RSMC

(actually being a CTMC) requires more computations. In our discrete time setting, we cannot simply abstract from self-loops, since each of them takes one time unit and thereby influences the average sojourn time. Then avoiding the additional embedding allows us to decrease the computation costs.

We can now explain the inequality presented above Theorem 5.2, by using its proof. Let $s, \tilde{s} \in DR_T(G)$. Then $SL_F(s)SL_{H'}(s) = SL^\diamond(s) > SL(s) = SL_F(s)$ implies $SL_{H'}(s) > 1$. Thus, $(PM^\diamond)^*(s,\tilde{s}) = SL^\diamond(s)PM^\diamond(s,\tilde{s}) = SL_F(s)SL_{H'}(s)PM^\diamond(s,\tilde{s}) > SL_F(s)PM^\diamond(s,\tilde{s}) = (PM^*)^\diamond(s,\tilde{s})$.

Definition 5.4. Let G be a dynamic expression, $s \in DR_T(G)$ while $SL_F(s)$ is the self-loops abstraction factor in s for the submatrix $\mathbf{F}$ (from the equation $\mathbf{P}^\diamond = \mathbf{F} + \mathbf{EGD}$ calculating the TPM for $RDTMC(G)$) and $SL_{H'}(s)$ is the self-loops abstraction factor in s in $REDTMC(G)$ (for the matrix $\mathbf{H}' = Diag(SL_F)\mathbf{EGD}$, whose elements are the probabilities to move from tangible to tangible states, via any *positive* number of vanishing states, without traversal of tangible states, in $EDTMC(G)$).

The *reduced SMC (RSMC)* of G, denoted by $RSMC(G)$, has the EDTMC $EREDTMC(G)$ and the sojourn time in every $s \in DR_T(G)$ is geometrically distributed with the parameter $\frac{1}{SL_F(s)SL_{H'}(s)}$.

The following proposition demonstrates coincidence of RSMC and RDTMC.

Proposition 5.4. *Let G be a dynamic expression. Then $RSMC(G) = RDTMC(G)$.*

Proof. By Theorem 5.2, $EREDTMC(G) = ERDTMC(G)$. The sojourn time in every $s \in DR_T(G)$ is geometrically distributed with the parameter $\frac{1}{SL_F(s)SL_{H'}(s)} = \frac{1}{SL^\diamond(s)} = 1 - PM^\diamond(s,s) = 1 - PM_F(s,s) - PM_H(s,s)$, where $SL_{H'}(s) = \frac{1}{1-SL_F(s)PM_H(s,s)}$. Here $PM_H(s,s)$ is the self-loop probability in s for the matrix $\mathbf{H} = \mathbf{EGD}$ (from the equation $\mathbf{P}^\diamond = \mathbf{F} + \mathbf{EGD}$ calculating the TPM for $RDTMC(G)$). Remember that $SL^\diamond(s)$ is the self-loops abstraction factor in s in $RDTMC(G)$. Hence, $RSMC(G) = RDTMC(G)$. □

Example 5.6. *Let E be from Example 3.6. The TPMs for $RDTMC(\overline{E})$ and $ERDTMC(\overline{E})$ are*

$$\mathbf{P}^\diamond = \begin{pmatrix} 1-\rho & \rho & 0 & 0 \\ 0 & 1-\chi & \frac{\chi l}{l+m} & \frac{\chi m}{l+m} \\ 0 & \theta & 1-\theta & 0 \\ 0 & \phi & 0 & 1-\phi \end{pmatrix}, \ (\mathbf{P}^\diamond)^* = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

The TPMs for $REDTMC(\overline{E})$ and $EREDTMC(\overline{E})$ are

$$(\mathbf{P}^*)^\diamond = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \ ((\mathbf{P}^*)^\diamond)^* = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

The self-loops abstraction subvector of $\overline{E}$ for the submatrix $\mathbf{F}$ (see Example 5.4) is $SL_F = \left(\frac{1}{\rho}, \frac{1}{\chi}, \frac{1}{\theta}, \frac{1}{\phi}\right)$. The self-loops abstraction vector of $\overline{E}$ in $REDTMC(\overline{E})$ (for the matrix $\mathbf{H}'$, see below) is $(SL^*)^\diamond = SL_{H'} = (1,1,1,1)$. The self-loops abstraction vector of $\overline{E}$ in $RDTMC(\overline{E})$ is $SL^\diamond = \mathbf{1}Diag(SL_F)Diag(SL_{H'}) = \left(\frac{1}{\rho}, \frac{1}{\chi}, \frac{1}{\theta}, \frac{1}{\phi}\right)$, where $\mathbf{1}$ is a row vector of n values 1.

The elements of the matrix $\mathbf{H}'$ are the probabilities to move from tangible to tangible states, via any positive *number of vanishing states, without traversal of tangible states, in $EDTMC(G)$. We have $\mathbf{H}' = Diag(SL_F)\mathbf{H}$, where elements of the matrix $\mathbf{H} = \mathbf{EGD}$ (see Example 5.4) are the probabilities to move from tangible to tangible states, via any* positive *number of vanishing states, without traversal of tangible states, in $DTMC(G)$. The matrices $\mathbf{H}$ and $\mathbf{H}'$ are*

$$\mathbf{H} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\chi l}{l+m} & \frac{\chi m}{l+m} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \ \mathbf{H}' = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Then it is easy to check that

$$((\mathbf{P}^*)^\diamond)^* = Diag(SL_{H'})Diag(SL_F)(\mathbf{P}^\diamond - \mathbf{I}) + \mathbf{I} = Diag(SL^\diamond)(\mathbf{P}^\diamond - \mathbf{I}) + \mathbf{I} = (\mathbf{P}^\diamond)^*.$$

Note that our reduction of the underlying SMC by eliminating its vanishing states, resulting in the reduced DTMC, resembles the method from [70] that removes instantaneous states of stochastically discontinuous Markov reward chains. The latter are "limits" of continuous time Markov chains with state rewards and fast transitions when the rates (speeds) of these transitions tend to infinity, making them immediate. By analogy with that work, we would consider DTMCs extended with instantaneous states instead of SMCs with geometrically distributed or zero sojourn times in the states. However, within dtsiPBC, we have decided to take SMCs as the underlying stochastic process to be able in the perspective to consider not only geometrically distributed and zero residence time in the states, but arbitrary fixed time delays as well.

6. Stochastic equivalences

Consider the expressions $E = (\{a\}, \frac{1}{2})$ and $E' = (\{a\}, \frac{1}{3})_1[](\{a\}, \frac{1}{3})_2$, for which $\overline{E} \neq_{ts} \overline{E'}$, since $TS(\overline{E})$ has only one transition from the initial to the final state (with probability $\frac{1}{2}$) while $TS(\overline{E'})$ has two such ones (with probabilities $\frac{1}{4}$). On the other hand, all the mentioned transitions are labeled by activities with the same multiaction part $\{a\}$. Moreover, the overall probabilities of the mentioned transitions of $TS(\overline{E})$ and $TS(\overline{E'})$ coincide: $\frac{1}{2} = \frac{1}{4} + \frac{1}{4}$. Further, $TS(\overline{E})$ (as well as $TS(\overline{E'})$) has one empty loop transition from the initial state to itself with probability $\frac{1}{2}$ and one empty loop transition from the final state to itself with probability 1. The empty loop transitions are labeled by the empty set of activities. For calculating the transition probabilities of $TS(\overline{E'})$, take $\rho = \chi = \frac{1}{3}$ in Example 3.5. Then you will see that the probability parts $\frac{1}{3}$ and $\frac{1}{3}$ of the activities $(\{a\}, \frac{1}{3})_1$ and $(\{a\}, \frac{1}{3})_2$ are "splitted" among probabilities $\frac{1}{4}$ and $\frac{1}{4}$ of the corresponding transitions and the probability $\frac{1}{2}$ of the empty loop transition. Unlike $=_{ts}$, most of the probabilistic and stochastic equivalences proposed in the literature do not differentiate between the processes such as those specified by E and E'. In Figure 13(a), the marked dtsi-boxes corresponding to the dynamic expressions $\overline{E}$ and $\overline{E'}$ are presented, i.e. $N = Box_{dtsi}(\overline{E})$ and $N' = Box_{dtsi}(\overline{E'})$.

Since the semantic equivalence $=_{ts}$ is too discriminating in many cases, 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.

6.1. 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.

Thus, we follow the approaches of [56, 64, 50, 52, 14, 10, 11], 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 [56], in contrast to the reactive model, treated in [64], and we take transition probabilities instead of transition rates from [50, 52, 14, 10, 11]. Thus, step stochastic bisimulation equivalence that we define further is (in the probabilistic sense) comparable only with interleaving probabilistic bisimulation one from [56], and our equivalence is obviously stronger.

In the definition below, we consider $\mathcal{L}(\Upsilon) \in \boldsymbol{N}^{\mathcal{L}}_{fin}$ for $\Upsilon \in \boldsymbol{N}^{SI\mathcal{L}}_{fin}$, 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 $A \in \boldsymbol{N}^{\mathcal{L}}_{fin}$, 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 $\exists \mathcal{P}\ s \xrightarrow{A}_{\mathcal{P}} \tilde{s}$.

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

Definition 6.1. Let G and G' be dynamic expressions. An *equivalence* relation $\mathcal{R} \subseteq (DR(G) \cup DR(G'))^2$ is a *step stochastic bisimulation* between G and G', denoted by $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$, if:

1. $([G]_{\approx}, [G']_{\approx}) \in \mathcal{R}$.
2. $(s_1, s_2) \in \mathcal{R} \ \Rightarrow\ \forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}},\ \forall A \in \boldsymbol{N}^{\mathcal{L}}_{fin}$,

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

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

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

Definition 6.2. 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

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

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

The multiaction transition systems of static expressions can be defined as well. For $E \in RegStatExpr$ let $TS_{\mathcal{L}}(E) = TS_{\mathcal{L}}(\overline{E})$.

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 [56].

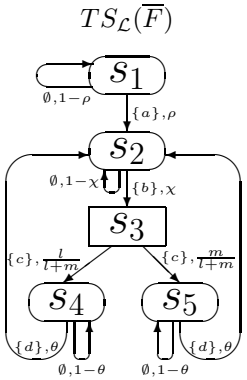

Figure 12: The multiaction transition system of $\overline{F}$ for $F = [(\{a\}, \rho) * ((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta)_1)[]((\{c\}, \natural_m); (\{d\}, \theta)_2))) * \mathsf{Stop}]$.

Example 6.1. *Let us consider an abstraction F of the static expression E from Example 3.6, such that $c = e$, $d = f$, $\theta = \phi$, i.e.*

$$F = [(\{a\}, \rho) * ((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta)_1)[]((\{c\}, \natural_m); (\{d\}, \theta)_2))) * \mathsf{Stop}].$$

Then $DR(\overline{F}) = \{s_1, s_2, s_3, s_4, s_5\}$ is obtained from $DR(\overline{E})$ via substitution of the symbols e, f, ϕ by c, d, θ, respectively, in the specifications of the corresponding states from the latter set. We have $DR_T(\overline{F}) = \{s_1, s_2, s_4, s_5\}$ and $DR_V(\overline{F}) = \{s_3\}$. In Figure 12, the multiaction transition system $TS_{\mathcal{L}}(\overline{F})$ is presented. For simplicity of the graphical representation, the singleton multisets of multiactions are written without outer braces.

The following proposition states that every step stochastic bisimulation binds tangible states only with tangible ones and the same is valid for vanishing states.

Proposition 6.1. *Let G and G' be dynamic expressions and $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$. Then*

$$\mathcal{R} \subseteq (DR_T(G) \cup DR_T(G'))^2 \uplus (DR_V(G) \cup DR_V(G'))^2.$$

Proof. By definition of transition systems of expressions, for every tangible state, there is an empty loop from it, and no empty loop transitions are possible from vanishing states.

Further, $\mathcal{R}$ preserves empty loops. To verify this fact, first take $A = \emptyset$ in its definition to get $\forall (s_1, s_2) \in \mathcal{R}$, $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$, $s_1 \xrightarrow{\emptyset}_{\mathcal{P}} \mathcal{H} \;\Leftrightarrow\; s_2 \xrightarrow{\emptyset}_{\mathcal{P}} \mathcal{H}$, and then observe that the empty loop transition from a state leads only to the same state. □

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

Proof. See Appendix A.3. □

In [3], an algorithm for strong probabilistic bisimulation on labeled probabilistic transition systems (a reformulation of probabilistic automata) was proposed with time complexity $O(n^2 m)$, where n is the number of states and m is the number of transitions. In [4], a decision algorithm for strong probabilistic bisimulation on generative labeled probabilistic transition systems was constructed with time complexity $O(m \log n)$ and space complexity $O(m + n)$. In [31], a polynomial algorithm for strong probabilistic bisimulation on probabilistic automata was presented. The mentioned algorithms for interleaving probabilistic bisimulation equivalence can be adapted for $\underline{\leftrightarrow}_{ss}$ using the method from [55], applied to get the decidability results for step bisimulation equivalence. The method takes into account that transition systems in interleaving and step semantics differ only by availability of the additional transitions corresponding to parallel execution of activities in the latter (which is our case).

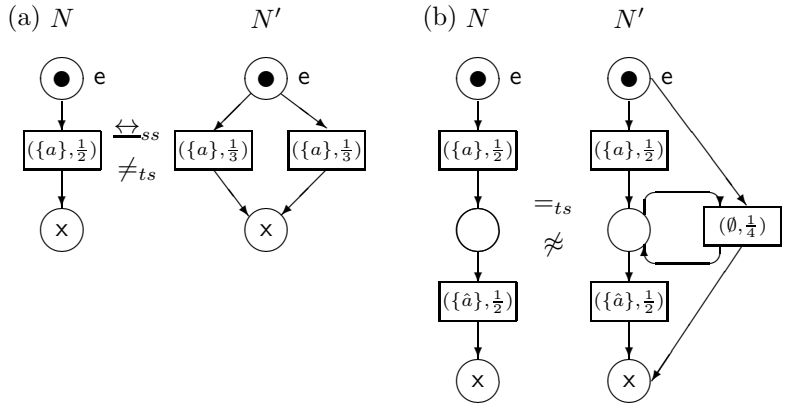

Figure 13: Dtsi-boxes of the dynamic expressions from equivalence examples of Theorem 6.1.

6.2. Interrelations of the stochastic equivalences

We now compare the discrimination power of the stochastic equivalences.

Theorem 6.1. *For dynamic expressions G and G' the following* strict *implications hold:*

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

Proof. Let us check the validity of the implications.

- The implication $=_{ts} \Rightarrow \underline{\leftrightarrow}_{ss}$ is proved as follows. Let $\beta : G =_{ts} G'$. Then it is easy to see that $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$, where $\mathcal{R} = \{(s, \beta(s)) \mid s \in DR(G)\}$.
- The implication $\approx \Rightarrow =_{ts}$ is valid, since the transition system of a dynamic formula is defined based on its structural equivalence class.

Let us see that that the implications are strict, i.e. the reverse ones do not work, by the following counterexamples.

(a) Let $E = (\{a\}, \frac{1}{2})$ and $E' = (\{a\}, \frac{1}{3})_1 [] (\{a\}, \frac{1}{3})_2$. Then $\overline{E} \underline{\leftrightarrow}_{ss} \overline{E'}$, but $\overline{E} \neq_{ts} \overline{E'}$, since $TS(\overline{E})$ has only one transition from the initial to the final state while $TS(\overline{E'})$ has two such ones.

(b) Let $E = (\{a\}, \frac{1}{2}); (\{\hat{a}\}, \frac{1}{2})$ and $E' = ((\{a\}, \frac{1}{2}); (\{\hat{a}\}, \frac{1}{2}))$ sy a. Then $\overline{E} =_{ts} \overline{E'}$, but $\overline{E} \not\approx \overline{E'}$, since $\overline{E}$ and $\overline{E'}$ cannot be reached from each other by applying inaction rules. □

Example 6.2. *In Figure 13, the marked dtsi-boxes corresponding to the dynamic expressions from equivalence examples of Theorem 6.1 are presented, i.e. $N = Box_{dtsi}(\overline{E})$ and $N' = Box_{dtsi}(\overline{E'})$ for each picture (a)–(b).*

7. Reduction modulo equivalences

The equivalences which we proposed can be used to reduce transition systems and SMCs of expressions (reachability graphs and SMCs of dtsi-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 properties. Thus, the reduction allows one to simplify the behavioural and performance analysis of systems.

7.1. Quotients of the transition systems and Markov chains

We now construct the quotient (by $\underline{\leftrightarrow}_{ss}$) transition systems and Markov chains (SMCs, DTMCs and RDTMCs).

An *autobisimulation* is a bisimulation between an expression and itself. For a dynamic expression G and a step stochastic autobisimulation on it $\mathcal{R} : G\underline{\leftrightarrow}_{ss}G$, let $\mathcal{K} \in DR(G)/_{\mathcal{R}}$ and $s_1, s_2 \in \mathcal{K}$. We have $\forall\widetilde{\mathcal{K}} \in DR(G)/_{\mathcal{R}}$, $\forall A \in \boldsymbol{N}^{\mathcal{L}}_{fin}$, $s_1 \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{K}} \Leftrightarrow s_2 \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{K}}$. The previous equality is valid for all $s_1, s_2 \in \mathcal{K}$, hence, we can rewrite it as $\mathcal{K} \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{K}}$, 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 $\mathcal{K} \xrightarrow{A} \widetilde{\mathcal{K}}$ if $\exists\mathcal{P}$, $\mathcal{K} \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{K}}$ and $\mathcal{K} \rightarrow \widetilde{\mathcal{K}}$ if $\exists A$, $\mathcal{K} \xrightarrow{A} \widetilde{\mathcal{K}}$. The similar arguments allow us to write $\mathcal{K} \rightarrow_{\mathcal{P}} \widetilde{\mathcal{K}}$, where $\mathcal{P} = PM(\mathcal{K}, \widetilde{\mathcal{K}}) = PM(s_1, \widetilde{\mathcal{K}}) = PM(s_2, \widetilde{\mathcal{K}})$.

By Proposition 6.1, $\mathcal{R} \subseteq (DR_T(G))^2 \uplus (DR_V(G))^2$. 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 6.2, $\mathcal{R}_{ss}(G)$ is the largest step stochastic autobisimulation on G. Based on the equivalence classes with respect to $\mathcal{R}_{ss}(G)$, the quotient (by $\underline{\leftrightarrow}_{ss}$) transition systems and the quotient (by $\underline{\leftrightarrow}_{ss}$) underlying 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 7.1. 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

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

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

The quotient (by $\underline{\leftrightarrow}_{ss}$) transition systems of static expressions can be defined as well. For $E \in RegStatExpr$, let $TS_{\underline{\leftrightarrow}_{ss}}(E) = TS_{\underline{\leftrightarrow}_{ss}}(\overline{E})$.

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 [56]. It is clear that 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 $TS_{\underline{\leftrightarrow}_{ss}}(G)$.

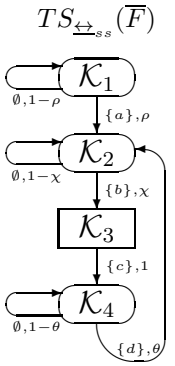

Figure 14: The quotient transition system of $\overline{F}$ for $F = [(\{a\}, \rho) * (((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta)_1)[]((\{c\}, \natural_m); (\{d\}, \theta)_2))) * \mathsf{Stop}]$.

Example 7.1. *Let F be from Example 6.1. Then $DR(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4\}$, where $\mathcal{K}_1 = \{s_1\}$, $\mathcal{K}_2 = \{s_2\}$, $\mathcal{K}_3 = \{s_3\}$, $\mathcal{K}_4 = \{s_4, s_5\}$. We also have $DR_T(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_4\}$ and $DR_V(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_3\}$. In Figure 14, the quotient transition system $TS_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is presented.*

The *quotient (by $\underline{\leftrightarrow}_{ss}$) average sojourn time vector* of G is defined as $SJ_{\underline{\leftrightarrow}_{ss}} = SJ_{\mathcal{R}_{ss}(G)}$.
The *quotient (by $\underline{\leftrightarrow}_{ss}$) sojourn time variance vector* of G is defined as $VAR_{\underline{\leftrightarrow}_{ss}} = VAR_{\mathcal{R}_{ss}(G)}$.

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 set of activities after possible self-loops* is

$$PM^*(\mathcal{K}, \widetilde{\mathcal{K}}) = \begin{cases} PM(\mathcal{K}, \widetilde{\mathcal{K}}) \sum_{k=0}^{\infty} PM(\mathcal{K}, \mathcal{K})^k = \frac{PM(\mathcal{K}, \widetilde{\mathcal{K}})}{1-PM(\mathcal{K}, \mathcal{K})}, & \mathcal{K} \to \mathcal{K}; \\ PM(\mathcal{K}, \widetilde{\mathcal{K}}), & \text{otherwise.} \end{cases}$$

The value $k = 0$ in the summation above corresponds to the case when no self-loops occur. Note that $\forall \mathcal{K} \in DR_T(G)/_{\mathcal{R}_{ss}(G)}$ with $PM(\mathcal{K}, \mathcal{K}) < 1$, it holds $PM^*(\mathcal{K}, \widetilde{\mathcal{K}}) = SJ_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) PM(\mathcal{K}, \widetilde{\mathcal{K}})$, since we always have the empty loop (which is a self-loop) $\mathcal{K} \xrightarrow{\emptyset} \mathcal{K}$ from every equivalence class of tangible states $\mathcal{K}$. Empty loops are not possible from equivalence classes of vanishing states, hence, $\forall \mathcal{K} \in DR_V(G)/_{\mathcal{R}_{ss}(G)}$ with $PM(\mathcal{K}, \mathcal{K}) < 1$, it holds $PM^*(\mathcal{K}, \widetilde{\mathcal{K}}) = \frac{PM(\mathcal{K}, \widetilde{\mathcal{K}})}{1-PM(\mathcal{K}, \mathcal{K})}$, when there are non-empty self-loops (produced by iteration) from $\mathcal{K}$, or $PM^*(\mathcal{K}, \widetilde{\mathcal{K}}) = PM(\mathcal{K}, \widetilde{\mathcal{K}})$, when there are no self-loops from $\mathcal{K}$.

Definition 7.2. 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 quotient (by $\underline{\leftrightarrow}_{ss}$) underlying SMCs of static expressions can be defined as well. For $E \in RegStatExpr$, let $SMC_{\underline{\leftrightarrow}_{ss}}(E) = SMC_{\underline{\leftrightarrow}_{ss}}(\overline{E})$.

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 the corresponding notions ψ^* for $EDTMC(G)$ and φ for $SMC(G)$, respectively.

Example 7.2. *Let F be from Example 6.1. In Figure 15, the quotient underlying SMC $SMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is presented. The average sojourn times in the states of the quotient underlying SMC are written next to them in bold font.*

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

$$SJ_{\underline{\leftrightarrow}_{ss}} = \left(\frac{1}{\rho}, \frac{1}{\chi}, 0, \frac{1}{\theta}\right).$$

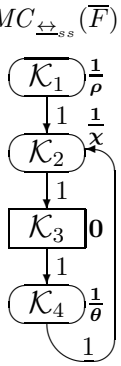

Figure 15: The quotient underlying SMC of $\overline{F}$ for $F = [(\{a\}, \rho) * (((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta)_1)[](\{c\}, \natural_m); (\{d\}, \theta)_2))) * \mathsf{Stop}]$.

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

$$VAR_{\underline{\leftrightarrow}_{ss}} = \left(\frac{1-\rho}{\rho^2}, \frac{1-\chi}{\chi^2}, 0, \frac{1-\theta}{\theta^2}\right).$$

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

$$\mathbf{P}^*_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

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

$$\psi^*_{\underline{\leftrightarrow}_{ss}} = \left(0, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right).$$

The steady-state PMF $\psi^*_{\underline{\leftrightarrow}_{ss}}$ *weighted by* $SJ_{\underline{\leftrightarrow}_{ss}}$ *is*

$$\left(0, \frac{1}{3\chi}, 0, \frac{l}{3\theta}\right).$$

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

$$\psi^*_{\underline{\leftrightarrow}_{ss}} SJ^T_{\underline{\leftrightarrow}_{ss}} = \frac{\chi + \theta}{3\chi\theta}.$$

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

$$\varphi_{\underline{\leftrightarrow}_{ss}} = \frac{1}{\chi + \theta}(0, \theta, 0, \chi).$$

The quotients of both transition systems and underlying SMCs are the minimal reductions of these 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. Such reduction method resembles that from [2] based on place bisimulation equivalence for PNs, excepting that the former merges states, while the latter merges places.

Moreover, the algorithms exist to construct the quotients of transition systems by an equivalence (like bisimulation one) [85] and those of (discrete or continuous time) Markov chains by ordinary lumping [37]. The algorithms have time complexity $O(m \log n)$ and space complexity $O(m + n)$, where n is the number of states and m is the number of transitions. As mentioned in [103], the algorithm from [37] can be easily adjusted to produce quotients of labeled

probabilistic transition systems by the probabilistic bisimulation equivalence. In [103], the symbolic partition refinement algorithm on state space of CTMCs was proposed. The algorithm can be straightforwardly accommodated to DTMCs, interactive MCs, Markov reward models, Markov decision processes, Kripke structures and labeled probabilistic transition systems. Such a symbolic lumping uses memory efficiently due to compact representation of the state space partition. The symbolic lumping is time efficient, since fast algorithm of the partition representation and refinement is applied. In [38], a polynomial-time algorithm for minimizing behaviour of probabilistic automata by probabilistic bisimulation equivalence was outlined that results in the canonical quotient structures. One could adapt the above algorithms for our framework of transition systems, (reduced) DTMCs and SMCs.

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

Definition 7.3. 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 $\mathcal{K} \to_{\mathcal{P}} \widetilde{\mathcal{K}}$, where $\mathcal{P} = PM(\mathcal{K}, \widetilde{\mathcal{K}})$.

The quotient (by $\underline{\leftrightarrow}_{ss}$) DTMCs of static expressions can be defined as well. For $E \in RegStatExpr$, let $DTMC_{\underline{\leftrightarrow}_{ss}}(E) = DTMC_{\underline{\leftrightarrow}_{ss}}(\overline{E})$.

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

Example 7.3. *Let F be from Example 6.1. In Figure 16, the quotient DTMC $DTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is presented.*

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

$$\mathbf{P}_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 1-\rho & \rho & 0 & 0 \\ 0 & 1-\chi & \chi & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \theta & 0 & 1-\theta \end{pmatrix}.$$

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

$$\psi_{\underline{\leftrightarrow}_{ss}} = \frac{1}{\chi + \theta + \chi\theta}(0, \theta, \chi\theta, \chi).$$

Remember that $DR_T(\overline{F})/_{\mathcal{R}_{ss}(F)} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_4\}$ and $DR_V(\overline{F})/_{\mathcal{R}_{ss}(F)} = \{\mathcal{K}_3\}$. Hence,

$$\sum_{\mathcal{K} \in DR_T(\overline{F})/_{\mathcal{R}_{ss}(F)}} \psi(\mathcal{K}) = \psi(\mathcal{K}_1) + \psi(\mathcal{K}_2) + \psi(\mathcal{K}_4) = \frac{\chi + \theta}{\chi + \theta + \chi\theta}.$$

By the "quotient" analogue of Proposition 5.2, we have

$$\begin{aligned} &\varphi_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_1) = 0 \cdot \tfrac{\chi+\theta+\chi\theta}{\chi+\theta} = 0, \\ &\varphi_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_2) = \tfrac{\theta}{\chi+\theta+\chi\theta} \cdot \tfrac{\chi+\theta+\chi\theta}{\chi+\theta} = \tfrac{\theta}{\chi+\theta}, \\ &\varphi_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_3) = 0, \\ &\varphi_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_4) = \tfrac{\chi}{\chi+\theta+\chi\theta} \cdot \tfrac{\chi+\theta+\chi\theta}{\chi+\theta} = \tfrac{\chi}{\chi+\theta}. \end{aligned}$$

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

$$\varphi_{\underline{\leftrightarrow}_{ss}} = \frac{1}{\chi + \theta}(0, \theta, 0, \chi).$$

This coincides with the result obtained in Example 7.2 with the use of $\psi^*_{\underline{\leftrightarrow}_{ss}}$ and $SJ_{\underline{\leftrightarrow}_{ss}}$.

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

Definition 7.4. The *reduced quotient (by $\underline{\leftrightarrow}_{ss}$) DTMC* of G, $RDTMC_{\underline{\leftrightarrow}_{ss}}(G)$, is defined like $RDTMC(G)$ in Section 5, but it is constructed from $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ instead of $DTMC(G)$.

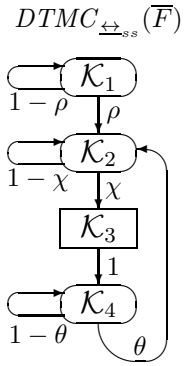

Figure 16: The quotient DTMC of $\overline{F}$ for $F = [(\{a\},\rho) * (((\{b\},\chi);((((\{c\},\natural_l);(\{d\},\theta)_1)[](\{c\},\natural_m));(\{d\},\theta)_2))) * \mathsf{Stop}]$.

The reduced quotient (by $\underline{\leftrightarrow}_{ss}$) DTMCs of static expressions can be defined as well. For $E \in RegStatExpr$, let $RDTMC_{\underline{\leftrightarrow}_{ss}}(E) = RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{E})$.

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

Example 7.4. *Let F be from Example 6.1. Remember that $DR_T(\overline{F})/_{\mathcal{R}_{ss}(F)} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_4\}$ and $DR_V(\overline{F})/_{\mathcal{R}_{ss}(F)} = \{\mathcal{K}_3\}$. We reorder the states from $DR(\overline{F})/_{\mathcal{R}_{ss}(F)}$, by moving vanishing states to the first positions: $\mathcal{K}_3, \mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_4$.*

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

$$\mathbf{P}_{r\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 1-\rho & \rho & 0 \\ \chi & 0 & 1-\chi & 0 \\ 0 & 0 & \theta & 1-\theta \end{pmatrix}.$$

The result of the decomposing $\mathbf{P}_{r\underline{\leftrightarrow}_{ss}}$ are the matrices

$$\mathbf{C}_{\underline{\leftrightarrow}_{ss}} = 0,\ \mathbf{D}_{\underline{\leftrightarrow}_{ss}} = (0,0,1),\ \mathbf{E}_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 0 \\ \chi \\ 0 \end{pmatrix},\ \mathbf{F}_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 1-\rho & \rho & 0 \\ 0 & 1-\chi & 0 \\ 0 & \theta & 1-\theta \end{pmatrix}.$$

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

$$\mathbf{G}_{\underline{\leftrightarrow}_{ss}} = \sum_{k=0}^{l} \mathbf{C}^k_{\underline{\leftrightarrow}_{ss}} = \mathbf{C}^0_{\underline{\leftrightarrow}_{ss}} = \mathbf{I}.$$

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

$$\mathbf{P}^{\diamond}_{\underline{\leftrightarrow}_{ss}} = \mathbf{F}_{\underline{\leftrightarrow}_{ss}} + \mathbf{E}_{\underline{\leftrightarrow}_{ss}}\mathbf{G}_{\underline{\leftrightarrow}_{ss}}\mathbf{D}_{\underline{\leftrightarrow}_{ss}} = \mathbf{F}_{\underline{\leftrightarrow}_{ss}} + \mathbf{E}_{\underline{\leftrightarrow}_{ss}}\mathbf{I}\mathbf{D}_{\underline{\leftrightarrow}_{ss}} = \mathbf{F}_{\underline{\leftrightarrow}_{ss}} + \mathbf{E}_{\underline{\leftrightarrow}_{ss}}\mathbf{D}_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 1-\rho & \rho & 0 \\ 0 & 1-\chi & \chi \\ 0 & \theta & 1-\theta \end{pmatrix}.$$

Let F be from Example 6.1. In Figure 17, the reduced quotient DTMC $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is presented. The steady-state PMF for $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is

$$\psi^{\diamond}_{\underline{\leftrightarrow}_{ss}} = \frac{1}{\chi+\theta}(0,\theta,\chi).$$

Note that $\psi^{\diamond}_{\underline{\leftrightarrow}_{ss}} = (\psi^{\diamond}_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_1), \psi^{\diamond}_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_2), \psi^{\diamond}_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_4))$. By the "quotient" analogue of Proposition 5.3, we have

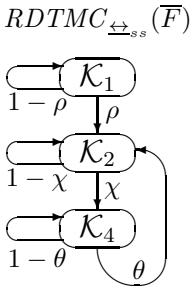

Figure 17: The reduced quotient DTMC of $\overline{F}$ for $F = [(\{a\}, \rho) * ((\{b\}, \chi); ((((\{c\}, \natural_l); (\{d\}, \theta)_1)[](((\{c\}, \natural_m); (\{d\}, \theta)_2))) * \mathsf{Stop}]$.

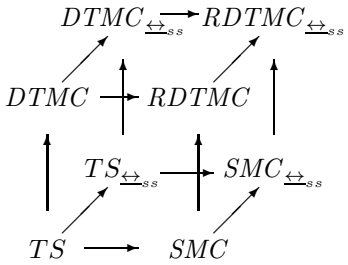

Figure 18: The cube of interrelations for the standard and quotient transition systems and Markov chains of the process expressions.

$$\begin{aligned}
&\varphi_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_1) = 0,\\
&\varphi_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_2) = \tfrac{\theta}{\chi+\theta},\\
&\varphi_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_3) = 0,\\
&\varphi_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}_4) = \tfrac{\chi}{\chi+\theta}.
\end{aligned}$$

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

$$\varphi_{\underline{\leftrightarrow}_{ss}} = \frac{1}{\chi+\theta}(0, \theta, 0, \chi).$$

This coincides with the result obtained in Example 7.2 with the use of $\psi^*_{\underline{\leftrightarrow}_{ss}}$ *and* $SJ_{\underline{\leftrightarrow}_{ss}}$.

Obviously, 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 $\psi^\circ_{\underline{\leftrightarrow}_{ss}}$, are the same as those determined between their "non-quotient" versions in Theorem 5.1, Proposition 5.2 and Proposition 5.3, respectively.

7.2. Interrelations of the standard and quotient behavioural structures

In Figure 18, the cube of interconnections by the relation "constructed from" is depicted for both standard and quotient transition systems and Markov chains (SMCs, DTMCs and RDTMCs) of the process expressions. The relations between *SMC* and $SMC_{\underline{\leftrightarrow}_{ss}}$, between *DTMC* and $DTMC_{\underline{\leftrightarrow}_{ss}}$, as well as between *RDTMC* and $RDTMC_{\underline{\leftrightarrow}_{ss}}$, can be obtained using the following corresponding transition functions, defined by analogy with those already introduced: $PM^*(\mathcal{K}, \widetilde{\mathcal{K}})$, based on $PM^*(s, \tilde{s})$, then $PM(\mathcal{K}, \widetilde{\mathcal{K}})$, based on $PM(s, \tilde{s})$, as well as $PM^\circ(\mathcal{K}, \widetilde{\mathcal{K}})$, based on $PM^\circ(s, \tilde{s})$ (all that to be proved below).

The relations between *SMC* and *RDTMC*, as well as between $SMC_{\underline{\leftrightarrow}_{ss}}$ and $RDTMC_{\underline{\leftrightarrow}_{ss}}$, can be obtained using the following corresponding transition functions: $PM^\circ(s, \tilde{s})$, based on $PM^*(s, \tilde{s})$, through $(PM^\circ)^*(s, \tilde{s})$, as well as $PM^\circ(\mathcal{K}, \widetilde{\mathcal{K}})$, based on $PM^*(\mathcal{K}, \widetilde{\mathcal{K}})$, through $(PM^\circ)^*(\mathcal{K}, \widetilde{\mathcal{K}})$ (by Theorem 5.2 and its "quotient" analogue).

In Figure 18, the relation (depicted by arrow) between *DTMC* and $DTMC_{\underline{\leftrightarrow}_{ss}}$ is obtained using the transition function $PM(\mathcal{K}, \widetilde{\mathcal{K}})$, based on $PM(s, \tilde{s})$. Let G be a dynamic expression. We shall prove that the (quotient) TPM $\mathbf{P}_{\underline{\leftrightarrow}_{ss}}$

for $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$, (forwardly) constructed by quotienting (by $\underline{\leftrightarrow}_{ss}$) $TS(G)$, followed by extracting $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ from $TS_{\underline{\leftrightarrow}_{ss}}(G)$, *coincides with* the TPM $(\mathbf{P})_{\underline{\leftrightarrow}_{ss}}$, (reversely) constructed by extracting $DTMC(G)$ from $TS(G)$, followed by quotienting $DTMC(G)$. The following proposition relates those quotient extracted TPM $(\mathbf{P})_{\underline{\leftrightarrow}_{ss}}$ and extracted quotient TPM $\mathbf{P}_{\underline{\leftrightarrow}_{ss}}$.

Proposition 7.1. *Let G be a dynamic expression, $\mathbf{P}_{\underline{\leftrightarrow}_{ss}}$ be the TPM for $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ and $(\mathbf{P})_{\underline{\leftrightarrow}_{ss}}$ results from quotienting (by $\underline{\leftrightarrow}_{ss}$) the TPM $\mathbf{P}$ for $DTMC(G)$. Then*

$$(\mathbf{P})_{\underline{\leftrightarrow}_{ss}} = \mathbf{P}_{\underline{\leftrightarrow}_{ss}}.$$

Proof. Let $\mathcal{K}, \widetilde{\mathcal{K}} \in DR(G)/_{\mathcal{R}_{ss}(G)}$ and $s \in \mathcal{K}$.

In $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$, we have $\sum_{A \in \mathbf{N}^{\mathcal{L}}_{fin}} PM_A(\mathcal{K}, \widetilde{\mathcal{K}}) = \sum_{A \in \mathbf{N}^{\mathcal{L}}_{fin}} PM_A(s, \widetilde{\mathcal{K}}) = \sum_{A \in \mathbf{N}^{\mathcal{L}}_{fin}} \sum_{\{\Upsilon \mid \exists \tilde{s} \in \widetilde{\mathcal{K}}\ s \xrightarrow{\Upsilon} \tilde{s},\ \mathcal{L}(\Upsilon) = A\}} PT(\Upsilon, s) = \sum_{\{\Upsilon \mid \exists \tilde{s} \in \widetilde{\mathcal{K}}\ s \xrightarrow{\Upsilon} \tilde{s}\}} PT(\Upsilon, s) = PM(s, \widetilde{\mathcal{K}}) = PM(\mathcal{K}, \widetilde{\mathcal{K}})$.

In the quotient of $DTMC(G)$, we have $\sum_{\tilde{s} \in \widetilde{\mathcal{K}}} PM(s, \tilde{s}) = \sum_{\tilde{s} \in \widetilde{\mathcal{K}}} \sum_{\{\Upsilon \mid s \xrightarrow{\Upsilon} \tilde{s}\}} PT(\Upsilon, s) = \sum_{\{\Upsilon \mid \exists \tilde{s} \in \widetilde{\mathcal{K}}\ s \xrightarrow{\Upsilon} \tilde{s}\}} PT(\Upsilon, s) = PM(s, \widetilde{\mathcal{K}}) = PM(\mathcal{K}, \widetilde{\mathcal{K}})$.

Thus, $(\mathbf{P})_{\underline{\leftrightarrow}_{ss}} = \mathbf{P}_{\underline{\leftrightarrow}_{ss}}$. □

Hence, the quotienting and extraction are permutable for transition systems of the process expressions. Applying extraction before the quotienting is useful to start from the level of Markov chains in the proofs.

Example 7.5. *Let F be from Example 6.1. The TPMs for $DTMC(\overline{F})$ and $DTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ are*

$$\mathbf{P} = \begin{pmatrix} 1-\rho & \rho & 0 & 0 & 0 \\ 0 & 1-\chi & \chi & 0 & 0 \\ 0 & 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & \theta & 0 & 1-\theta & 0 \\ 0 & \theta & 0 & 0 & 1-\theta \end{pmatrix}, \quad \mathbf{P}_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 1-\rho & \rho & 0 & 0 \\ 0 & 1-\chi & \chi & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \theta & 0 & 1-\theta \end{pmatrix}.$$

The TPM for the quotient of $DTMC(\overline{F})$ is

$$(\mathbf{P})_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 1-\rho & \rho & 0 & 0 \\ 0 & 1-\chi & \chi & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \theta & 0 & 1-\theta \end{pmatrix}.$$

Then it is clear that

$$(\mathbf{P})_{\underline{\leftrightarrow}_{ss}} = \mathbf{P}_{\underline{\leftrightarrow}_{ss}}.$$

In Figure 18, the relation (depicted by arrow) between SMC and $SMC_{\underline{\leftrightarrow}_{ss}}$ is obtained using the transition function $PM^*(\mathcal{K}, \widetilde{\mathcal{K}})$, based on $PM^*(s, \tilde{s})$. Let G be a dynamic expression. We shall prove that the (quotient) TPM $\mathbf{P}^*_{\underline{\leftrightarrow}_{ss}}$ for $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$, (forwardly) constructed by quotienting (by $\underline{\leftrightarrow}_{ss}$) $DTMC(G)$, followed by embedding $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$ into $SMC_{\underline{\leftrightarrow}_{ss}}(G)$, *coincides with* the (finally) embedded TPM $(\mathbf{P}^*)^*_{\underline{\leftrightarrow}_{ss}}$, (reversely) constructed by embedding $EDTMC(G)$ into $SMC(G)$, followed by quotienting $EDTMC(G)$, and final embedding a new EDTMC $EDTMC'(G)$ into the quotient of $EDTMC(G)$. The final embedding in the reverse construction is needed, since new self-loops may arise after quotienting $EDTMC(G)$, i.e. it may become not an EDTMC, but a DTMC featuring self-loops with probability less than 1. Note that for $\mathcal{K} \in DR(G)/_{\mathcal{R}_{ss}(G)}$ and $s \in \mathcal{K}$, we have $PM^*(\mathcal{K}, \widetilde{\mathcal{K}}) = SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) PM(\mathcal{K}, \widetilde{\mathcal{K}}) = SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) PM(s, \widetilde{\mathcal{K}})$ in $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$. This corresponds to a different expression $\sum_{\tilde{s} \in \widetilde{\mathcal{K}}} PM^*(s, \tilde{s}) = \sum_{\tilde{s} \in \widetilde{\mathcal{K}}} SL(s) PM(s, \tilde{s}) = SL(s) \sum_{\tilde{s} \in \widetilde{\mathcal{K}}} PM(s, \tilde{s}) = SL(s) PM(s, \widetilde{\mathcal{K}})$ in the quotient of $EDTMC(G)$. In particular, $SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) > SL(s)$ when $PM(s, \mathcal{K} \setminus \{s\}) > 0$, which is the reason for a new self-loop associated with s in the quotient of $EDTMC(G)$. The following proposition relates those finally embedded quotient embedded TPM $(\mathbf{P}^*)^*_{\underline{\leftrightarrow}_{ss}}$ (i.e. the TPM for $EDTMC'(G)$) and embedded quotient TPM $\mathbf{P}^*_{\underline{\leftrightarrow}_{ss}}$.

Proposition 7.2. *Let G be a dynamic expression, $\mathbf{P}^*_{\underline{\leftrightarrow}_{ss}}$ be the TPM for $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$ and $(\mathbf{P}^*)^*_{\underline{\leftrightarrow}_{ss}}$ results from quotienting (by $\underline{\leftrightarrow}_{ss}$) and final embedding the TPM $\mathbf{P}^*$ for $EDTMC(G)$. Then*

$$(\mathbf{P}^*)^*_{\underline{\leftrightarrow}_{ss}} = \mathbf{P}^*_{\underline{\leftrightarrow}_{ss}}.$$

Proof. See Appendix A.4. □

Thus, the quotienting before embedding is more optimal computationally for DTMCs of the process expressions, since only one embedding is needed in that case. This is especially important when the DTMCs have many loops with two or more equivalent states. Such loops remain after the first embedding, and they become self-loops after the subsequent quotienting, which are removed just after the second embedding.

By Proposition 7.2, $EDTMC'(G) = EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$. The sojourn time in every $\mathcal{K} \in DR_T(G)/_{\mathcal{R}_{ss}(G)}$ is geometrically distributed with the parameter $\frac{1}{SL(s)SL'(s,\mathcal{K})} = \frac{1}{SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K})}$, where $SL'(s,\mathcal{K}) = \frac{1}{1-SL(s)PM(s,\mathcal{K}\setminus\{s\})}$, while the sojourn time in every $\mathcal{K} \in DR_V(G)/_{\mathcal{R}_{ss}(G)}$ is equal to 0. Here $SL'(s,\mathcal{K})$ is the self-loops abstraction factor in the equivalence class $\mathcal{K}$ with respect to the state $s \in \mathcal{K}$ for the quotient of $EDTMC(G)$. Hence, $SMC'(G) = SMC_{\underline{\leftrightarrow}_{ss}}(G)$, where $SMC'(G)$ is the SMC with the EDTMC $EDTMC'(G)$, such that $\frac{1}{SL(s)SL'(s,\mathcal{K})}$ is the geometrical distribution parameter of the sojourn time in every $\mathcal{K} \in DR_T(G)/_{\mathcal{R}_{ss}(G)}$ while the sojourn time is zero in every $\mathcal{K} \in DR_V(G)/_{\mathcal{R}_{ss}(G)}$.

Example 7.6. *Let F be from Example 6.1. The TPMs for $EDTMC(\overline{F})$ and $EDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ are*

$$\mathbf{P}^* = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}, \ \mathbf{P}^*_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

The TPMs for the quotient of $EDTMC(\overline{F})$ and EDTMC of the quotient of $EDTMC(\overline{F})$ ($EDTMC'(\overline{F})$) are

$$(\mathbf{P}^*)_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \ (\mathbf{P}^*)^*_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

Then it is clear that

$$(\mathbf{P}^*)^*_{\underline{\leftrightarrow}_{ss}} = \mathbf{P}^*_{\underline{\leftrightarrow}_{ss}}.$$

Let G be a dynamic expression. We now construct the quotient (by $\underline{\leftrightarrow}_{ss}$) of the TPM for $DTMC(G)$ using special collector and distributor matrices. Let $DR(G) = \{s_1, \ldots, s_n\}$ and $DR(G)/_{\mathcal{R}_{ss}(G)} = \{\mathcal{K}_1, \ldots, \mathcal{K}_l\}$.

The elements $(\mathcal{P}_{\underline{\leftrightarrow}_{ss}})_{rs}$ $(1 \le r, s \le l)$ of the TPM $\mathbf{P}_{\underline{\leftrightarrow}_{ss}}$ for $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ are defined as

$$(\mathcal{P}_{\underline{\leftrightarrow}_{ss}})_{rs} = \begin{cases} PM(\mathcal{K}_r, \mathcal{K}_s), & \mathcal{K}_r \to \mathcal{K}_s; \\ 0, & \text{otherwise.} \end{cases}$$

Like it has been done for strong performance bisimulation on labeled CTSPNs in [28], the $l \times l$ TPM $\mathbf{P}_{\underline{\leftrightarrow}_{ss}}$ for $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ can be constructed from the $n \times n$ TPM $\mathbf{P}$ for $DTMC(G)$ using the $n \times l$ *collector matrix* $\mathbf{V}$ for the largest step stochastic autobisimulation $\mathcal{R}_{ss}(G)$ on G and the $l \times n$ *distributor matrix* $\mathbf{W}$ for $\mathbf{V}$. Then $\mathbf{W}$ should be a non-negative matrix (i.e. all its elements must be non-negative) with the elements of each its row summed to one, such that $\mathbf{WV} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix of order l, i.e. $\mathbf{W}$ is a *left-inverse matrix* for $\mathbf{V}$. It is known that for each collector matrix there is at least one distributor matrix, in particular, the matrix obtained by transposing $\mathbf{V}$ and subsequent normalizing its rows, to guarantee that the elements of each row of the transposed matrix are summed to one. We now present the formal definitions.

The elements $\mathcal{V}_{ir}$ $(1 \le i \le n,\ 1 \le r \le l)$ of the *collector matrix* $\mathbf{V}$ for the largest step stochastic autobisimulation $\mathcal{R}_{ss}(G)$ on G are defined as

$$\mathcal{V}_{ir} = \begin{cases} 1, & s_i \in \mathcal{K}_r; \\ 0, & \text{otherwise.} \end{cases}$$

Thus, all the elements of $\mathbf{V}$ are non-negative, as required. The row elements of $\mathbf{V}$ are summed to one, since for each s_i $(1 \le i \le n)$ there exists exactly one $\mathcal{K}_r$ $(1 \le r \le l)$ such that $s_i \in \mathcal{K}_r$. Hence,

$$\mathbf{V}\mathbf{1}^T = \mathbf{1}^T,$$

where $\mathbf{1}$ on the left side is the row vector of l values 1 while $\mathbf{1}$ on the right side is the row vector of n values 1.

The *distributor matrix* $\mathbf{W}$ for the collector matrix $\mathbf{V}$ is defined as

$$\mathbf{W} = (Diag(\mathbf{V}^T\mathbf{1}^T))^{-1}\mathbf{V}^T,$$

where $\mathbf{1}$ is the row vector of n values 1. One can check that $\mathbf{WV} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix of order l.

The elements $(\mathcal{P}\mathcal{V})_{is}$ $(1 \le i \le n,\ 1 \le s \le l)$ of the matrix $\mathbf{PV}$ are

$$(\mathcal{P}\mathcal{V})_{is} = \sum_{j=1}^{n} \mathcal{P}_{ij}\mathcal{V}_{js} = \sum_{\{j|1\le j\le n,\ s_j\in\mathcal{K}_s\}} PM(s_i, s_j) = PM(s_i, \mathcal{K}_s).$$

As we know, for each s_i $(1 \le i \le n)$ there exists exactly one $\mathcal{K}_r$ $(1 \le r \le l)$ such that $s_i \in \mathcal{K}_r$. For all $s_i \in \mathcal{K}_r$ we have $PM(\mathcal{K}_r, \mathcal{K}_s) = PM(s_i, \mathcal{K}_s)$ $(1 \le i \le n,\ 1 \le r, s \le l)$. Then the elements $(\mathcal{V}\mathcal{P}_{\underline{\leftrightarrow}_{ss}})_{is}$ $(1 \le i \le n,\ 1 \le s \le l)$ of the matrix $\mathbf{VP}_{\underline{\leftrightarrow}_{ss}}$ are

$$(\mathcal{V}\mathcal{P}_{\underline{\leftrightarrow}_{ss}})_{is} = \sum_{r=1}^{l} \mathcal{V}_{ir}(\mathcal{P}_{\underline{\leftrightarrow}_{ss}})_{rs} = \sum_{\{r|1\le r\le l,\ s_i\in\mathcal{K}_r\}} PM(\mathcal{K}_r, \mathcal{K}_s) = PM(s_i, \mathcal{K}_s).$$

Therefore, we have

$$\mathbf{PV} = \mathbf{VP}_{\underline{\leftrightarrow}_{ss}},\ \mathbf{WPV} = \mathbf{P}_{\underline{\leftrightarrow}_{ss}}.$$

Example 7.7. *Let F be from Example 6.1. The TPMs for $DTMC(\overline{F})$ and $DTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ are*

$$\mathbf{P} = \begin{pmatrix} 1-\rho & \rho & 0 & 0 & 0 \\ 0 & 1-\chi & \chi & 0 & 0 \\ 0 & 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & \theta & 0 & 1-\theta & 0 \\ 0 & \theta & 0 & 0 & 1-\theta \end{pmatrix},\ \mathbf{P}_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 1-\rho & \rho & 0 & 0 \\ 0 & 1-\chi & \chi & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \theta & 0 & 1-\theta \end{pmatrix}.$$

The collector matrix $\mathbf{V}$ for $\mathcal{R}_{ss}(\overline{F})$ and the distributor matrix $\mathbf{W}$ for $\mathbf{V}$ are

$$\mathbf{V} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix},\ \mathbf{W} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}.$$

Then it is easy to check that

$$\mathbf{WPV} = \mathbf{P}_{\underline{\leftrightarrow}_{ss}}.$$

In Figure 18, the relation (depicted by arrow) between *RDTMC* and $RDTMC_{\underline{\leftrightarrow}_{ss}}$ is obtained using the transition function $PM^{\circ}(\mathcal{K}, \widetilde{\mathcal{K}})$, based on $PM^{\circ}(s, \tilde{s})$. Let G be a dynamic expression. We shall prove that the TPM $\mathbf{P}^{\diamond}_{\underline{\leftrightarrow}_{ss}}$, (forwardly) constructed by quotienting (by $\underline{\leftrightarrow}_{ss}$) $DTMC(G)$, followed by reduction (eliminating vanishing states) of $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$, *coincides with* the TPM $(\mathbf{P}^{\diamond})_{\underline{\leftrightarrow}_{ss}}$, (reversely) constructed by reduction of $DTMC(G)$, followed by quotienting $RDTMC(G)$. The following proposition relates those quotient reduced TPM $(\mathbf{P}^{\diamond})_{\underline{\leftrightarrow}_{ss}}$ and reduced quotient TPM $\mathbf{P}^{\diamond}_{\underline{\leftrightarrow}_{ss}}$.

Proposition 7.3. *Let G be a dynamic expression,* $\mathbf{P}^{\diamond}_{\underline{\leftrightarrow}_{ss}}$ *be the TPM for* $RDTMC_{\underline{\leftrightarrow}_{ss}}(G)$ *and* $(\mathbf{P}^{\diamond})_{\underline{\leftrightarrow}_{ss}}$ *results from quotienting (by* $\underline{\leftrightarrow}_{ss}$*) the TPM* $\mathbf{P}^{\diamond}$ *for* $RDTMC(G)$*. Then*

$$(\mathbf{P}^{\diamond})_{\underline{\leftrightarrow}_{ss}} = \mathbf{P}^{\diamond}_{\underline{\leftrightarrow}_{ss}}.$$

Proof. See Appendix A.5. □

Thus, quotienting and reduction are permutable for DTMCs of the process expressions. This may simplify the performance evaluation when eliminating vanishing states makes the subsequent quotienting more efficient. The reverse construction (reduction first) is particularly preferable in case of small equivalence classes of vanishing states when quotienting does not merge many of them before eliminating.

Example 7.8. *Let F be from Example 6.1. The reordered TPMs for* $DTMC(\overline{F})$ *and* $DTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ *are*

$$\mathbf{P}_r = \begin{pmatrix} 0 & 0 & 0 & \frac{l}{l+m} & \frac{m}{l+m} \\ 0 & 1-\rho & \rho & 0 & 0 \\ \chi & 0 & 1-\chi & 0 & 0 \\ 0 & 0 & \theta & 1-\theta & 0 \\ 0 & 0 & \theta & 0 & 1-\theta \end{pmatrix}, \ \mathbf{P}_{r\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 1-\rho & \rho & 0 \\ \chi & 0 & 1-\chi & 0 \\ 0 & 0 & \theta & 1-\theta \end{pmatrix}.$$

The reordered collector matrix $\mathbf{V}_r$ *for* $\mathcal{R}_{ss}(\overline{F})$ *and the reordered distributor matrix* $\mathbf{W}_r$ *for* $\mathbf{V}_r$ *are*

$$\mathbf{V}_r = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \ \mathbf{W}_r = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}.$$

Then it is easy to check that

$$\mathbf{W}_r\mathbf{P}_r\mathbf{V}_r = \mathbf{P}_{r\underline{\leftrightarrow}_{ss}}.$$

Example 7.9. *Let F be from Example 6.1. The TPMs for* $RDTMC(\overline{F})$ *and* $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ *are*

$$\mathbf{P}^{\diamond} = \begin{pmatrix} 1-\rho & \rho & 0 & 0 \\ 0 & 1-\chi & \frac{\chi l}{l+m} & \frac{\chi m}{l+m} \\ 0 & \theta & 1-\theta & 0 \\ 0 & \theta & 0 & 1-\theta \end{pmatrix}, \ \mathbf{P}^{\diamond}_{\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} 1-\rho & \rho & 0 \\ 0 & 1-\chi & \chi \\ 0 & \theta & 1-\theta \end{pmatrix}.$$

The result of the decomposing the reordered collector matrix $\mathbf{V}_r$ *for* $\mathcal{R}_{ss}(\overline{F})$ *and the reordered distributor matrix* $\mathbf{W}_r$ *for* $\mathbf{V}_r$ *are the matrices*

$$\mathbf{V}_C = 1, \ \mathbf{V}_F = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \ \mathbf{W}_C = 1, \ \mathbf{W}_F = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}.$$

Then it is easy to check that

$$(\mathbf{P}^{\diamond})_{\underline{\leftrightarrow}_{ss}} = \mathbf{W}_F \mathbf{P}^{\diamond} \mathbf{V}_F = \mathbf{P}^{\diamond}_{\underline{\leftrightarrow}_{ss}}.$$

The comprehensive quotient and reduction example will be presented in Section 9.

In [26], the ordinary, exact and strict lumpability relations on finite DTMCs are explored. It is investigated which properties of transient and stationary behaviour of DTMCs are preserved by aggregation w.r.t. the three mentioned kinds of lumping and their approximate "nearly" versions. It is proved that irreducibility is preserved by aggregation w.r.t. any partition (or equivalence relation) on the states of DTMCs. Since only finite irreducible DTMCs are considered (with a finite number of states), these all are positive recurrent. Aggregation can only decrease the number of states, hence, the aggregated DTMCs are also finite and positive recurrence is preserved by every aggregation. It is known [91, 62] that irreducible and positive recurrent DTMCs have a single stationary PMF. Note that the original and aggregated DTMCs may be periodic, thus having a unique stationary distribution, but no steady-state (limiting) one. For example, it may happen that the original DTMC is aperiodic while the aggregated DTMC is periodic due to merging some states of the former. Thus, both finite irreducible DTMCs and their arbitrary aggregates have a single stationary PMF. Then the relationship between stationary probabilities of DTMCs and their aggregates w.r.t. ordinary, exact and strict lumpability is established in [26]. In particular, it is shown that for every DTMC aggregated by ordinary lumpability, the stationary probability of each aggregate state is a sum of the stationary probabilities of all its constituent states from the original DTMC. The information about individual stationary probabilities of the original DTMC is lost after such a summation, but in many cases, the stationary probabilities of the aggregated DTMC are enough to calculate performance measures of the high-level model, from which the original DTMC is extracted. As mentioned in [26], in some practical applications, the aggregated DTMC can be extracted directly from the high-level model. Thus, the aggregation techniques based on lumping are of practical importance, since they allow one to reduce the state space of the modeled systems, hence, the computational costs for evaluating their performance.

Let G be a dynamic expression. By definition of $\underline{\leftrightarrow}_{ss}$, the relation $\mathcal{R}_{ss}(G)$ on $TS(G)$ induces ordinary lumping on $SMC(G)$, i.e. if the states of $TS(G)$ are related by $\mathcal{R}_{ss}(G)$ then the same states in $SMC(G)$ are related by ordinary lumping. The quotient (maximal aggregate) of $SMC(G)$ by such an induced ordinary lumping is $SMC_{\underline{\leftrightarrow}_{ss}}(G)$. Since we consider only finite SMCs, irreducibility of $SMC(G)$ will imply irreducibility of $SMC_{\underline{\leftrightarrow}_{ss}}(G)$ and they both are positive recurrent. Then a unique quotient stationary PMF of $SMC_{\underline{\leftrightarrow}_{ss}}(G)$ can be calculated from a unique original stationary PMF of $SMC(G)$ by summing some elements of the latter, as described in [26]. Similar arguments demonstrate that the same results hold for $DTMC(G)$ and $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$, as well as for $RDTMC(G)$ and $RDTMC_{\underline{\leftrightarrow}_{ss}}(G)$.

8. 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, 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.

Like in the framework of SMCs, in LDTSIPNs the most common systems for performance analysis are *ergodic* (irreducible, positive recurrent and aperiodic) ones. For ergodic LDTSIPNs, the steady-state marking probabilities exist and can be determined. In [80, 81], the following sufficient (but not necessary) conditions for ergodicity of DTSPNs are stated: *liveness* (for each transition and any reachable marking there exists a sequence of markings from it leading to the marking enabling that transition), *boundedness* (for any reachable marking the number of tokens in every place is not greater than some fixed number) and *nondeterminism* (the transition probabilities are strictly less than 1).

Consider dtsi-box of a dynamic expression $G = \overline{[E * F * \mathsf{Stop}]}$ specifying a process for which we assume that it has no deadlocks while (repetitive) running the body F of the iteration operator. If, starting in $[[E * \overline{F} * \mathsf{Stop}]]_{\approx}$ and

ending in $[[E * \underline{F} * \mathsf{Stop}]]_{\approx}$, only tangible states are passed through then the three ergodicity conditions are satisfied: the subnet corresponding to the looping of the iteration body F is live, safe (1-bounded) and nondeterministic (since all markings of the subnet are tangible and non-terminal, the probabilities of transitions from them are strictly less than 1). Hence, according to [80, 81], for the dtsi-box, its underlying SMC, restricted to the markings of the mentioned subnet, is ergodic. The isomorphism between SMCs of expressions and those of the corresponding dtsi-boxes, which is stated by Proposition 5.1, guarantees that $SMC(G)$ is ergodic, if restricted to the states between $[[E * \overline{F} * \mathsf{Stop}]]_{\approx}$ and $[[E * \underline{F} * \mathsf{Stop}]]_{\approx}$.

The ergodicity conditions above are not necessary, i.e. there exist dynamic expressions with vanishing states traversed while executing their iteration bodies, such that the properly restricted underlying SMCs are nevertheless ergodic, as Example 5.1 demonstrated. However, it has been shown in [7] that even live, safe and nondeterministic DTSPNs (as well as live and safe CTSPNs and GSPNs) may be non-ergodic.

In this section, we consider only the process 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, 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.

8.1. Steady state, residence time and equivalences

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

Proposition 8.1. *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. See Appendix A.6. □

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 8.1 (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 $\varphi_{\underline{\leftrightarrow}_{ss}}$ corresponding to $\mathcal{K}$ is the sum of all values of φ corresponding to the states from $\mathcal{K}$.

Let $\mathbf{V}$ be the collector matrix for $\mathcal{R}_{ss}(G)$. One can see that

$$\varphi \mathbf{V} = \varphi_{\underline{\leftrightarrow}_{ss}}.$$

Hence, using $SMC_{\underline{\leftrightarrow}_{ss}}(G)$ instead of $SMC(G)$ may simplify the analytical solution, since we may have less states, but constructing the TPM for $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$, denoted by $\mathbf{P}^*_{\underline{\leftrightarrow}_{ss}}$, also requires some efforts, including determining $\mathcal{R}_{ss}(G)$ and calculating the probabilities to move from one equivalence class to other. The behaviour of $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$ may stabilize quicker than that of $EDTMC(G)$ (if each of them has a single steady state), since $\mathbf{P}^*_{\underline{\leftrightarrow}_{ss}}$ is generally denser matrix than $\mathbf{P}^*$ (the TPM for $EDTMC(G)$) due to the fact that the former matrix is usually smaller and the transitions between the equivalence classes "include" all the transitions between the states belonging to these equivalence classes.

By Proposition 8.1, $\underline{\leftrightarrow}_{ss}$ 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 8.1. A *derived step trace* of a dynamic expression G is a chain $\Sigma = A_1 \cdots A_n \in (\boldsymbol{N}^{\mathcal{L}}_{fin})^*$, 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

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

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 8.1. *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. See Appendix A.7. □

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 8.1 (modified for $\mathcal{R}_{\mathcal{L}ss}(G)$), we have $\forall \mathcal{K} \in DR(G)/_{\mathcal{R}_{ss}(G)}$,

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

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

Let $DR(G) = \{s_1, \ldots, s_n\}$ and $DR(G)/_{\mathcal{R}_{ss}(G)} = \{\mathcal{K}_1, \ldots, \mathcal{K}_l\}$ while $\mathbf{V}$ be the collector matrix for $\mathcal{R}_{ss}(G)$ and $\mathbf{W}$ be the distributor matrix for $\mathbf{V}$. We denote $PT(\Sigma) = (PT(\Sigma, s_1), \ldots, PT(\Sigma, s_n))$ and $PT_{\underline{\leftrightarrow}_{ss}}(\Sigma) = (PT(\Sigma, \mathcal{K}_1), \ldots, PT(\Sigma, \mathcal{K}_l))$. One can see that $Diag(PT(\Sigma))\mathbf{V} = \mathbf{V} Diag(PT_{\underline{\leftrightarrow}_{ss}}(\Sigma))$ and $\mathbf{W} Diag(PT(\Sigma))\mathbf{V} = Diag(PT_{\underline{\leftrightarrow}_{ss}}(\Sigma))$. Then we have

$$\varphi Diag(PT(\Sigma))\mathbf{V} = \varphi \mathbf{V} Diag(PT_{\underline{\leftrightarrow}_{ss}}(\Sigma)) = \varphi_{\underline{\leftrightarrow}_{ss}} Diag(PT_{\underline{\leftrightarrow}_{ss}}(\Sigma)).$$

We now present a result not concerning the steady-state probabilities, but revealing very important properties of residence time in the equivalence classes. 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 8.2. *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. See Appendix A.8. □

Example 8.1. *Let*

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

We have $\overline{E} \underline{\leftrightarrow}_{ss} \overline{E'}$.

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

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

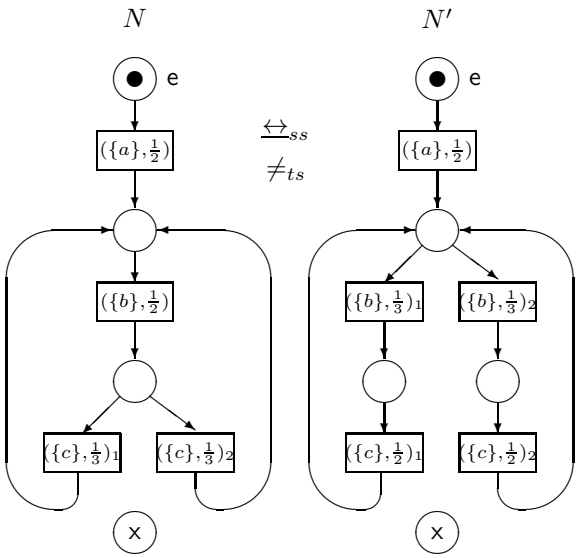

Figure 19: $\underline{\leftrightarrow}_{ss}$ preserves steady-state behaviour and sojourn time properties in the equivalence classes.

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

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

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

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

Consider the equivalence class (with respect to $\mathcal{R}_{ss}(\overline{E},\overline{E'})$) $\mathcal{H} = \{s_3, s'_3, s'_4\}$. One can see that the steady-state probabilities for $\mathcal{H}$ coincide: $\sum_{s\in\mathcal{H}\cap DR(\overline{E})}\varphi(s) = \varphi(s_3) = \frac{1}{2} = \frac{1}{4}+\frac{1}{4} = \varphi'(s'_3)+\varphi'(s'_4) = \sum_{s'\in\mathcal{H}\cap DR(\overline{E'})}\varphi'(s')$.

Let $\Sigma = \{\{c\}\}$. The steady-state probabilities to enter into the equivalence class $\mathcal{H}$ and start the derived step trace Σ from it coincide as well: $\varphi(s_3)(PT(\{(\{c\},\frac{1}{3})_1\}, s_3) + PT(\{(\{c\},\frac{1}{3})_2\}, s_3)) = \frac{1}{2}\left(\frac{1}{4}+\frac{1}{4}\right) = \frac{1}{4} = \frac{1}{4}\cdot\frac{1}{2}+\frac{1}{4}\cdot\frac{1}{2} = \varphi'(s'_3)PT(\{(\{c\},\frac{1}{2})_1\}, s'_3) + \varphi'(s'_4)PT(\{(\{c\},\frac{1}{2})_2\}, s'_4)$.

Further, the sojourn time averages in the equivalence class $\mathcal{H}$ coincide: $SJ_{\mathcal{R}_{ss}(\overline{E},\overline{E'})\cap(DR(\overline{E}))^2}(\mathcal{H}\cap DR(G)) = SJ_{\mathcal{R}_{ss}(\overline{E},\overline{E'})\cap(DR(\overline{E}))^2}(\{s_3\}) = \frac{1}{1-PM(\{s_3\},\{s_3\})} = \frac{1}{1-PM(s_3,s_3)} = \frac{1}{1-\frac{1}{2}} = 2 = \frac{1}{1-\frac{1}{2}} = \frac{1}{1-PM(s'_3,s'_3)} = \frac{1}{1-PM(s'_4,s'_4)} = \frac{1}{1-PM(\{s'_3,s'_4\},\{s'_3,s'_4\})} = SJ_{\mathcal{R}_{ss}(\overline{E},\overline{E'})\cap(DR(\overline{E'}))^2}(\{s'_3, s'_4\}) = SJ_{\mathcal{R}_{ss}(\overline{E},\overline{E'})\cap(DR(\overline{E'}))^2}(\mathcal{H}\cap DR(G'))$.

Finally, the sojourn time variances in the equivalence class $\mathcal{H}$ coincide: $VAR_{\mathcal{R}_{ss}(\overline{E},\overline{E'})\cap(DR(\overline{E}))^2}(\mathcal{H}\cap DR(G)) = VAR_{\mathcal{R}_{ss}(\overline{E},\overline{E'})\cap(DR(\overline{E}))^2}(\{s_3\}) = \frac{PM(\{s_3\},\{s_3\})}{(1-PM(\{s_3\},\{s_3\}))^2} = \frac{PM(s_3,s_3)}{(1-PM(s_3,s_3))^2} = \frac{\frac{1}{2}}{\left(1-\frac{1}{2}\right)^2} = 2 = \frac{\frac{1}{2}}{\left(1-\frac{1}{2}\right)^2} = \frac{PM(s'_3,s'_3)}{(1-PM(s'_3,s'_3))^2} = \frac{PM(s'_4,s'_4)}{(1-PM(s'_4,s'_4))^2} = \frac{PM(\{s'_3,s'_4\},\{s'_3,s'_4\})}{(1-PM(\{s'_3,s'_4\},\{s'_3,s'_4\}))^2} = VAR_{\mathcal{R}_{ss}(\overline{E},\overline{E'})\cap(DR(\overline{E'}))^2}(\{s'_3, s'_4\}) = VAR_{\mathcal{R}_{ss}(\overline{E},\overline{E'})\cap(DR(\overline{E'}))^2}(\mathcal{H}\cap DR(G'))$.

In Figure 19, the marked dtsi-boxes corresponding to the dynamic expressions above are presented, i.e. $N = Box_{dtsi}(\overline{E})$ and $N' = Box_{dtsi}(\overline{E'})$.

8.2. 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. The similarity of states is usually captured by an equivalence

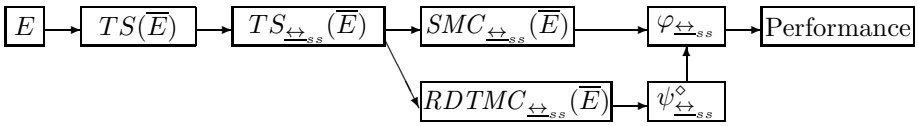

Figure 20: Equivalence-based simplification of performance evaluation.

relation, hence, the sets are often the equivalence classes. Proposition 8.1, Theorem 8.1 and Proposition 8.2 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 dtsiPBC.

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

1. The investigated system is specified by a static expression of dtsiPBC.
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 underlying SMC is constructed from the quotient transition system.
5. Stationary probabilities and performance indices are calculated using the SMC.

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 [62]. Note that 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. It is worth applying the method only to the systems with similar subprocesses.

Before calculating stationary probabilities, we can further reduce the quotient underlying SMC, using an analogue of the deterministic barrier partitioning method from [44] for semi-Markov processes (SMPs), which allows one to perform quicker the first passage-time analysis. Another option is the method of stochastic state classes from [54] for generalized SMPs (GSMPs) reduction, allowing one to simplify transient performance analysis (based on the transient probabilities of being in the states of GSMPs).

Alternatively, the results at the end of Section 7 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 this 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 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.

We first merge the equivalent states in transition systems and only then eliminate the vanishing states in Markov chains. The reason is that transition systems, being a higher-level formalism than Markov chains, describe both functional (qualitative) and performance (quantitative) aspects of behaviour while Markov chains represent only performance ones. Thus, eliminating vanishing states first would destroy the functional behaviour (which is respected by the equivalence used for quotienting), since the steps with different multiaction parts may lead to or start from different vanishing states.

Figure 20 presents the main stages of the standard and alternative equivalence-based simplification of performance evaluation described above.

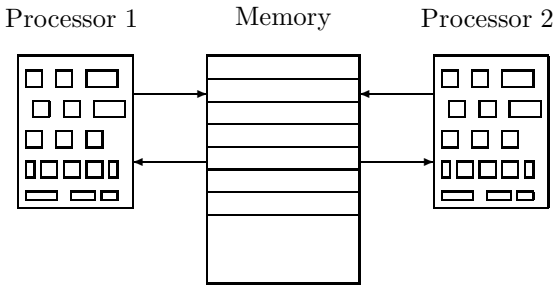

Figure 21: The diagram of the shared memory system.

9. Shared memory system

In this section with a case study of the shared memory system we show how steady-state distribution can be used for performance evaluation. The example also illustrates the method of performance analysis simplification above.

9.1. The standard system

Consider a model of two processors accessing a common shared memory described in [75, 5, 6] in the continuous time setting on GSPNs. We shall analyze this shared memory system in the discrete time stochastic setting of dtsiPBC, where concurrent execution of activities is possible, while no two transitions of a GSPN may fire simultaneously (in parallel). The model works 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. 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 the available common memory. The diagram of the system is depicted in Figure 21.

Let us explain the meaning of actions from the syntax of dtsiPBC expressions which will specify the system modules. The action a corresponds to the system activation. The actions r_i $(1 \le i \le 2)$ represent the common memory request 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 $a_1, \ldots, a_n \in Act$ $(n \in \mathbf{N})$, we shall abbreviate $\mathsf{sy}\ a_1 \cdots \mathsf{sy}\ a_n\ \mathsf{rs}\ a_1 \cdots \mathsf{rs}\ a_n$ to $\mathsf{sr}\ (a_1, \ldots, a_n)$.

The static expression of the first processor is

$$E_1 = [(\{x_1\}, \frac{1}{2}) * ((\{r_1\}, \frac{1}{2}); (\{d_1, y_1\}, \natural_1); (\{m_1, z_1\}, \frac{1}{2})) * \mathsf{Stop}].$$

The static expression of the second processor is

$$E_2 = [(\{x_2\}, \frac{1}{2}) * ((\{r_2\}, \frac{1}{2}); (\{d_2, y_2\}, \natural_1); (\{m_2, z_2\}, \frac{1}{2})) * \mathsf{Stop}].$$

The static expression of the shared memory is

$$E_3 = [(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); (\{\widehat{z_1}\}, \frac{1}{2}))[](((\{\widehat{y_2}\}, \natural_1); (\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}].$$

The static expression of the shared memory system with two processors is

$$E = (E_1 \| E_2 \| E_3)\ \mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2).$$

Let us illustrate an effect of synchronization. As result of the synchronization of immediate multiactions $(\{d_i, y_i\}, \natural_1)$ and $(\{\widehat{y_i}\}, \natural_1)$ we obtain $(\{d_i\}, \natural_2)$ $(1 \le i \le 2)$. The synchronization of stochastic multiactions $(\{m_i, z_i\}, \frac{1}{2})$ and $(\{\widehat{z_i}\}, \frac{1}{2})$ produces $(\{m_i\}, \frac{1}{4})$ $(1 \le i \le 2)$. The result of synchronization of $(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2})$ with $(\{x_1\}, \frac{1}{2})$ is $(\{a, \widehat{x_2}\}, \frac{1}{4})$,

and that of synchronization of $(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2})$ with $(\{x_2\}, \frac{1}{2})$ is $(\{a, \widehat{x_1}\}, \frac{1}{4})$. After applying synchronization to $(\{a, \widehat{x_2}\}, \frac{1}{4})$ and $(\{x_2\}, \frac{1}{2})$, as well as to $(\{a, \widehat{x_1}\}, \frac{1}{4})$ and $(\{x_1\}, \frac{1}{2})$, we obtain the same activity $(\{a\}, \frac{1}{8})$.

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

$$
\begin{array}{c}
s_1 = [(\overline{[(\overline{\{x_1\}, \frac{1}{2}}) * ((\{r_1\}, \frac{1}{2}); (\{d_1, y_1\}, \natural_1); (\{m_1, z_1\}, \frac{1}{2})) * \mathsf{Stop}]}\|| \\
\overline{[(\{x_2\}, \frac{1}{2}) * ((\{r_2\}, \frac{1}{2}); (\{d_2, y_2\}, \natural_1); (\{m_2, z_2\}, \frac{1}{2})) * \mathsf{Stop}]}\| \\
[(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); (\{\widehat{z_1}\}, \frac{1}{2}))[](\overline{(\{\widehat{y_2}\}, \natural_1)}; (\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}]) \\
\mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},
\end{array}
$$

$$
\begin{array}{c}
s_2 = [(\overline{[(\{x_1\}, \frac{1}{2}) * ((\{r_1\}, \frac{1}{2}); (\{d_1, y_1\}, \natural_1); (\{m_1, z_1\}, \frac{1}{2})) * \mathsf{Stop}]}\|| \\
[(\{x_2\}, \frac{1}{2}) * \overline{((\{r_2\}, \frac{1}{2}); (\{d_2, y_2\}, \natural_1); (\{m_2, z_2\}, \frac{1}{2})) * \mathsf{Stop}]}\| \\
[(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); (\{\widehat{z_1}\}, \frac{1}{2}))[](\overline{(\{\widehat{y_2}\}, \natural_1)}; (\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}]) \\
\mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},
\end{array}
$$

$$
\begin{array}{c}
s_3 = [(\overline{[(\{x_1\}, \frac{1}{2}) * ((\{r_1\}, \frac{1}{2}); (\{d_1, y_1\}, \natural_1); (\{m_1, z_1\}, \frac{1}{2})) * \mathsf{Stop}]}\|| \\
[(\{x_2\}, \frac{1}{2}) * ((\overline{\{r_2\}, \frac{1}{2}); (\{d_2, y_2\}, \natural_1); (\{m_2, z_2\}, \frac{1}{2})) * \mathsf{Stop}]}\| \\
[(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); (\{\widehat{z_1}\}, \frac{1}{2}))[](\overline{(\{\widehat{y_2}\}, \natural_1)}; (\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}]) \\
\mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},
\end{array}
$$

$$
\begin{array}{c}
s_4 = [(\overline{[(\{x_1\}, \frac{1}{2}) * ((\{r_1\}, \frac{1}{2}); (\{d_1, y_1\}, \natural_1); (\{m_1, z_1\}, \frac{1}{2})) * \mathsf{Stop}]}\|| \\
[(\{x_2\}, \frac{1}{2}) * ((\{r_2\}, \frac{1}{2}); \overline{(\{d_2, y_2\}, \natural_1); (\{m_2, z_2\}, \frac{1}{2})) * \mathsf{Stop}]}\| \\
[(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); (\{\widehat{z_1}\}, \frac{1}{2}))[](\overline{(\{\widehat{y_2}\}, \natural_1)}; (\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}]) \\
\mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},
\end{array}
$$

$$
\begin{array}{c}
s_5 = [(\overline{[(\{x_1\}, \frac{1}{2}) * ((\{r_1\}, \frac{1}{2}); (\{d_1, y_1\}, \natural_1); (\{m_1, z_1\}, \frac{1}{2})) * \mathsf{Stop}]}\|| \\
[(\{x_2\}, \frac{1}{2}) * ((\{r_2\}, \frac{1}{2}); (\{d_2, y_2\}, \natural_1); (\{m_2, z_2\}, \frac{1}{2})) * \mathsf{Stop}]}\| \\
[(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); \overline{(\{\widehat{z_1}\}, \frac{1}{2}))[](\overline{(\{\widehat{y_2}\}, \natural_1)}; (\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}]) \\
\mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},
\end{array}
$$

$$
\begin{array}{c}
s_6 = [(\overline{[(\{x_1\}, \frac{1}{2}) * ((\{r_1\}, \frac{1}{2}); (\{d_1, y_1\}, \natural_1); (\{m_1, z_1\}, \frac{1}{2})) * \mathsf{Stop}]}\|| \\
[(\{x_2\}, \frac{1}{2}) * \overline{((\{r_2\}, \frac{1}{2}); (\{d_2, y_2\}, \natural_1); (\{m_2, z_2\}, \frac{1}{2})) * \mathsf{Stop}]}\| \\
[(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); (\{\widehat{z_1}\}, \frac{1}{2}))[](\overline{(\{\widehat{y_2}\}, \natural_1)}; (\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}]) \\
\mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},
\end{array}
$$

$$
\begin{array}{c}
s_7 = [(\overline{[(\{x_1\}, \frac{1}{2}) * ((\{r_1\}, \frac{1}{2}); (\{d_1, y_1\}, \natural_1); (\{m_1, z_1\}, \frac{1}{2})) * \mathsf{Stop}]}\|| \\
[(\{x_2\}, \frac{1}{2}) * ((\{r_2\}, \frac{1}{2}); (\{d_2, y_2\}, \natural_1); \overline{(\{m_2, z_2\}, \frac{1}{2})) * \mathsf{Stop}]}\| \\
[(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); (\{\widehat{z_1}\}, \frac{1}{2}))[](\overline{(\{\widehat{y_2}\}, \natural_1)}; \overline{(\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}]) \\
\mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},
\end{array}
$$

$$
\begin{array}{c}
s_8 = [(\overline{[(\{x_1\}, \frac{1}{2}) * ((\{r_1\}, \frac{1}{2}); (\{d_1, y_1\}, \natural_1); (\{m_1, z_1\}, \frac{1}{2})) * \mathsf{Stop}]}\|| \\
[(\{x_2\}, \frac{1}{2}) * ((\{r_2\}, \frac{1}{2}); \overline{(\{d_2, y_2\}, \natural_1)}; (\{m_2, z_2\}, \frac{1}{2})) * \mathsf{Stop}]\| \\
[(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); \overline{(\{\widehat{z_1}\}, \frac{1}{2}))[](\overline{(\{\widehat{y_2}\}, \natural_1)}; (\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}]) \\
\mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx},
\end{array}
$$

$$
\begin{array}{c}
s_9 = [(\overline{[(\{x_1\}, \frac{1}{2}) * ((\{r_1\}, \frac{1}{2}); (\{d_1, y_1\}, \natural_1); (\{m_1, z_1\}, \frac{1}{2})) * \mathsf{Stop}]}\|| \\
[(\{x_2\}, \frac{1}{2}) * ((\{r_2\}, \frac{1}{2}); (\{d_2, y_2\}, \natural_1); \overline{(\{m_2, z_2\}, \frac{1}{2})) * \mathsf{Stop}]}\| \\
[(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); (\{\widehat{z_1}\}, \frac{1}{2}))[](\overline{(\{\widehat{y_2}\}, \natural_1)}; \overline{(\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}]) \\
\mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2)]_{\approx}.
\end{array}
$$

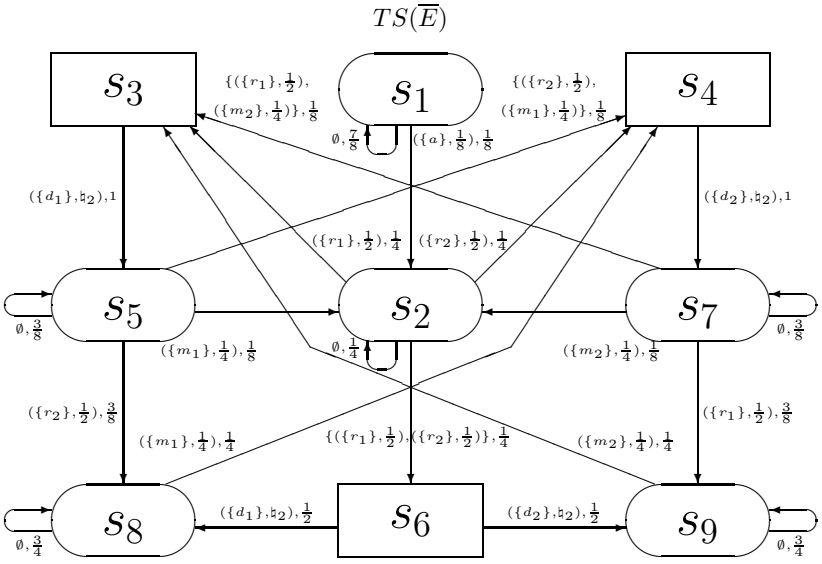

Figure 22: The transition system of the shared memory system.

We have $DR_T(\overline{E}) = \{s_1, s_2, s_5, s_7, s_8, s_9\}$ and $DR_V(\overline{E}) = \{s_3, s_4, s_6\}$.
The states are interpreted as follows: s_1 is the initial state, s_2: the system is activated and the memory is not requested, s_3: the memory is requested by the first processor, s_4: the memory is requested by the second processor, s_5: the memory is allocated to the first processor, s_6: the memory is requested by two processors, s_7: the memory is allocated to the second processor, s_8: the memory is allocated to the first processor and the memory is requested by the second processor, s_9: the memory is allocated to the second processor and the memory is requested by the first processor.

In Figure 22, the transition system $TS(\overline{E})$ is presented. In Figure 23, the underlying SMC $SMC(\overline{E})$ is depicted. Note that, in step semantics, we may execute the following activities in parallel: $(\{r_1\}, \frac{1}{2}), (\{r_2\}, \frac{1}{2})$, as well as $(\{r_1\}, \frac{1}{2}), (\{m_2\}, \frac{1}{4})$, and $(\{r_2\}, \frac{1}{2}), (\{m_1\}, \frac{1}{4})$. The state s_6 only exists in step semantics, since it is reachable exclusively by executing $(\{r_1\}, \frac{1}{2})$ and $(\{r_2\}, \frac{1}{2})$ in parallel.

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

$$SJ = \left(8, \frac{4}{3}, 0, 0, \frac{8}{5}, 0, \frac{8}{5}, 4, 4\right).$$

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

$$VAR = \left(56, \frac{4}{9}, 0, 0, \frac{24}{25}, 0, \frac{24}{25}, 12, 12\right).$$

The TPM for $EDTMC(\overline{E})$ is

$$\mathbf{P}^* = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{1}{5} & 0 & \frac{1}{5} & 0 & 0 & 0 & \frac{3}{5} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & \frac{1}{5} & \frac{1}{5} & 0 & 0 & 0 & 0 & 0 & \frac{3}{5} \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

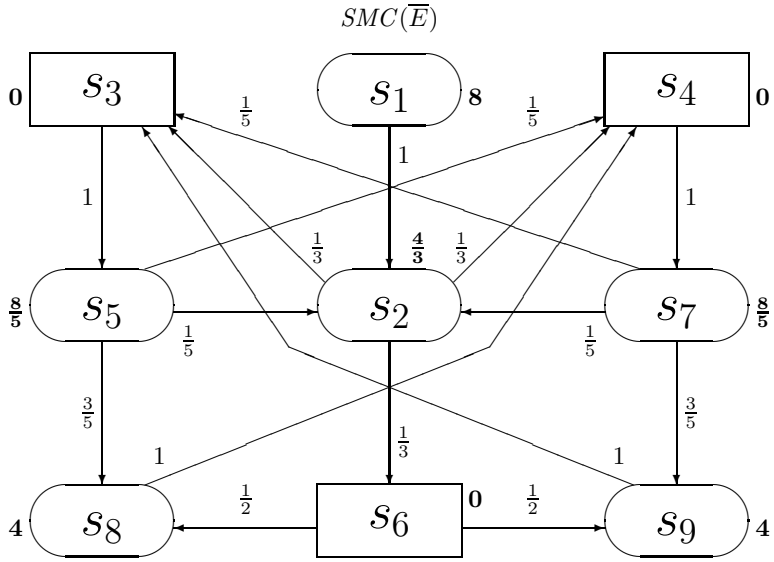

Figure 23: The underlying SMC of the shared memory system.

Table 7: Transient and steady-state probabilities for the EDTMC of the shared memory system.

k	0	5	10	15	20	25	30	35	40	45	50	∞
$\psi_1^*[k]$	1	0	0	0	0	0	0	0	0	0	0	0
$\psi_2^*[k]$	0	0	0.0754	0.0859	0.0677	0.0641	0.0680	0.0691	0.0683	0.0680	0.0681	0.0682
$\psi_3^*[k]$	0	0.2444	0.2316	0.1570	0.1554	0.1726	0.1741	0.1702	0.1696	0.1705	0.1707	0.1705
$\psi_5^*[k]$	0	0.2333	0.0982	0.1516	0.1859	0.1758	0.1672	0.1690	0.1711	0.1708	0.1703	0.1705
$\psi_6^*[k]$	0	0.0444	0.0323	0.0179	0.0202	0.0237	0.0234	0.0226	0.0226	0.0228	0.0228	0.0227
$\psi_8^*[k]$	0	0	0.1163	0.1395	0.1147	0.1077	0.1130	0.1150	0.1139	0.1133	0.1136	0.1136

In Table 7, the transient and the steady-state probabilities $\psi_i^*[k]$ ($i \in \{1, 2, 3, 5, 6, 8\}$) for the EDTMC of the shared memory system at the time moments $k \in \{0, 5, 10, \ldots, 50\}$ and $k = \infty$ are presented, and in Figure 24, the alteration diagram (evolution in time) for the transient probabilities is depicted. It is sufficient to consider the probabilities for the states s_1, s_2, s_3, s_5, s_6, s_8 only, since the corresponding values coincide for s_3, s_4, as well as for s_5, s_7, and for s_8, s_9.

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

$$\psi^* = \left(0, \frac{3}{44}, \frac{15}{88}, \frac{15}{88}, \frac{15}{88}, \frac{1}{44}, \frac{15}{88}, \frac{5}{44}, \frac{5}{44}\right).$$

The steady-state PMF ψ^* weighted by SJ is

$$\left(0, \frac{1}{11}, 0, 0, \frac{3}{11}, 0, \frac{3}{11}, \frac{5}{11}, \frac{5}{11}\right).$$

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

$$\psi^* SJ^T = \frac{17}{11}.$$

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

$$\varphi = \left(0, \frac{1}{17}, 0, 0, \frac{3}{17}, 0, \frac{3}{17}, \frac{5}{17}, \frac{5}{17}\right).$$

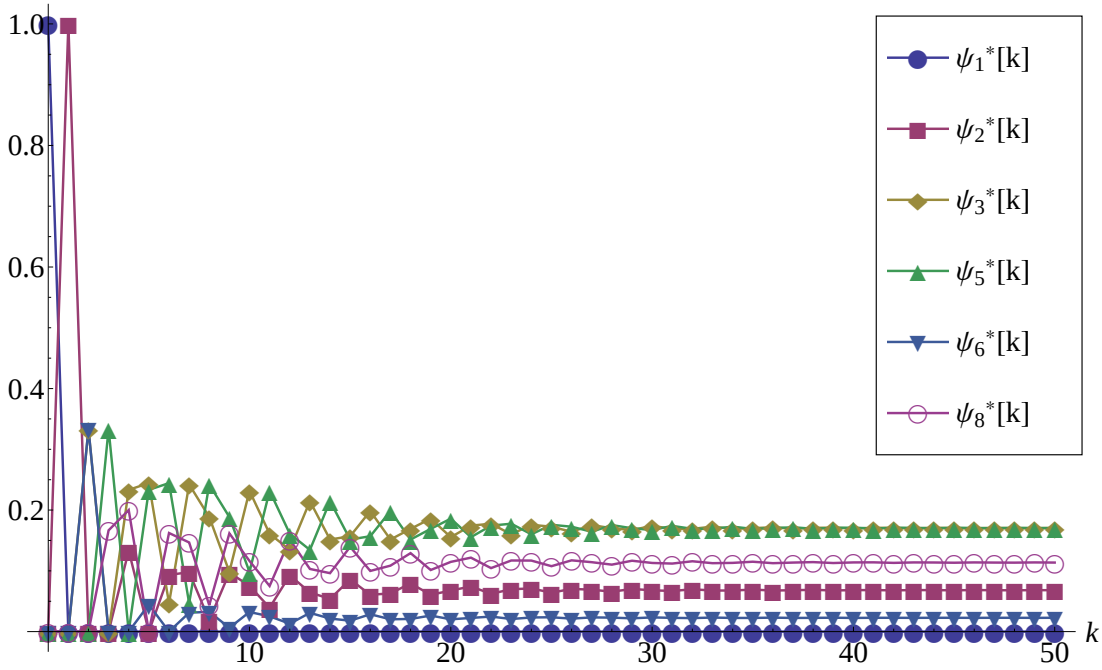

Figure 24: Transient probabilities alteration diagram for the EDTMC of the shared memory system.

Otherwise, from $TS(\overline{E})$, we can construct the DTMC of $\overline{E}$, $DTMC(\overline{E})$, and then calculate φ using it.
In Figure 25, the DTMC $DTMC(\overline{E})$ is depicted.
The TPM for $DTMC(\overline{E})$ is

$$\mathbf{P} = \begin{pmatrix} \frac{7}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{1}{8} & 0 & \frac{1}{8} & \frac{3}{8} & 0 & 0 & \frac{3}{8} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & \frac{1}{8} & \frac{1}{8} & 0 & 0 & 0 & \frac{3}{8} & 0 & \frac{3}{8} \\ 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & \frac{3}{4} & 0 \\ 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & \frac{3}{4} \end{pmatrix}.$$

In Table 8, the transient and the steady-state probabilities $\psi_i[k]$ ($i \in \{1, 2, 3, 5, 6, 8\}$) for the DTMC of the shared memory system at the time moments $k \in \{0, 5, 10, \ldots, 50\}$ and $k = \infty$ are presented, and in Figure 26, the alteration diagram (evolution in time) for the transient probabilities is depicted. It is sufficient to consider the probabilities for the states $s_1, s_2, s_3, s_5, s_6, s_8$ only, since the corresponding values coincide for s_3, s_4, as well as for s_5, s_7, and for s_8, s_9.
The steady-state PMF for $DTMC(\overline{E})$ is

$$\psi = \left(0, \frac{1}{21}, \frac{5}{56}, \frac{5}{56}, \frac{1}{7}, \frac{1}{84}, \frac{1}{7}, \frac{5}{21}, \frac{5}{21}\right).$$

Remember that $DR_T(\overline{E}) = \{s_1, s_2, s_5, s_7, s_8, s_9\}$ and $DR_V(\overline{E}) = \{s_3, s_4, s_6\}$. Hence,

$$\sum_{s \in DR_T(\overline{E})} \psi(s) = \psi(s_1) + \psi(s_2) + \psi(s_5) + \psi(s_7) + \psi(s_8) + \psi(s_9) = \frac{17}{21}.$$

By Proposition 5.2, we have

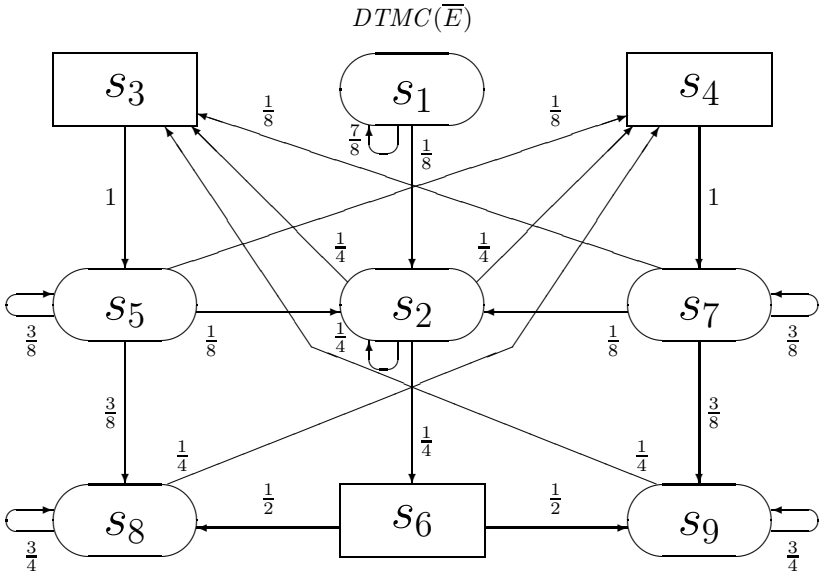

Figure 25: The DTMC of the shared memory system.

Table 8: Transient and steady-state probabilities for the DTMC of the shared memory system.

k	0	5	10	15	20	25	30	35	40	45	50	∞
$\psi_1[k]$	1	0.5129	0.2631	0.1349	0.0692	0.0355	0.0182	0.0093	0.0048	0.0025	0.0013	0
$\psi_2[k]$	0	0.1161	0.0829	0.0657	0.0569	0.0524	0.0501	0.0489	0.0483	0.0479	0.0478	0.0476
$\psi_3[k]$	0	0.0472	0.0677	0.0782	0.0836	0.0864	0.0878	0.0885	0.0889	0.0891	0.0892	0.0893
$\psi_5[k]$	0	0.0581	0.0996	0.1207	0.1315	0.1370	0.1399	0.1413	0.1421	0.1425	0.1427	0.1429
$\psi_6[k]$	0	0.0311	0.0220	0.0171	0.0146	0.0133	0.0126	0.0123	0.0121	0.0120	0.0120	0.0119
$\psi_8[k]$	0	0.0647	0.1487	0.1923	0.2146	0.2260	0.2319	0.2349	0.2365	0.2373	0.2377	0.2381

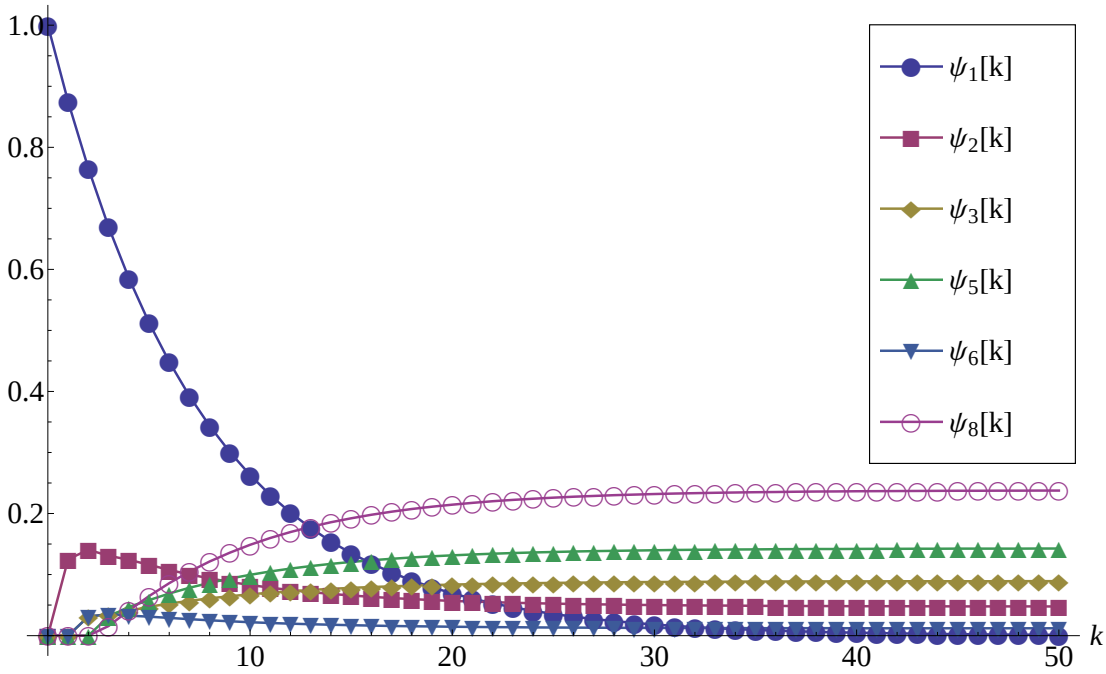

Figure 26: Transient probabilities alteration diagram for the DTMC of the shared memory system.

$$
\begin{aligned}
\varphi(s_1) &= 0 \cdot \tfrac{21}{17} = 0,\\
\varphi(s_2) &= \tfrac{1}{21} \cdot \tfrac{21}{17} = \tfrac{1}{17},\\
\varphi(s_3) &= 0,\\
\varphi(s_4) &= 0,\\
\varphi(s_5) &= \tfrac{1}{7} \cdot \tfrac{21}{17} = \tfrac{3}{17},\\
\varphi(s_6) &= 0,\\
\varphi(s_7) &= \tfrac{1}{7} \cdot \tfrac{21}{17} = \tfrac{3}{17},\\
\varphi(s_8) &= \tfrac{5}{21} \cdot \tfrac{21}{17} = \tfrac{5}{17},\\
\varphi(s_9) &= \tfrac{5}{21} \cdot \tfrac{21}{17} = \tfrac{5}{17}.
\end{aligned}
$$

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

$$\varphi = \left(0, \frac{1}{17}, 0, 0, \frac{3}{17}, 0, \frac{3}{17}, \frac{5}{17}, \frac{5}{17}\right).$$

This coincides with the result obtained with the use of ψ^* and SJ.

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

Remember that $DR_T(\overline{E}) = \{s_1, s_2, s_5, s_7, s_8, s_9\}$ and $DR_V(\overline{E}) = \{s_3, s_4, s_6\}$. We reorder the elements of $DR(\overline{E})$, by moving vanishing states to the first positions: $s_3, s_4, s_6, s_1, s_2, s_5, s_7, s_8, s_9$.

The reordered TPM for $DTMC(\overline{E})$ is

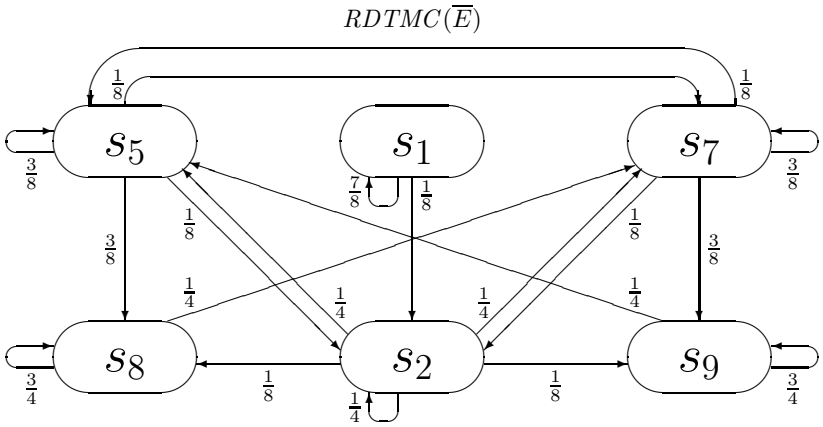

Figure 27: The reduced DTMC of the shared memory system.

$$\mathbf{P}_r = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & \frac{7}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{8} & 0 & 0 & \frac{1}{8} & \frac{3}{8} & 0 & \frac{3}{8} & 0 \\ \frac{1}{8} & 0 & 0 & 0 & \frac{1}{8} & 0 & \frac{3}{8} & 0 & \frac{3}{8} \\ 0 & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & \frac{3}{4} & 0 \\ \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3}{4} \end{pmatrix}.$$

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

$$\mathbf{C} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \ \mathbf{D} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}, \ \mathbf{E} = \begin{pmatrix} 0 & 0 & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ 0 & \frac{1}{8} & 0 \\ \frac{1}{8} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ \frac{1}{4} & 0 & 0 \end{pmatrix}, \ \mathbf{F} = \begin{pmatrix} \frac{7}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{8} & \frac{3}{8} & 0 & \frac{3}{8} & 0 \\ 0 & \frac{1}{8} & 0 & \frac{3}{8} & 0 & \frac{3}{8} \\ 0 & 0 & 0 & 0 & \frac{3}{4} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{3}{4} \end{pmatrix}.$$

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

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

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

$$\mathbf{P}^{\diamond} = \mathbf{F} + \mathbf{EGD} = \mathbf{F} + \mathbf{EID} = \mathbf{F} + \mathbf{ED} = \begin{pmatrix} \frac{7}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{8} & \frac{1}{8} \\ 0 & \frac{1}{8} & \frac{3}{8} & \frac{1}{8} & \frac{3}{8} & 0 \\ 0 & \frac{1}{8} & \frac{1}{8} & \frac{3}{8} & 0 & \frac{3}{8} \\ 0 & 0 & 0 & \frac{1}{4} & \frac{3}{4} & 0 \\ 0 & 0 & \frac{1}{4} & 0 & 0 & \frac{3}{4} \end{pmatrix}.$$

In Figure 27, the reduced DTMC $RDTMC(\overline{E})$ is presented.

Table 9: Transient and steady-state probabilities for the RDTMC of the shared memory system.

k	0	5	10	15	20	25	30	35	40	45	50	∞
$\psi_1^\diamond[k]$	1	0.5129	0.2631	0.1349	0.0692	0.0355	0.0182	0.0093	0.0048	0.0025	0.0013	0
$\psi_2^\diamond[k]$	0	0.1244	0.0931	0.0764	0.0679	0.0635	0.0612	0.0600	0.0594	0.0591	0.0590	0.0588
$\psi_3^\diamond[k]$	0	0.0863	0.1307	0.1530	0.1644	0.1703	0.1733	0.1748	0.1756	0.1760	0.1763	0.1765
$\psi_5^\diamond[k]$	0	0.0951	0.1912	0.2413	0.2670	0.2802	0.2870	0.2905	0.2922	0.2932	0.2936	0.2941

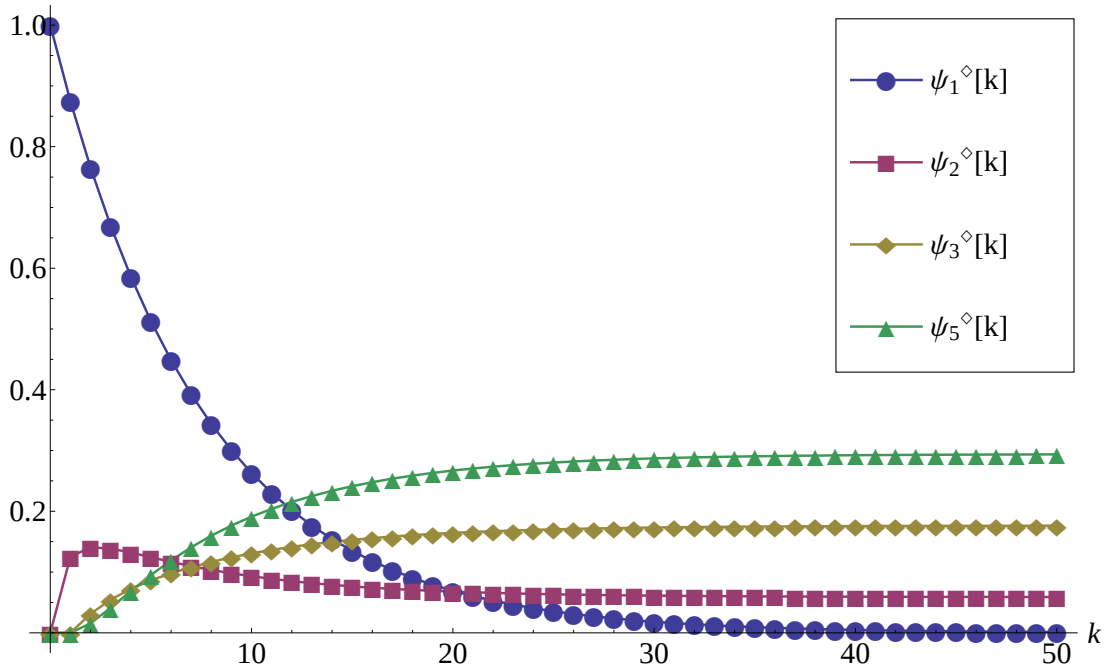

Figure 28: Transient probabilities alteration diagram for the RDTMC of the shared memory system.

In Table 9, the transient and the steady-state probabilities $\psi_i^\diamond[k]$ ($i \in \{1,2,3,5\}$) for the RDTMC of the shared memory system at the time moments $k \in \{0,5,10,\ldots,50\}$ and $k=\infty$ are presented, and in Figure 28, the alteration diagram (evolution in time) for the transient probabilities is depicted. It is sufficient to consider the probabilities for the states s_1, s_2, s_5, s_8 only, since the corresponding values coincide for s_5, s_7, as well as for s_8, s_9.

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

$$\psi^\diamond = \left(0, \frac{1}{17}, \frac{3}{17}, \frac{3}{17}, \frac{5}{17}, \frac{5}{17}\right).$$

Note that $\psi^\diamond = (\psi^\diamond(s_1), \psi^\diamond(s_2), \psi^\diamond(s_5), \psi^\diamond(s_7), \psi^\diamond(s_8), \psi^\diamond(s_9))$. By Proposition 5.3, we have

$$\varphi(s_1)=0, \quad \varphi(s_2)=\tfrac{1}{17}, \quad \varphi(s_3)=0 \quad \varphi(s_4)=0 \quad \varphi(s_5)=\tfrac{3}{17}, \quad \varphi(s_6)=0 \quad \varphi(s_7)=\tfrac{3}{17}, \quad \varphi(s_8)=\tfrac{5}{17}, \quad \varphi(s_9)=\tfrac{5}{17}.$$

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

$$\varphi = \left(0, \frac{1}{17}, 0, 0, \frac{3}{17}, 0, \frac{3}{17}, \frac{5}{17}, \frac{5}{17}\right).$$

This coincides with the result obtained with the use of ψ^* and SJ.

We can now calculate the main performance indices.

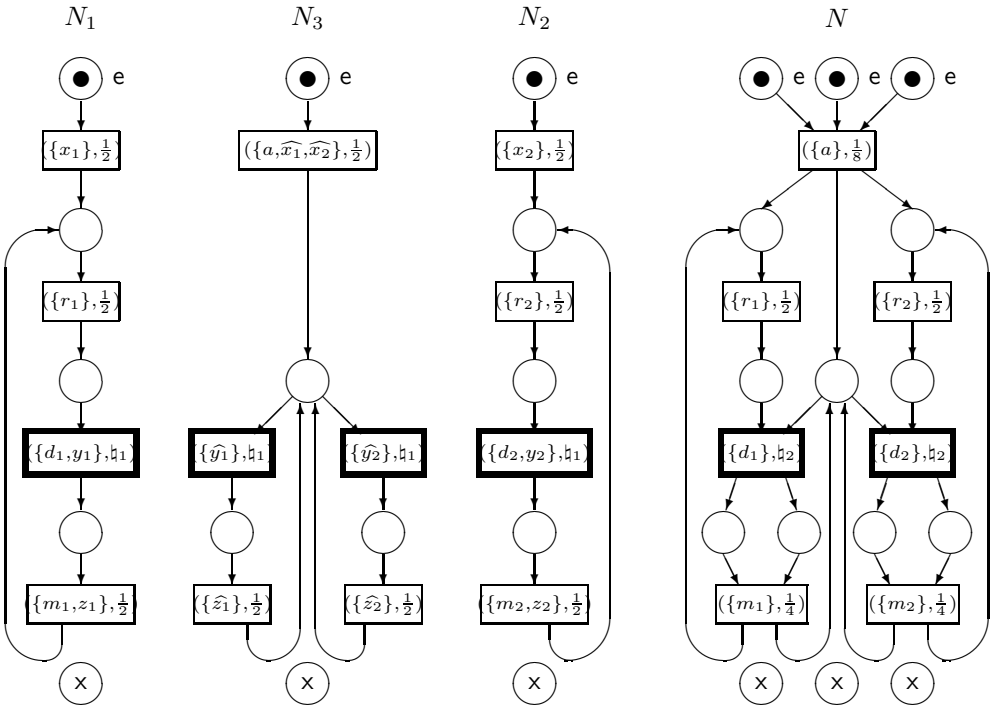

Figure 29: The marked dtsi-boxes of two processors, shared memory and the shared memory system.

- The average recurrence time in the state s_2, where no processor requests the memory, called the *average system run-through*, is $\frac{1}{\varphi_2} = 17$.
- The common memory is available only in the states s_2, s_3, s_4, s_6. The steady-state probability that the memory is available is $\varphi_2 + \varphi_3 + \varphi_4 + \varphi_6 = \frac{1}{17} + 0 + 0 + 0 = \frac{1}{17}$. Then the steady-state probability that the memory is used (i.e. not available), called the *shared memory utilization*, is $1 - \frac{1}{17} = \frac{16}{17}$.
- After activation of the system, we leave the state s_1 for ever, and the common memory is either requested or allocated in every remaining state, with exception of s_2. Thus, the *rate with which the necessity of shared memory emerges* coincides with the rate of leaving s_2, calculated as $\frac{\varphi_2}{SJ_2} = \frac{1}{17} \cdot \frac{3}{4} = \frac{3}{68}$.
- The parallel common memory request of two processors $\{(\{r_1\}, \frac{1}{2}), (\{r_2\}, \frac{1}{2})\}$ is only possible from the state s_2. In this state, the request probability is the sum of the execution probabilities for all multisets of activities containing both $(\{r_1\}, \frac{1}{2})$ and $(\{r_2\}, \frac{1}{2})$. Thus, the *steady-state probability of the shared memory request from two processors* is $\varphi_2 \sum_{\{\Upsilon | (\{(\{r_1\},\frac{1}{2}),(\{r_2\},\frac{1}{2})\} \subseteq \Upsilon\}} PT(\Upsilon, s_2) = \frac{1}{17} \cdot \frac{1}{4} = \frac{1}{68}$.
- The common memory request of the first processor $(\{r_1\}, \frac{1}{2})$ is only possible from the states s_2, s_7. In each of the states, the request probability is the sum of the execution probabilities for all sets of activities containing $(\{r_1\}, \frac{1}{2})$. The *steady-state probability of the shared memory request from the first processor* is $\varphi_2 \sum_{\{\Upsilon | (\{r_1\},\frac{1}{2}) \in \Upsilon\}} PT(\Upsilon, s_2) + \varphi_7 \sum_{\{\Upsilon | (\{r_1\},\frac{1}{2}) \in \Upsilon\}} PT(\Upsilon, s_7) = \frac{1}{17}\left(\frac{1}{4} + \frac{1}{4}\right) + \frac{3}{17}\left(\frac{3}{8} + \frac{1}{8}\right) = \frac{2}{17}$.

In Figure 29, the marked dtsi-boxes corresponding to the dynamic expressions of two processors, shared memory and the shared memory system are presented, i.e. $N_i = Box_{dtsi}(\overline{E_i})$ $(1 \leq i \leq 3)$ and $N = Box_{dtsi}(\overline{E})$.

9.2. The abstract system and its reduction

Let us consider a modification of the shared memory system with abstraction from identifiers of the processors, i.e. such that they 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 shared memory one. 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

$$F_1 = [(\{x_1\}, \frac{1}{2}) * ((\{r\}, \frac{1}{2}); (\{d, y_1\}, \natural_1); (\{m, z_1\}, \frac{1}{2})) * \mathsf{Stop}].$$

The static expression of the second processor is

$$F_2 = [(\{x_2\}, \frac{1}{2}) * ((\{r\}, \frac{1}{2}); (\{d, y_2\}, \natural_1); (\{m, z_2\}, \frac{1}{2})) * \mathsf{Stop}].$$

The static expression of the shared memory is

$$F_3 = [(\{a, \widehat{x_1}, \widehat{x_2}\}, \frac{1}{2}) * ((((\{\widehat{y_1}\}, \natural_1); (\{\widehat{z_1}\}, \frac{1}{2}))[]((\{\widehat{y_2}\}, \natural_1); (\{\widehat{z_2}\}, \frac{1}{2}))) * \mathsf{Stop}].$$

The static expression of the abstract shared memory system with two processors is

$$F = (F_1 \| F_2 \| F_3)\ \mathsf{sr}\ (x_1, x_2, y_1, y_2, z_1, z_2).$$

$DR(\overline{F})$ resembles $DR(\overline{E})$, and $TS(\overline{F})$ is similar to $TS(\overline{E})$. We have $SMC(\overline{F}) \simeq SMC(\overline{E})$. Thus, the average sojourn time vectors of $\overline{F}$ and $\overline{E}$, as well as the TPMs and the steady-state PMFs for $EDTMC(\overline{F})$ and $EDTMC(\overline{E})$, coincide.

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

- The common memory request of a processor $(\{r\}, \frac{1}{2})$ is only possible from the states s_2, s_5, s_7. In each of the states, the request probability is the sum of the execution probabilities for all sets of activities containing $(\{r\}, \frac{1}{2})$. The *steady-state probability of the shared memory request from a processor* is $\varphi_2 \sum_{\{\Upsilon | (\{r\}, \frac{1}{2}) \in \Upsilon\}} PT(\Upsilon, s_2) + \varphi_5 \sum_{\{\Upsilon | (\{r\}, \frac{1}{2}) \in \Upsilon\}} PT(\Upsilon, s_5) + \varphi_7 \sum_{\{\Upsilon | (\{r\}, \frac{1}{2}) \in \Upsilon\}} PT(\Upsilon, s_7) = \frac{1}{17}\left(\frac{1}{4} + \frac{1}{4} + \frac{1}{4}\right) + \frac{3}{17}\left(\frac{3}{8} + \frac{1}{8}\right) + \frac{3}{17}\left(\frac{3}{8} + \frac{1}{8}\right) = \frac{15}{68}$.

The marked dtsi-boxes corresponding to the dynamic expressions of the standard and the abstract two processors and shared memory are similar, as well as the marked dtsi-boxes corresponding to the dynamic expression of the standard and the abstract shared memory systems.

We have $DR(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4, \mathcal{K}_5, \mathcal{K}_6\}$, where $\mathcal{K}_1 = \{s_1\}$ (the initial state), $\mathcal{K}_2 = \{s_2\}$ (the system is activated and the memory is not requested), $\mathcal{K}_3 = \{s_3, s_4\}$ (the memory is requested by one processor), $\mathcal{K}_4 = \{s_5, s_7\}$ (the memory is allocated to a processor), $\mathcal{K}_5 = \{s_6\}$ (the memory is requested by two processors), $\mathcal{K}_6 = \{s_8, s_9\}$ (the memory is allocated to a processor and the memory is requested by another processor).

We also have $DR_T(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_4, \mathcal{K}_6\}$ and $DR_V(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_3, \mathcal{K}_5\}$.

In Figure 30, the quotient transition system $TS_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is presented. In Figure 31, the quotient underlying SMC $SMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is depicted. Note that, in step semantics, we may execute the following multiactions in parallel: $\{r\}, \{r\}$, as well as $\{r\}, \{m\}$. The state $\mathcal{K}_5$ only exists in step semantics, since it is reachable exclusively by executing $\{r\}$ and $\{r\}$ in parallel.

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

$$SJ' = \left(8, \frac{4}{3}, 0, \frac{8}{5}, 0, 4\right).$$

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

$$VAR' = \left(56, \frac{4}{9}, 0, \frac{24}{25}, 0, 12\right).$$

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

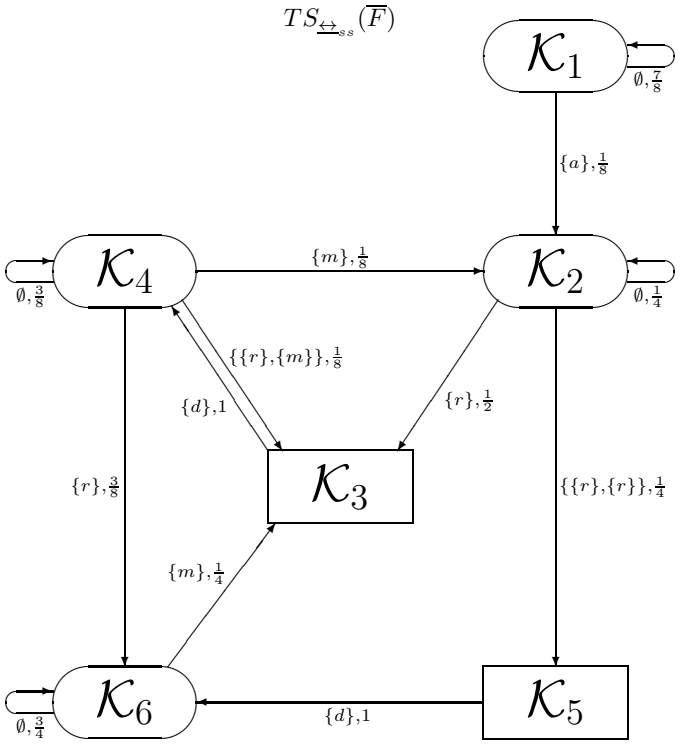

Figure 30: The quotient transition system of the abstract shared memory system.

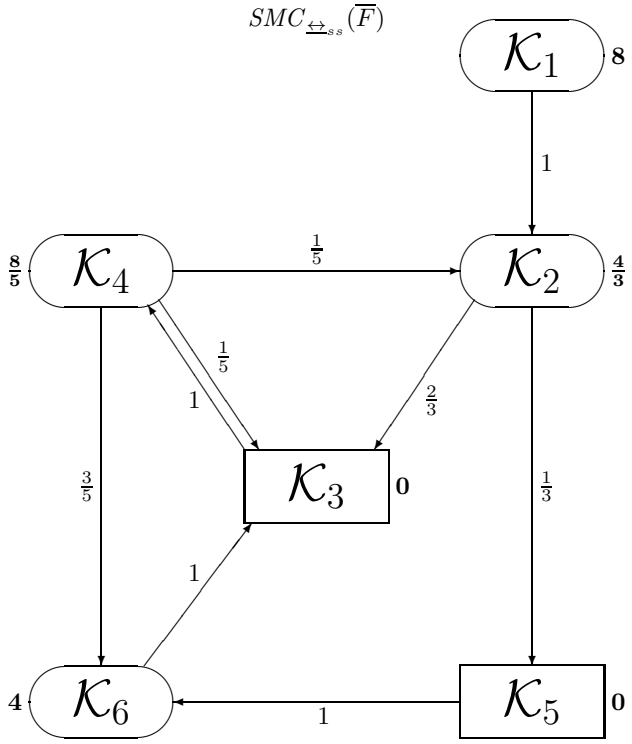

Figure 31: The quotient underlying SMC of the abstract shared memory system.

Table 10: Transient and steady-state probabilities for the quotient EDTMC of the abstract shared memory system.

k	0	5	10	15	20	25	30	35	40	45	50	∞
$\psi_1'^*[k]$	1	0	0	0	0	0	0	0	0	0	0	0
$\psi_2'^*[k]$	0	0	0.0754	0.0859	0.0677	0.0641	0.0680	0.0691	0.0683	0.0680	0.0681	0.0682
$\psi_3'^*[k]$	0	0.4889	0.4633	0.3140	0.3108	0.3452	0.3482	0.3404	0.3392	0.3409	0.3413	0.3409
$\psi_4'^*[k]$	0	0.4667	0.1964	0.3031	0.3719	0.3517	0.3344	0.3380	0.3422	0.3417	0.3407	0.3409
$\psi_5'^*[k]$	0	0.0444	0.0323	0.0179	0.0202	0.0237	0.0234	0.0226	0.0226	0.0228	0.0228	0.0227
$\psi_6'^*[k]$	0	0	0.2325	0.2791	0.2294	0.2154	0.2260	0.2299	0.2277	0.2267	0.2271	0.2273

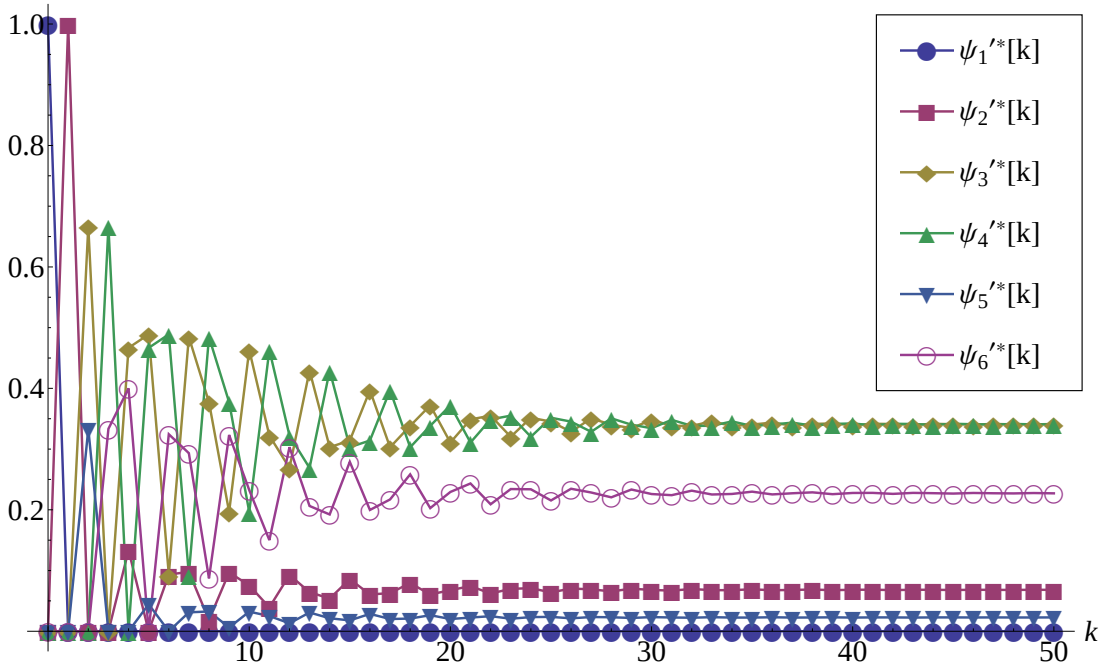

Figure 32: Transient probabilities alteration diagram for the quotient EDTMC of the abstract shared memory system.

$$\mathbf{P}'^* = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{2}{3} & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{1}{5} & \frac{1}{5} & 0 & 0 & \frac{3}{5} \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

In Table 10, the transient and the steady-state probabilities $\psi_i'^*[k]$ $(1 \leq i \leq 6)$ for the quotient EDTMC of the abstract shared memory system at the time moments $k \in \{0, 5, 10, \ldots, 50\}$ and $k = \infty$ are presented, and in Figure 32, the alteration diagram (evolution in time) for the transient probabilities is depicted.

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

$$\psi'^* = \left(0, \frac{3}{44}, \frac{15}{44}, \frac{15}{44}, \frac{1}{44}, \frac{5}{22}\right).$$

The steady-state PMF ψ'^* weighted by SJ' is

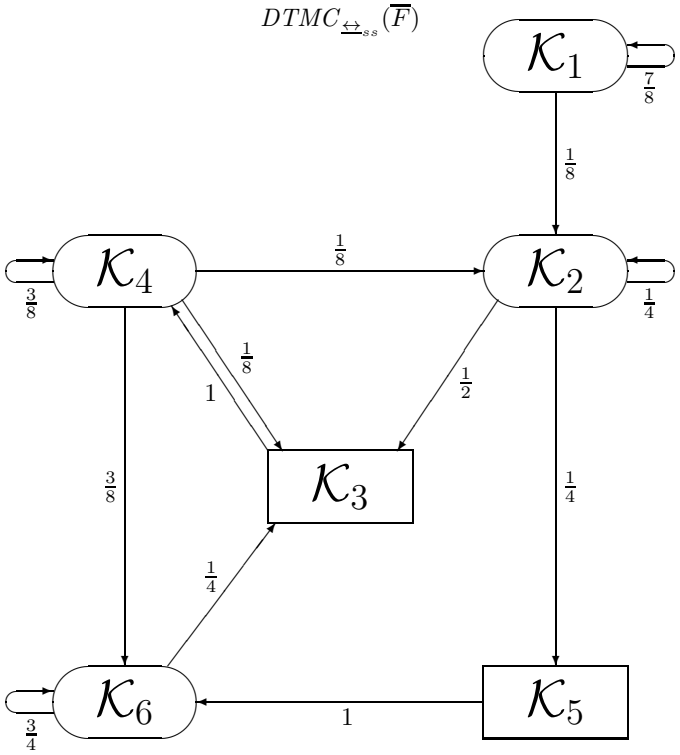

Figure 33: The quotient DTMC of the abstract shared memory system.

$$\left(0, \frac{1}{11}, 0, \frac{6}{11}, 0, \frac{10}{11}\right).$$

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

$$\psi'^{*} S J'^{T} = \frac{17}{11}.$$

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

$$\varphi' = \left(0, \frac{1}{17}, 0, \frac{6}{17}, 0, \frac{10}{17}\right).$$

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

In Figure 33, the quotient DTMC $DTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is depicted.

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

$$\mathbf{P}' = \begin{pmatrix} \frac{7}{8} & \frac{1}{8} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & \frac{1}{2} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{1}{8} & \frac{1}{8} & \frac{3}{8} & 0 & \frac{3}{8} \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{1}{4} & 0 & 0 & \frac{3}{4} \end{pmatrix}.$$

In Table 11, the transient and the steady-state probabilities $\psi'_i[k]$ $(1 \leq i \leq 6)$ for the quotient DTMC of the abstract shared memory system at the time moments $k \in \{0, 5, 10, \ldots, 50\}$ and $k = \infty$ are presented, and in Figure 34, the alteration diagram (evolution in time) for the transient probabilities is depicted.

Table 11: Transient and steady-state probabilities for the quotient DTMC of the abstract shared memory system.

k	0	5	10	15	20	25	30	35	40	45	50	∞
$\psi'_1[k]$	1	0.5129	0.2631	0.1349	0.0692	0.0355	0.0182	0.0093	0.0048	0.0025	0.0013	0
$\psi'_2[k]$	0	0.1161	0.0829	0.0657	0.0569	0.0524	0.0501	0.0489	0.0483	0.0479	0.0478	0.0476
$\psi'_3[k]$	0	0.0944	0.1353	0.1564	0.1672	0.1727	0.1756	0.1770	0.1778	0.1782	0.1784	0.1786
$\psi'_4[k]$	0	0.1162	0.1992	0.2414	0.2630	0.2740	0.2797	0.2826	0.2841	0.2849	0.2853	0.2857
$\psi'_5[k]$	0	0.0311	0.0220	0.0171	0.0146	0.0133	0.0126	0.0123	0.0121	0.0120	0.0120	0.0119
$\psi'_6[k]$	0	0.1294	0.2974	0.3845	0.4292	0.4521	0.4638	0.4698	0.4729	0.4745	0.4753	0.4762

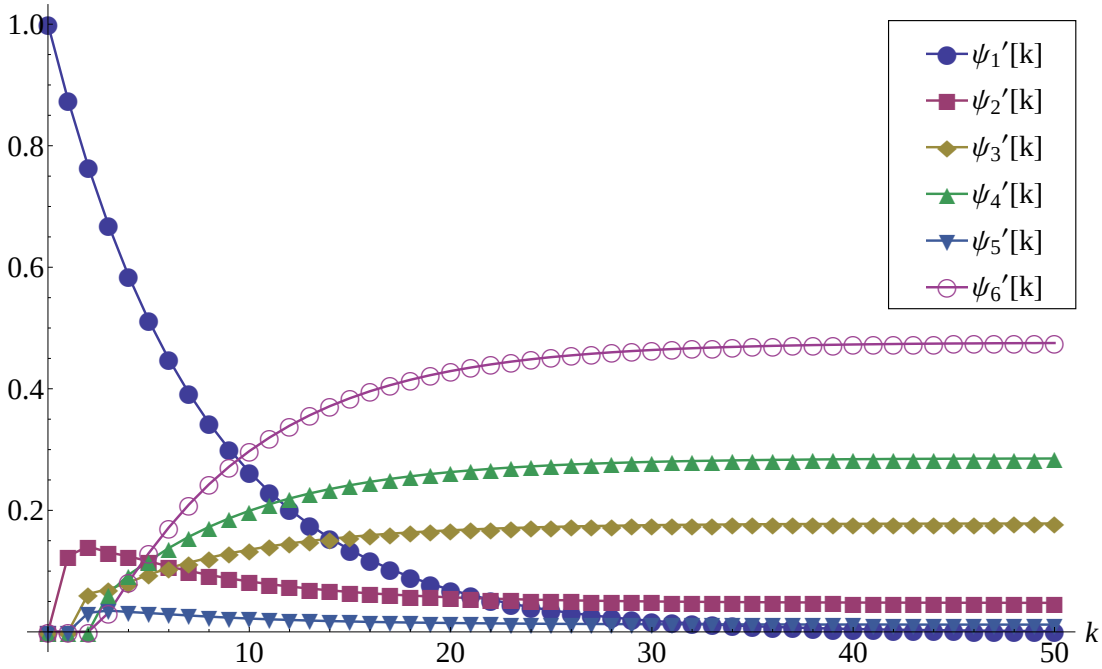

Figure 34: Transient probabilities alteration diagram for the quotient DTMC of the abstract shared memory system.

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

$$\psi' = \left(0, \frac{1}{21}, \frac{5}{28}, \frac{2}{7}, \frac{1}{84}, \frac{10}{21}\right).$$

Remember that $DR_T(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_4, \mathcal{K}_6\}$ and $DR_V(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_3, \mathcal{K}_5\}$. Hence,

$$\sum_{\mathcal{K} \in DR_T(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})}} \psi'(\mathcal{K}) = \psi'(\mathcal{K}_1) + \psi'(\mathcal{K}_2) + \psi'(\mathcal{K}_4) + \psi'(\mathcal{K}_6) = \frac{17}{21}.$$

By the "quotient" analogue of Proposition 5.2, we have

$$\begin{aligned}
&\varphi'(\mathcal{K}_1) = 0 \cdot \tfrac{21}{17} = 0,\\
&\varphi'(\mathcal{K}_2) = \tfrac{1}{21} \cdot \tfrac{21}{17} = \tfrac{1}{17},\\
&\varphi'(\mathcal{K}_3) = 0,\\
&\varphi'(\mathcal{K}_4) = \tfrac{2}{7} \cdot \tfrac{21}{17} = \tfrac{6}{17},\\
&\varphi'(\mathcal{K}_5) = 0,\\
&\varphi'(\mathcal{K}_6) = \tfrac{10}{21} \cdot \tfrac{21}{17} = \tfrac{10}{17}.
\end{aligned}$$

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

$$\varphi' = \left(0, \frac{1}{17}, 0, \frac{6}{17}, 0, \frac{10}{17}\right).$$

This coincides with the result obtained with the use of ψ'^* and SJ'.

Alternatively, from $TS_{\underline{\leftrightarrow}_{ss}}(\overline{F})$, we can construct the reduced quotient DTMC of $\overline{F}$, $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$, and then calculate φ' using it. By Proposition 7.3, it coincides with the quotient RDTMC of $\overline{F}$.

Remember that $DR_T(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_4, \mathcal{K}_6\}$ and $DR_V(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})} = \{\mathcal{K}_3, \mathcal{K}_5\}$. We reorder the elements of $DR(\overline{F})/_{\mathcal{R}_{ss}(\overline{F})}$, by moving the equivalence classes of vanishing states to the first positions: $\mathcal{K}_3, \mathcal{K}_5, \mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_4, \mathcal{K}_6$.

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

$$\mathbf{P}'_r = \begin{pmatrix}
0 & 0 & 0 & 0 & 1 & 0\\
0 & 0 & 0 & 0 & 0 & 1\\
0 & 0 & \frac{7}{8} & \frac{1}{8} & 0 & 0\\
\frac{1}{2} & \frac{1}{4} & 0 & \frac{1}{4} & 0 & 0\\
\frac{1}{8} & 0 & 0 & \frac{1}{8} & \frac{3}{8} & \frac{3}{8}\\
\frac{1}{4} & 0 & 0 & 0 & 0 & \frac{3}{4}
\end{pmatrix}.$$

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

$$\mathbf{C}' = \begin{pmatrix} 0 & 0\\ 0 & 0 \end{pmatrix},\ \mathbf{D}' = \begin{pmatrix} 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1 \end{pmatrix},\ \mathbf{E}' = \begin{pmatrix} 0 & 0\\ \frac{1}{2} & \frac{1}{4}\\ \frac{1}{8} & 0\\ \frac{1}{4} & 0 \end{pmatrix},\ \mathbf{F}' = \begin{pmatrix} \frac{7}{8} & \frac{1}{8} & 0 & 0\\ 0 & \frac{1}{4} & 0 & 0\\ 0 & \frac{1}{8} & \frac{3}{8} & \frac{3}{8}\\ 0 & 0 & 0 & \frac{3}{4} \end{pmatrix}.$$

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

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

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

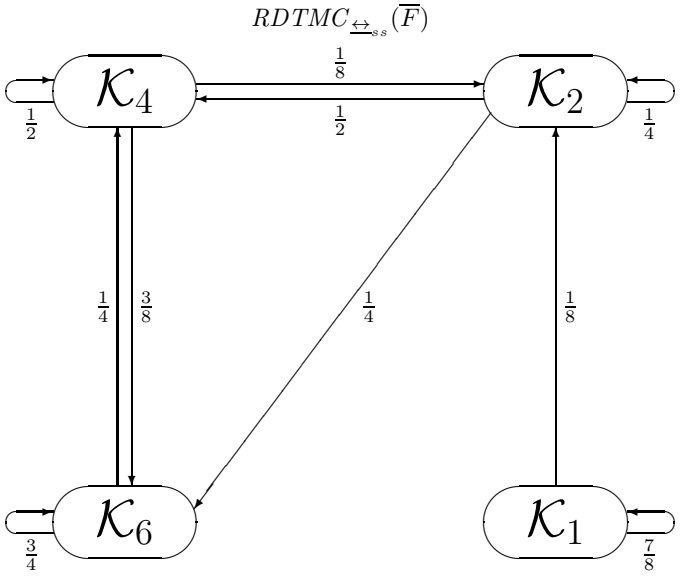

Figure 35: The reduced quotient DTMC of the abstract shared memory system.

Table 12: Transient and steady-state probabilities for the reduced quotient DTMC of the abstract shared memory system.

k	0	5	10	15	20	25	30	35	40	45	50	∞
$\psi'^{\diamond}_1[k]$	1	0.5129	0.2631	0.1349	0.0692	0.0355	0.0182	0.0093	0.0048	0.0025	0.0013	0
$\psi'^{\diamond}_2[k]$	0	0.1244	0.0931	0.0764	0.0679	0.0635	0.0612	0.0600	0.0594	0.0591	0.0590	0.0588
$\psi'^{\diamond}_3[k]$	0	0.1726	0.2614	0.3060	0.3289	0.3406	0.3466	0.3497	0.3513	0.3521	0.3525	0.3529
$\psi'^{\diamond}_4[k]$	0	0.1901	0.3824	0.4826	0.5341	0.5605	0.5740	0.5810	0.5845	0.5863	0.5872	0.5882

$$\mathbf{P}'^{\diamond} = \mathbf{F}' + \mathbf{E}'\mathbf{G}'\mathbf{D}' = \mathbf{F}' + \mathbf{E}'\mathbf{I}\mathbf{D}' = \mathbf{F}' + \mathbf{E}'\mathbf{D}' = \begin{pmatrix} \frac{7}{8} & \frac{1}{8} & 0 & 0 \\ 0 & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & \frac{1}{8} & \frac{1}{2} & \frac{3}{8} \\ 0 & 0 & \frac{1}{4} & \frac{3}{4} \end{pmatrix}.$$

In Figure 35, the reduced quotient DTMC $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is presented.

In Table 12, the transient and the steady-state probabilities $\psi'^{\diamond}_i[k]$ $(1 \le i \le 4)$ for the reduced quotient DTMC of the abstract shared memory system at the time moments $k \in \{0, 5, 10, \ldots, 50\}$ and $k = \infty$ are presented, and in Figure 36, the alteration diagram (evolution in time) for the transient probabilities is depicted.

Then the steady-state PMF for $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is

$$\psi'^{\diamond} = \left(0, \frac{1}{17}, \frac{6}{17}, \frac{10}{17}\right).$$

Note that $\psi'^{\diamond} = (\psi'^{\diamond}(\mathcal{K}_1), \psi'^{\diamond}(\mathcal{K}_2), \psi'^{\diamond}(\mathcal{K}_4), \psi'^{\diamond}(\mathcal{K}_6))$. By the "quotient" analogue of Proposition 5.3, we have

$$\varphi'(\mathcal{K}_1) = 0, \quad \varphi'(\mathcal{K}_2) = \tfrac{1}{17}, \quad \varphi'(\mathcal{K}_3) = 0, \quad \varphi'(\mathcal{K}_4) = \tfrac{6}{17}, \quad \varphi'(\mathcal{K}_5) = 0, \quad \varphi'(\mathcal{K}_6) = \tfrac{10}{17}.$$

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

$$\varphi' = \left(0, \frac{1}{17}, 0, \frac{6}{17}, 0, \frac{10}{17}\right).$$

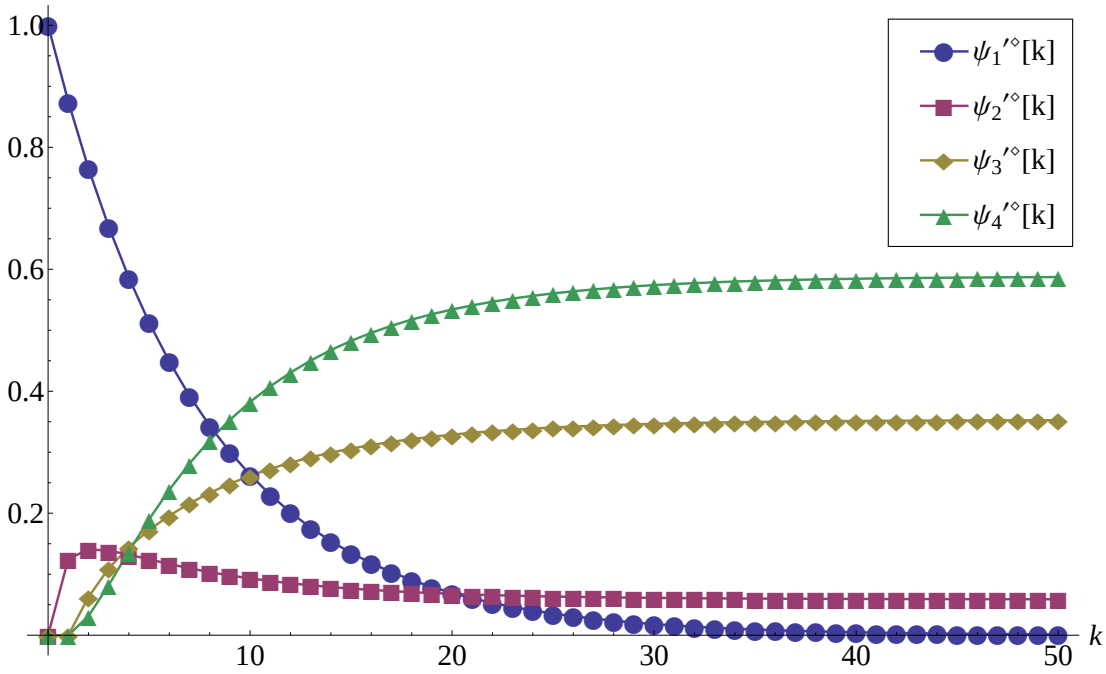

Figure 36: Transient probabilities alteration diagram for the reduced quotient DTMC of the abstract shared memory system.

This coincides with the result obtained with the use of ψ'^{*} and SJ'.
We can now calculate the main performance indices.

- The average recurrence time in the state $\mathcal{K}_2$, where no processor requests the memory, called the *average system run-through*, is $\frac{1}{\varphi'_2} = \frac{17}{1} = 17$.
- The common memory is available only in the states $\mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_5$. The steady-state probability that the memory is available is $\varphi'_2 + \varphi'_3 + \varphi'_5 = \frac{1}{17} + 0 + 0 = \frac{1}{17}$. Then the steady-state probability that the memory is used (i.e. not available), called the *shared memory utilization*, is $1 - \frac{1}{17} = \frac{16}{17}$.
- After activation of the system, we leave the state $\mathcal{K}_1$ for ever, and the common memory is either requested or allocated in every remaining state, with exception of $\mathcal{K}_2$. Thus, the *rate with which the necessity of shared memory emerges* coincides with the rate of leaving $\mathcal{K}_2$, calculated as $\frac{\varphi'_2}{SJ'_2} = \frac{1}{17} \cdot \frac{3}{4} = \frac{3}{68}$.
- The parallel common memory request of two processors $\{\{r\},\{r\}\}$ is only possible from the state $\mathcal{K}_2$. In this state, the request probability is the sum of the execution probabilities for all multisets of multiactions containing $\{r\}$ twice. Thus, the *steady-state probability of the shared memory request from two processors* is $\varphi'_2 \sum_{\{A,\mathcal{K}|\{\{r\},\{r\}\}\subseteq A,\ \mathcal{K}_2 \xrightarrow{A} \mathcal{K}\}} PM_A(\mathcal{K}_2, \mathcal{K}) = \frac{1}{17} \cdot \frac{1}{4} = \frac{1}{68}$.
- The common memory request of a processor $\{r\}$ is only possible from the states $\mathcal{K}_2, \mathcal{K}_4$. In each of the states, the request probability is the sum of the execution probabilities for all multisets of multiactions containing $\{r\}$. The *steady-state probability of the shared memory request from a processor* is $\varphi'_2 \sum_{\{A,\mathcal{K}|\{r\}\in A,\ \mathcal{K}_2 \xrightarrow{A} \mathcal{K}\}} PM_A(\mathcal{K}_2, \mathcal{K}) + \varphi'_4 \sum_{\{A,\mathcal{K}|\{r\}\in A,\ \mathcal{K}_4 \xrightarrow{A} \mathcal{K}\}} PM_A(\mathcal{K}_4, \mathcal{K}) = \frac{1}{17}\left(\frac{1}{2} + \frac{1}{4}\right) + \frac{6}{17}\left(\frac{3}{8} + \frac{1}{8}\right) = \frac{15}{68}$.

One can see that the performance indices are the same for the complete and the quotient abstract shared memory systems. The coincidence of the first, second and third performance indices obviously illustrates the results of Proposition 8.1 and Proposition 8.2 (both modified for $\mathcal{R}_{\mathcal{L}ss}(\overline{F})$). The coincidence of the fourth performance index is due to Theorem 8.1 (modified for $\mathcal{R}_{\mathcal{L}ss}(\overline{F})$): one should just apply its result to the derived step trace $\{\{r\},\{r\}\}$ of the expression $\overline{L}$ and itself. The coincidence of the fifth performance index is due to Theorem 8.1 (modified for $\mathcal{R}_{\mathcal{L}ss}(\overline{F})$):

one should just apply its result to the derived step traces $\{\{r\}\}$, $\{\{r\},\{r\}\}$, $\{\{r\},\{m\}\}$ of the expression $\overline{F}$ and itself, and then sum the left and right parts of the three resulting equalities.

9.3. The generalized system

We now obtain the performance indices taking general values for all multiaction probabilities and weights. Let us suppose that all the mentioned stochastic multiactions have the same generalized probability $\rho \in (0;1)$, and all the immediate ones have the same generalized weight $l \in \boldsymbol{R}_{>0}$. The resulting specification K of the generalized shared memory system is defined as follows.

The static expression of the first processor is

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

The static expression of the second processor is

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

The static expression of the shared memory is

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

The static expression of the generalized shared memory system with two processors is

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

We have $DR_T(\overline{K}) = \{\tilde{s}_1,\tilde{s}_2,\tilde{s}_5,\tilde{s}_5,\tilde{s}_8,\tilde{s}_9\}$ and $DR_V(\overline{K}) = \{\tilde{s}_3,\tilde{s}_4,\tilde{s}_6\}$.
The states are interpreted as follows: $\tilde{s}_1$ is the initial state, $\tilde{s}_2$: the system is activated and the memory is not requested, $\tilde{s}_3$: the memory is requested by the first processor, $\tilde{s}_4$: the memory is requested by the second processor, $\tilde{s}_5$: the memory is allocated to the first processor, $\tilde{s}_6$: the memory is requested by two processors, $\tilde{s}_7$: the memory is allocated to the second processor, $\tilde{s}_8$: the memory is allocated to the first processor and the memory is requested by the second processor, $\tilde{s}_9$: the memory is allocated to the second processor and the memory is requested by the first processor.

In Figure 37, the transition system $TS(\overline{K})$ is presented. In Figure 38, the underlying SMC $SMC(\overline{K})$ is depicted. Note that, in step semantics, we may execute the following activities in parallel: $(\{r_1\},\rho),(\{r_2\},\rho)$, as well as $(\{r_1\},\rho),(\{m_2\},\rho^2)$, and $(\{r_2\},\rho),(\{m_1\},\rho^2)$. The state $\tilde{s}_6$ only exists in step semantics, since it is reachable exclusively by executing $(\{r_1\},\rho)$ and $(\{r_2\},\rho)$ in parallel.

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

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

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

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

The TPM for $EDTMC(\overline{K})$ is

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

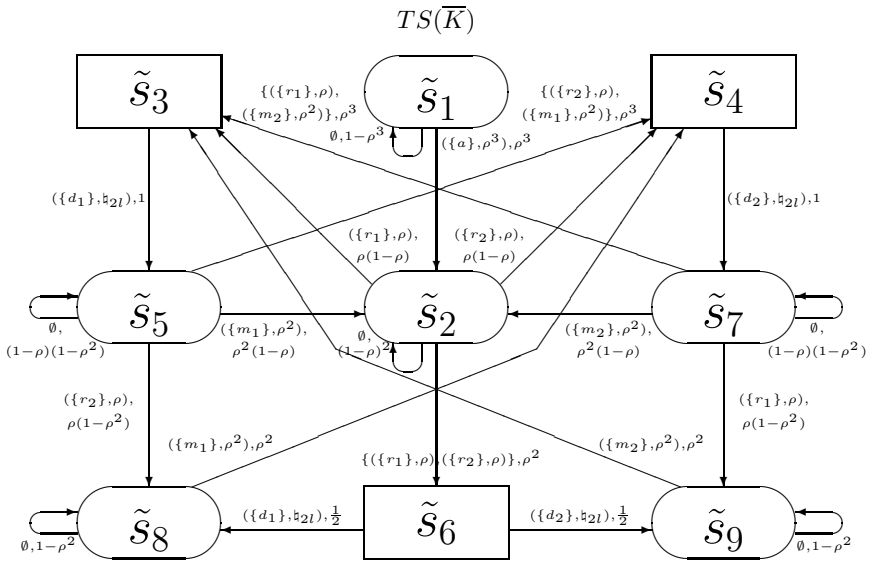

Figure 37: The transition system of the generalized shared memory system.

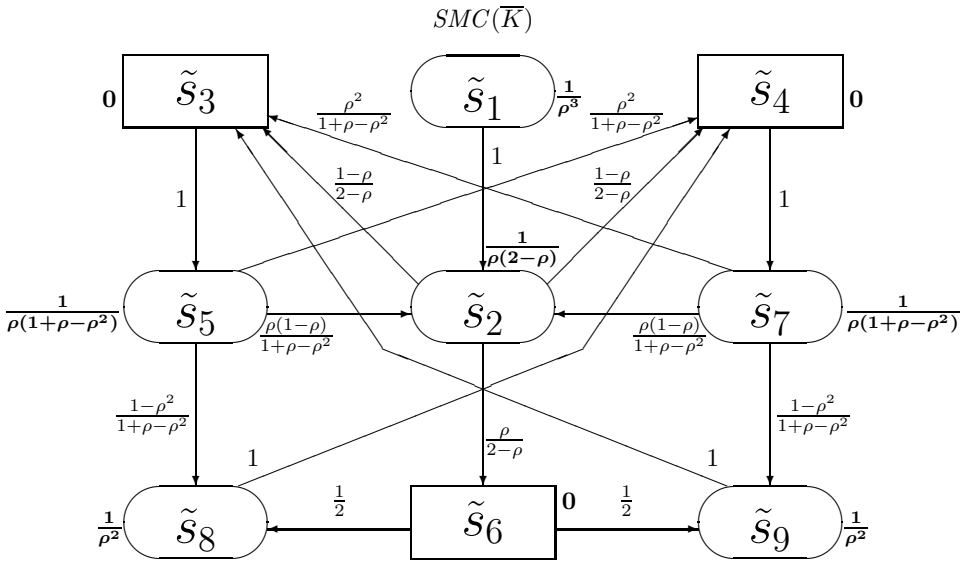

Figure 38: The underlying SMC of the generalized shared memory system.

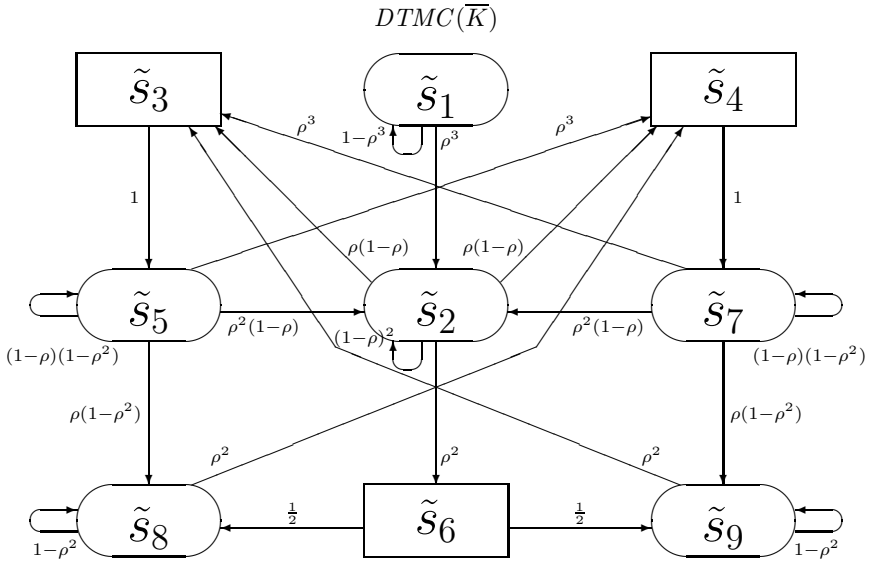

Figure 39: The DTMC of the generalized shared memory system.

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

$$\tilde{\psi}^* = \tfrac{1}{2(6+3\rho-9\rho^2+2\rho^3)}(0, 2\rho(1-\rho)(2-\rho), (2-\rho)(1+\rho-\rho^2), (2-\rho)(1+\rho-\rho^2), (2-\rho)(1+\rho-\rho^2), 2\rho^2(1-\rho),$$
$$(2-\rho)(1+\rho-\rho^2), (2+\rho)(1-\rho), (2+\rho)(1-\rho)).$$

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

$$\frac{1}{2\rho^2(6+3\rho-9\rho^2+2\rho^3)}(0, 2\rho^2(1-\rho), 0, 0, \rho(2-\rho), 0, \rho(2-\rho), (2+\rho)(1-\rho), (2+\rho)(1-\rho)).$$

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

$$\tilde{\psi}^* \widetilde{SJ}^T = \frac{2+\rho-\rho^2-\rho^3}{\rho^2(6+3\rho-9\rho^2+2\rho^3)}.$$

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

$$\tilde{\varphi} = \frac{1}{2(2+\rho-\rho^2-\rho^3)}(0, 2\rho^2(1-\rho), 0, 0, \rho(2-\rho), 0, \rho(2-\rho), (2+\rho)(1-\rho), (2+\rho)(1-\rho)).$$

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 39, the DTMC $DTMC(\overline{K})$ is depicted.
The TPM for $DTMC(\overline{K})$ is

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

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

$$\tilde{\psi} = \frac{1}{2((1+\rho)(2-\rho+2\rho^2-2\rho^3))}(0, 2\rho^2(1-\rho), \rho^2(2-\rho)(1+\rho-\rho^2), \rho^2(2-\rho)(1+\rho-\rho^2), \rho(2-\rho), 2\rho^4(1-\rho), \rho(2-\rho), \\ (2+\rho)(1-\rho), (2+\rho)(1-\rho)).$$

Remember that $DR_T(\overline{K}) = \{\tilde{s}_1, \tilde{s}_2, \tilde{s}_5, \tilde{s}_5, \tilde{s}_8, \tilde{s}_9\}$ and $DR_V(\overline{K}) = \{\tilde{s}_3, \tilde{s}_4, \tilde{s}_6\}$. Hence,

$$\sum_{\tilde{s}\in DR_T(\overline{K})} \tilde{\psi}(\tilde{s}) = \tilde{\psi}(\tilde{s}_1) + \tilde{\psi}(\tilde{s}_2) + \tilde{\psi}(\tilde{s}_5) + \tilde{\psi}(\tilde{s}_7) + \tilde{\psi}(\tilde{s}_8) + \tilde{\psi}(\tilde{s}_9) = \frac{2+\rho-\rho^2-\rho^3}{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}.$$

By Proposition 5.2, we have

$$\begin{aligned}
\tilde{\varphi}(\tilde{s}_1) &= 0 \cdot \tfrac{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}{2+\rho-\rho^2-\rho^3} = 0,\\
\tilde{\varphi}(\tilde{s}_2) &= \tfrac{\rho^2(1-\rho)}{(1+\rho)(2-\rho+2\rho^2-2\rho^3)} \cdot \tfrac{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}{2+\rho-\rho^2-\rho^3} = \tfrac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3},\\
\tilde{\varphi}(\tilde{s}_3) &= 0,\\
\tilde{\varphi}(\tilde{s}_4) &= 0,\\
\tilde{\varphi}(\tilde{s}_5) &= \tfrac{\rho(2-\rho)}{2(1+\rho)(2-\rho+2\rho^2-2\rho^3)} \cdot \tfrac{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}{2+\rho-\rho^2-\rho^3} = \tfrac{\rho(2-\rho)}{2(2+\rho-\rho^2-\rho^3)},\\
\tilde{\varphi}(\tilde{s}_6) &= 0,\\
\tilde{\varphi}(\tilde{s}_7) &= \tfrac{\rho(2-\rho)}{2(1+\rho)(2-\rho+2\rho^2-2\rho^3)} \cdot \tfrac{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}{2+\rho-\rho^2-\rho^3} = \tfrac{\rho(2-\rho)}{2(2+\rho-\rho^2-\rho^3)},\\
\tilde{\varphi}(\tilde{s}_8) &= \tfrac{(2+\rho)(1-\rho)}{2(1+\rho)(2-\rho+2\rho^2-2\rho^3)} \cdot \tfrac{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}{2+\rho-\rho^2-\rho^3} = \tfrac{(2+\rho)(1-\rho)}{2(2+\rho-\rho^2-\rho^3)},\\
\tilde{\varphi}(\tilde{s}_9) &= \tfrac{(2+\rho)(1-\rho)}{2(1+\rho)(2-\rho+2\rho^2-2\rho^3)} \cdot \tfrac{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}{2+\rho-\rho^2-\rho^3} = \tfrac{(2+\rho)(1-\rho)}{2(2+\rho-\rho^2-\rho^3)}.
\end{aligned}$$

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

$$\tilde{\varphi} = \frac{1}{2(2+\rho-\rho^2-\rho^3)}(0, 2\rho^2(1-\rho), 0, 0, \rho(2-\rho), 0, \rho(2-\rho), (2+\rho)(1-\rho), (2+\rho)(1-\rho)).$$

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 $DR_T(\overline{K}) = \{\tilde{s}_1, \tilde{s}_2, \tilde{s}_5, \tilde{s}_7, \tilde{s}_8, \tilde{s}_9\}$ and $DR_V(\overline{K}) = \{\tilde{s}_3, \tilde{s}_4, \tilde{s}_6\}$. We reorder the elements of $DR(\overline{K})$, by moving vanishing states to the first positions: $\tilde{s}_3, \tilde{s}_4, \tilde{s}_6, \tilde{s}_1, \tilde{s}_2, \tilde{s}_5, \tilde{s}_7, \tilde{s}_8, \tilde{s}_9$.

The reordered TPM for $DTMC(\overline{K})$ is

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

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

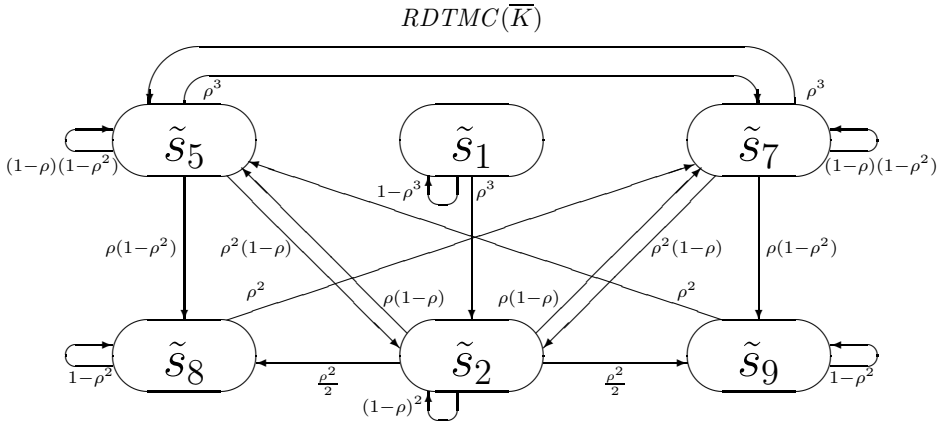

Figure 40: The reduced DTMC of the generalized shared memory system.

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

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

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

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

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

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

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

In Figure 40, the reduced DTMC $RDTMC(\overline{K})$ is presented.
Then the steady-state PMF for $RDTMC(\overline{K})$ is

$$\tilde{\psi}^{\diamond} = \frac{1}{2(2+\rho-\rho^2-\rho^3)}(0, 2\rho^2(1-\rho), \rho(2-\rho), \rho(2-\rho), (2+\rho)(1-\rho), (2+\rho)(1-\rho)).$$

Note that $\tilde{\psi}^{\diamond} = (\tilde{\psi}^{\diamond}(\tilde{s}_1), \tilde{\psi}^{\diamond}(\tilde{s}_2), \tilde{\psi}^{\diamond}(\tilde{s}_5), \tilde{\psi}^{\diamond}(\tilde{s}_7), \tilde{\psi}^{\diamond}(\tilde{s}_8), \tilde{\psi}^{\diamond}(\tilde{s}_9))$. By Proposition 5.3, we have

$\tilde{\varphi}(\tilde{s}_1) = 0, \quad \tilde{\varphi}(\tilde{s}_2) = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3}, \quad \tilde{\varphi}(\tilde{s}_3) = 0, \quad \tilde{\varphi}(\tilde{s}_4) = 0, \quad \tilde{\varphi}(\tilde{s}_5) = \frac{\rho(2-\rho)}{2(2+\rho-\rho^2-\rho^3)},$
$\tilde{\varphi}(\tilde{s}_6) = 0, \quad \tilde{\varphi}(\tilde{s}_7) = \frac{\rho(2-\rho)}{2(2+\rho-\rho^2-\rho^3)}, \quad \tilde{\varphi}(\tilde{s}_8) = \frac{(2+\rho)(1-\rho)}{2(2+\rho-\rho^2-\rho^3)}, \quad \tilde{\varphi}(\tilde{s}_9) = \frac{(2+\rho)(1-\rho)}{2(2+\rho-\rho^2-\rho^3)}.$

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

$$\tilde{\varphi} = \frac{1}{2(2+\rho-\rho^2-\rho^3)}(0, 2\rho^2(1-\rho), 0, 0, \rho(2-\rho), 0, \rho(2-\rho), (2+\rho)(1-\rho), (2+\rho)(1-\rho)).$$

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, called the *average system run-through*, is $\frac{1}{\tilde{\varphi}_2} = \frac{2+\rho-\rho^2-\rho^3}{\rho^2(1-\rho)}$.
- The common memory is available only in the states $\tilde{s}_2, \tilde{s}_3, \tilde{s}_4, \tilde{s}_6$. The steady-state probability that the memory is available is $\tilde{\varphi}_2 + \tilde{\varphi}_3 + \tilde{\varphi}_4 + \tilde{\varphi}_6 = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3} + 0 + 0 + 0 = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3}$. Then the steady-state probability that the memory is used (i.e. not available), called the *shared memory utilization*, is $1 - \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3} = \frac{2+\rho-2\rho^2}{2+\rho-\rho^2-\rho^3}$.
- After activation of the system, we leave the state $\tilde{s}_1$ for ever, and the common memory is either requested or allocated in every remaining state, with exception of $\tilde{s}_2$. Thus, the *rate with which the necessity of shared memory emerges* coincides with the rate of leaving $\tilde{s}_2$, calculated as $\frac{\tilde{\varphi}_2}{\widetilde{SJ}_2} = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3} \cdot \frac{\rho(2-\rho)}{1} = \frac{\rho^3(1-\rho)(2-\rho)}{2+\rho-\rho^2-\rho^3}$.
- 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 of activities containing both $(\{r_1\}, \rho)$ and $(\{r_2\}, \rho)$. Thus, the *steady-state probability of the shared memory request from two processors* is $\tilde{\varphi}_2 \sum_{\{\Upsilon | (\{(\{r_1\},\rho),(\{r_2\},\rho)\} \subseteq \Upsilon\}} PT(\Upsilon, \tilde{s}_2) = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3}\rho^2 = \frac{\rho^4(1-\rho)}{2+\rho-\rho^2-\rho^3}$.
- The common memory request of the first processor $(\{r_1\}, \rho)$ is only possible from the states $\tilde{s}_2, \tilde{s}_7$. In each of the states, 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 $\tilde{\varphi}_2 \sum_{\{\Upsilon | (\{r_1\},\rho) \in \Upsilon\}} PT(\Upsilon, \tilde{s}_2) + \tilde{\varphi}_7 \sum_{\{\Upsilon | (\{r_1\},\rho) \in \Upsilon\}} PT(\Upsilon, \tilde{s}_7) = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3}(\rho(1-\rho)+\rho^2) + \frac{\rho(2-\rho)}{2(2+\rho-\rho^2-\rho^3)}(\rho(1-\rho^2)+\rho^3) = \frac{\rho^2(2+\rho-2\rho^2)}{2(2+\rho-\rho^2-\rho^3)}$.

9.4. The abstract generalized system and its reduction

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

The static expression of the first processor is

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

The static expression of the second processor is

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

The static expression of the shared memory is

$$L_3 = [(\{a, \widehat{x_1}, \widehat{x_2}\}, \rho) * ((((\{\widehat{y_1}\}, \natural_l); (\{\widehat{z_1}\}, \rho))[]((\{\widehat{y_2}\}, \natural_l); (\{\widehat{z_2}\}, \rho))) * \mathsf{Stop}].$$

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

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

$DR(\overline{L})$ 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 $\overline{K}$, as well as 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 again specific to the abstract system.

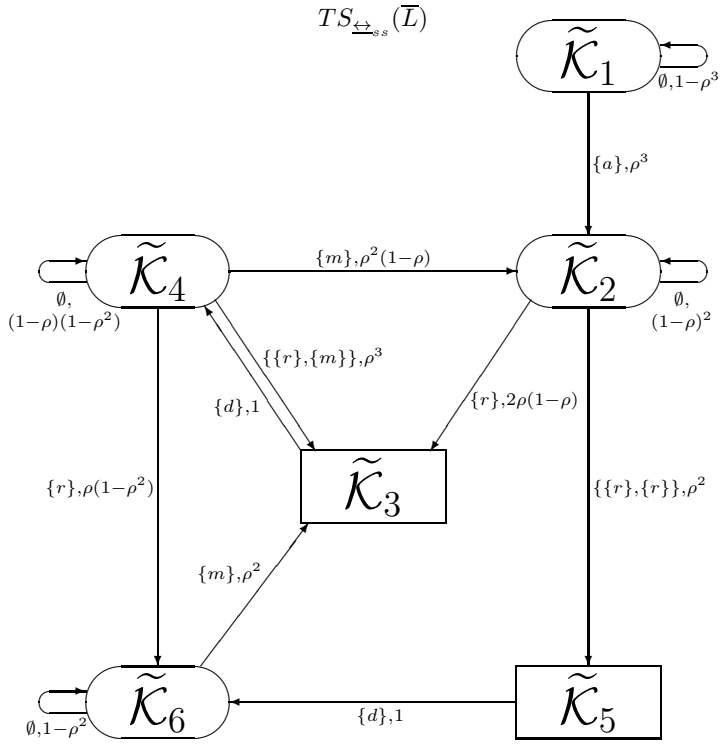

Figure 41: The quotient transition system of the abstract generalized shared memory system.

- The common memory request of a processor $(\{r\},\rho)$ is only possible from the states $\tilde{s}_2, \tilde{s}_5, \tilde{s}_7$. In each of the states, 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 $\tilde{\varphi}_2 \sum_{\{\Upsilon|(\{r\},\rho)\in\Upsilon\}} PT(\Upsilon,\tilde{s}_2) + \tilde{\varphi}_5 \sum_{\{\Upsilon|(\{r\},\rho)\in\Upsilon\}} PT(\Upsilon,\tilde{s}_5) + \tilde{\varphi}_7 \sum_{\{\Upsilon|(\{r\},\rho)\in\Upsilon\}} PT(\Upsilon,\tilde{s}_7) = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3}(\rho(1-\rho)+\rho(1-\rho)+\rho^2) + \frac{\rho(2-\rho)}{2(2+\rho-\rho^2-\rho^3)}(\rho(1-\rho^2)+\rho^3) + \frac{\rho(2-\rho)}{2(2+\rho-\rho^2-\rho^3)}(\rho(1-\rho^2)+\rho^3) = \frac{\rho^2(2-\rho)(1+\rho-\rho^2)}{2+\rho-\rho^2-\rho^3}$.

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

We also have $DR_T(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_1, \widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_4, \widetilde{\mathcal{K}}_6\}$ and $DR_V(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_5\}$.

In Figure 41, the quotient transition system $TS_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is presented. In Figure 42, 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 $\{r\},\{m\}$. The state $\widetilde{\mathcal{K}}_5$ only exists in step semantics, since it is reachable exclusively by executing $\{r\}$ and $\{r\}$ in parallel.

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

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

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

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

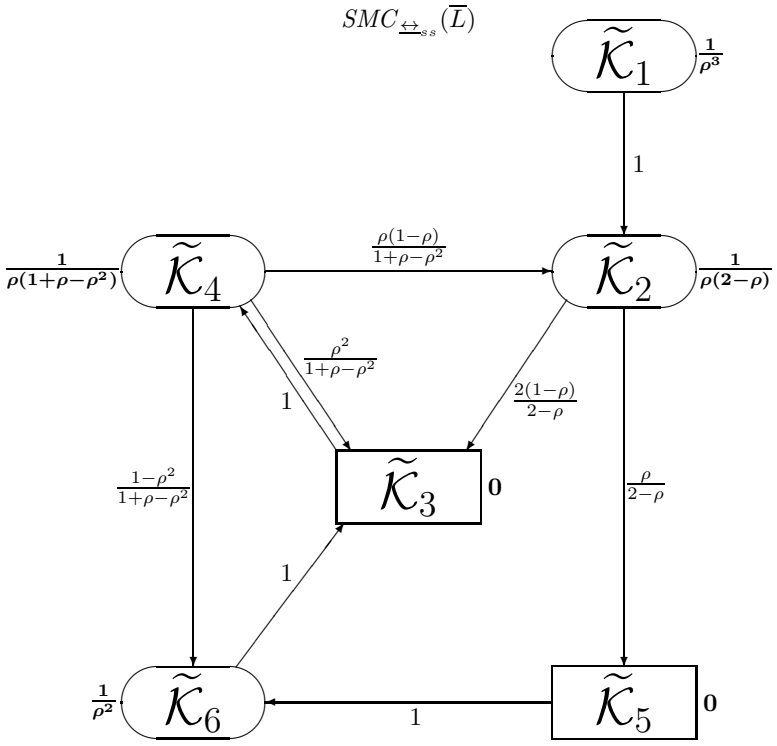

Figure 42: The quotient underlying SMC of the abstract generalized shared memory system.

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

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

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

$$\tilde{\psi}'^{*} = \frac{1}{6+3\rho-9\rho^2+2\rho^3}(0, \rho(1-\rho)(2-\rho), (2-\rho)(1+\rho-\rho^2), (2-\rho)(1+\rho-\rho^2), \rho^2(1-\rho), (2+\rho)(1-\rho)).$$

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

$$\frac{1}{\rho^2(6+3\rho-9\rho^2+2\rho^3)}(0, \rho^2(1-\rho), 0, \rho(2-\rho), 0, (2+\rho)(1-\rho)).$$

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

$$\tilde{\psi}'^{*}\widetilde{SJ}'^{T} = \frac{2+\rho-\rho^2-\rho^3}{\rho^2(6+3\rho-9\rho^2+2\rho^3)}.$$

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

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

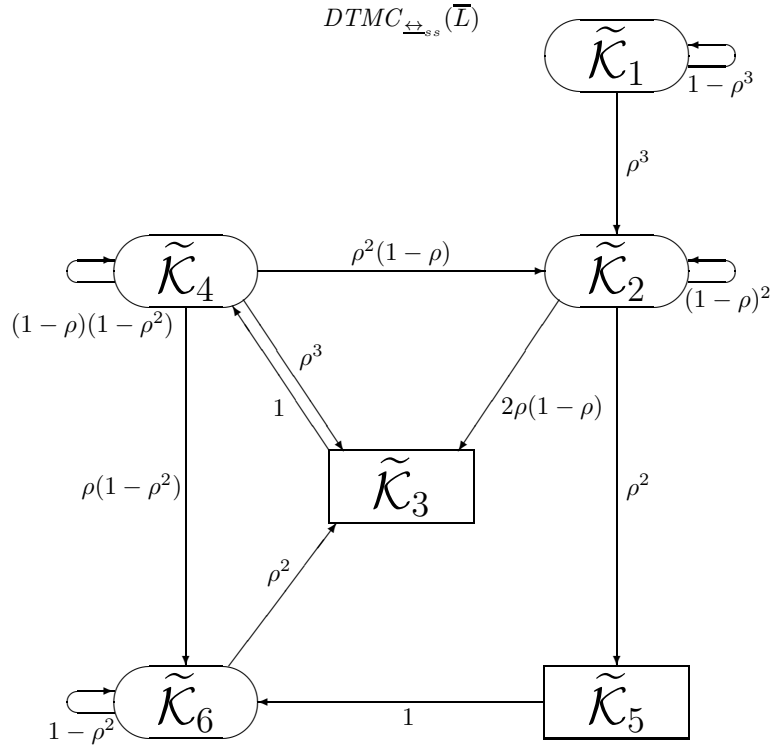

Figure 43: The quotient DTMC of the abstract generalized shared memory system.

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

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

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

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

$$\tilde{\psi}' = \frac{1}{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}(0, \rho^2(1-\rho), \rho^2(2-\rho)(1+\rho-\rho^2), \rho(2-\rho), \rho^4(1-\rho), (2+\rho)(1-\rho)).$$

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

$$\sum_{\widetilde{\mathcal{K}} \in DR_T(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})}} \tilde{\psi}'(\widetilde{\mathcal{K}}) = \tilde{\psi}'(\widetilde{\mathcal{K}}_1) + \tilde{\psi}'(\widetilde{\mathcal{K}}_2) + \tilde{\psi}'(\widetilde{\mathcal{K}}_4) + \tilde{\psi}'(\widetilde{\mathcal{K}}_6) = \frac{2+\rho-\rho^2-\rho^3}{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}.$$

By the "quotient" analogue of Proposition 5.2, we have

$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_1) = 0 \cdot \tfrac{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}{2+\rho-\rho^2-\rho^3} = 0,$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_2) = \tfrac{\rho^2(1-\rho)}{(1+\rho)(2-\rho+2\rho^2-2\rho^3)} \cdot \tfrac{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}{2+\rho-\rho^2-\rho^3} = \tfrac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3},$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_3) = 0,$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_4) = \tfrac{\rho(2-\rho)}{(1+\rho)(2-\rho+2\rho^2-2\rho^3)} \cdot \tfrac{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}{2+\rho-\rho^2-\rho^3} = \tfrac{\rho(2-\rho)}{2+\rho-\rho^2-\rho^3},$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_5) = 0,$$
$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_6) = \tfrac{(2+\rho)(1-\rho)}{(1+\rho)(2-\rho+2\rho^2-2\rho^3)} \cdot \tfrac{(1+\rho)(2-\rho+2\rho^2-2\rho^3)}{2+\rho-\rho^2-\rho^3} = \tfrac{(2+\rho)(1-\rho)}{2+\rho-\rho^2-\rho^3}.$$

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

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

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 7.3, it coincides with the quotient RDTMC of $\overline{L}$.

Remember that $DR_T(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_1, \widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_4, \widetilde{\mathcal{K}}_6\}$ and $DR_V(\overline{L})/_{\mathcal{R}_{ss}(\overline{L})} = \{\widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_5\}$. 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: $\widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_5, \widetilde{\mathcal{K}}_1, \widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_4, \widetilde{\mathcal{K}}_6$.

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

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

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

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

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

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

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

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

$$\widetilde{\mathbf{P}}'^{\diamond} = \widetilde{\mathbf{F}}' + \widetilde{\mathbf{E}}'\widetilde{\mathbf{G}}'\widetilde{\mathbf{D}}' = \widetilde{\mathbf{F}}' + \widetilde{\mathbf{E}}'\mathbf{I}\widetilde{\mathbf{D}}' = \widetilde{\mathbf{F}}' + \widetilde{\mathbf{E}}'\widetilde{\mathbf{D}}' = \begin{pmatrix} 1-\rho^3 & \rho^3 & 0 & 0 \\ 0 & (1-\rho)^2 & 2\rho(1-\rho) & \rho^2 \\ 0 & \rho^2(1-\rho) & 1-\rho-\rho^2+2\rho^3 & \rho(1-\rho^2) \\ 0 & 0 & \rho^2 & 1-\rho^2 \end{pmatrix}.$$

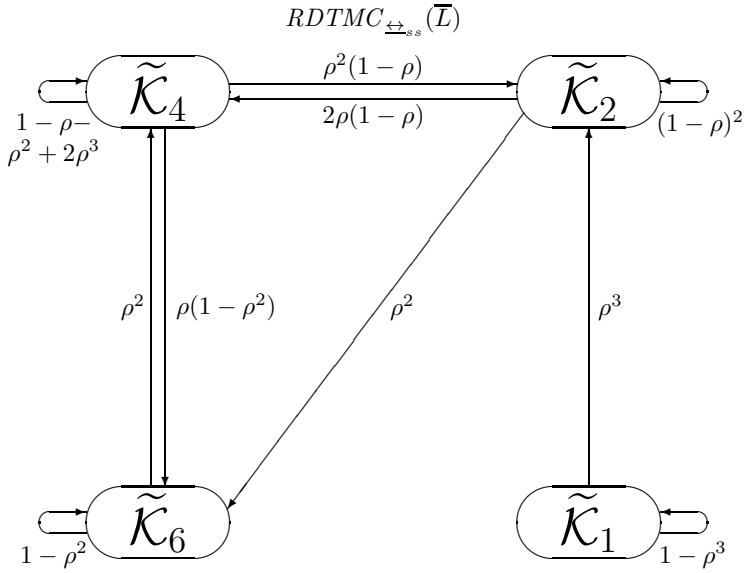

Figure 44: The reduced quotient DTMC of the abstract generalized shared memory system.

In Figure 44, 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

$$\tilde{\psi}'^{\diamond} = \frac{1}{2+\rho-\rho^2-\rho^3}(0, \rho^2(1-\rho), \rho(2-\rho), (2+\rho)(1-\rho)).$$

Note that $\tilde{\psi}'^{\diamond} = (\tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_1), \tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_2), \tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_4), \tilde{\psi}'^{\diamond}(\widetilde{\mathcal{K}}_6))$. By the "quotient" analogue of Proposition 5.3, we have

$$\tilde{\varphi}'(\widetilde{\mathcal{K}}_1) = 0, \quad \tilde{\varphi}'(\widetilde{\mathcal{K}}_2) = \tfrac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3}, \quad \tilde{\varphi}'(\widetilde{\mathcal{K}}_3) = 0, \quad \tilde{\varphi}'(\widetilde{\mathcal{K}}_4) = \tfrac{\rho(2-\rho)}{2+\rho-\rho^2-\rho^3}, \quad \tilde{\varphi}'(\widetilde{\mathcal{K}}_5) = 0, \quad \tilde{\varphi}'(\widetilde{\mathcal{K}}_6) = \tfrac{(2+\rho)(1-\rho)}{2+\rho-\rho^2-\rho^3}.$$

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

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

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, called the *average system run-through*, is $\frac{1}{\tilde{\varphi}'_2} = \frac{2+\rho-\rho^2-\rho^3}{\rho^2(1-\rho)}$.
- The common memory is available only in the states $\widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_3, \widetilde{\mathcal{K}}_5$. The steady-state probability that the memory is available is $\tilde{\varphi}'_2 + \tilde{\varphi}'_3 + \tilde{\varphi}'_5 = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3} + 0 + 0 = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3}$. Then the steady-state probability that the memory is used (i.e. not available), called the *shared memory utilization*, is $1 - \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3} = \frac{2+\rho-2\rho^2}{2+\rho-\rho^2-\rho^3}$.
- After activation of the system, we leave the state $\widetilde{\mathcal{K}}_1$ for ever, and the common memory is either requested or allocated in every remaining state, with exception of $\widetilde{\mathcal{K}}_2$. Thus, the *rate with which the necessity of shared memory emerges* coincides with the rate of leaving $\widetilde{\mathcal{K}}_2$, calculated as $\frac{\tilde{\varphi}'_2}{\widetilde{SJ}'_2} = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3} \cdot \frac{\rho(2-\rho)}{1} = \frac{\rho^3(1-\rho)(2-\rho)}{2+\rho-\rho^2-\rho^3}$.

- The parallel common memory request of two processors $\{\{r\},\{r\}\}$ is only 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. Thus, the *steady-state probability of the shared memory request from two processors* is $\tilde{\varphi}'_2 \sum_{\{A,\widetilde{\mathcal{K}}|\{\{r\},\{r\}\}\subseteq A,\ \widetilde{\mathcal{K}}_2 \xrightarrow{A} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_2,\widetilde{\mathcal{K}}) = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3}\rho^2 = \frac{\rho^4(1-\rho)}{2+\rho-\rho^2-\rho^3}$.

- The common memory request of a processor $\{r\}$ is only possible from the states $\widetilde{\mathcal{K}}_2, \widetilde{\mathcal{K}}_4$. In each of the states, the request probability is the sum of the execution probabilities for all multisets of multiactions containing $\{r\}$. The *steady-state probability of the shared memory request from a processor* is $\tilde{\varphi}'_2 \sum_{\{A,\widetilde{\mathcal{K}}|\{r\}\in A,\ \widetilde{\mathcal{K}}_2 \xrightarrow{A} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_2,\widetilde{\mathcal{K}}) + \tilde{\varphi}'_4 \sum_{\{A,\widetilde{\mathcal{K}}|\{r\}\in A,\ \widetilde{\mathcal{K}}_4 \xrightarrow{A} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_4,\widetilde{\mathcal{K}}) = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3}(2\rho(1-\rho)+\rho^2) + \frac{\rho(2-\rho)}{2+\rho-\rho^2-\rho^3}(\rho(1-\rho^2)+\rho^3) = \frac{\rho^2(2-\rho)(1+\rho-\rho^2)}{2+\rho-\rho^2-\rho^3}$.

One can see that the performance indices are the same for the complete and the quotient abstract generalized shared memory systems. The coincidence of the first, second and third performance indices obviously illustrates the results of Proposition 8.1 and Proposition 8.2 (both modified for $\mathcal{R}_{\mathcal{L}ss}(\overline{L})$). The coincidence of the fourth performance index is due to Theorem 8.1 (modified for $\mathcal{R}_{\mathcal{L}ss}(\overline{L})$): one should just apply its result to the derived step trace $\{\{r\},\{r\}\}$ of the expression $\overline{L}$ and itself. The coincidence of the fifth performance index is due to Theorem 8.1 (modified for $\mathcal{R}_{\mathcal{L}ss}(\overline{L})$): one should just apply its result to the derived step traces $\{\{r\}\}$, $\{\{r\},\{r\}\}$, $\{\{r\},\{m\}\}$ of the expression $\overline{L}$ and itself, and then sum the left and right parts of the three resulting equalities.

Let us consider what is the effect of quantitative changes of the parameter ρ upon performance of the quotient abstract generalized shared memory system in its steady state. Remember that $\rho \in (0;1)$ is the probability of every 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 $\tilde{\varphi}'_1 = \tilde{\varphi}'_3 = \tilde{\varphi}'_5 = 0$, only $\tilde{\varphi}'_2 = \frac{\rho^2(1-\rho)}{2+\rho-\rho^2-\rho^3}$, $\tilde{\varphi}'_4 = \frac{\rho(2-\rho)}{2+\rho-\rho^2-\rho^3}$, $\tilde{\varphi}'_6 = \frac{(2+\rho)(1-\rho)}{2+\rho-\rho^2-\rho^3}$ depend on ρ. In Figure 45, the plots of $\tilde{\varphi}'_2$, $\tilde{\varphi}'_4$, $\tilde{\varphi}'_6$ as functions of ρ are depicted. Notice that, however, we do not allow $\rho = 0$ or $\rho = 1$.

One can see that $\tilde{\varphi}'_2$, $\tilde{\varphi}'_4$ tend to 0 and $\tilde{\varphi}'_6$ 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 the memory is requested by another processor increases, hence, we have *more unsatisfied memory requests*.

Further, $\tilde{\varphi}'_2$, $\tilde{\varphi}'_6$ tend to 0 and $\tilde{\varphi}'_4$ 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 0.0797 of $\tilde{\varphi}'_2$ is reached when $\rho \approx 0.7433$. In this case, the probability that the system is activated and the memory is not requested is maximal, i.e. the *maximal shared memory availability* is about 8%.

In Figure 46, 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 12.5516 is reached when $\rho \approx 0.7433$. To speed up operation of the system, one should take the parameter ρ closer to 0.7433.

The first curve in Figure 47 represents the shared memory utilization, calculated as $1 - \tilde{\varphi}'_2 - \tilde{\varphi}'_3 - \tilde{\varphi}'_5$, as a function of ρ. One can see that the utilization tends to 1 both when ρ approaches 0 and when ρ approaches 1. The minimal value 0.9203 of the utilization is reached when $\rho \approx 0.7433$. Thus, the *minimal shared memory utilization* is about 92%. To increase the utilization, one should take the parameter ρ closer to 0 or 1.

The second curve in Figure 47 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 0.0751 of the rate is reached when $\rho \approx 0.7743$. Thus, the *maximal rate with which the necessity of shared memory emerges* is about $\frac{1}{13}$. To decrease the mentioned rate, one should take the parameter ρ closer to 0 or 1.

The third curve in Figure 47 represents the steady-state probability of the shared memory request from two processors, calculated as $\tilde{\varphi}'_2\widetilde{\mathcal{P}}'_{25}$, where $\widetilde{\mathcal{P}}'_{25} = \sum_{\{A,\widetilde{\mathcal{K}}|\{\{r\},\{r\}\}\subseteq A,\ \widetilde{\mathcal{K}}_2 \xrightarrow{A} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_2,\widetilde{\mathcal{K}}) = PM(\widetilde{\mathcal{K}}_2,\widetilde{\mathcal{K}}_5)$, 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 0.0517 of the probability is reached when $\rho \approx 0.8484$. To decrease the mentioned probability, one should take the parameter ρ closer to 0 or 1.

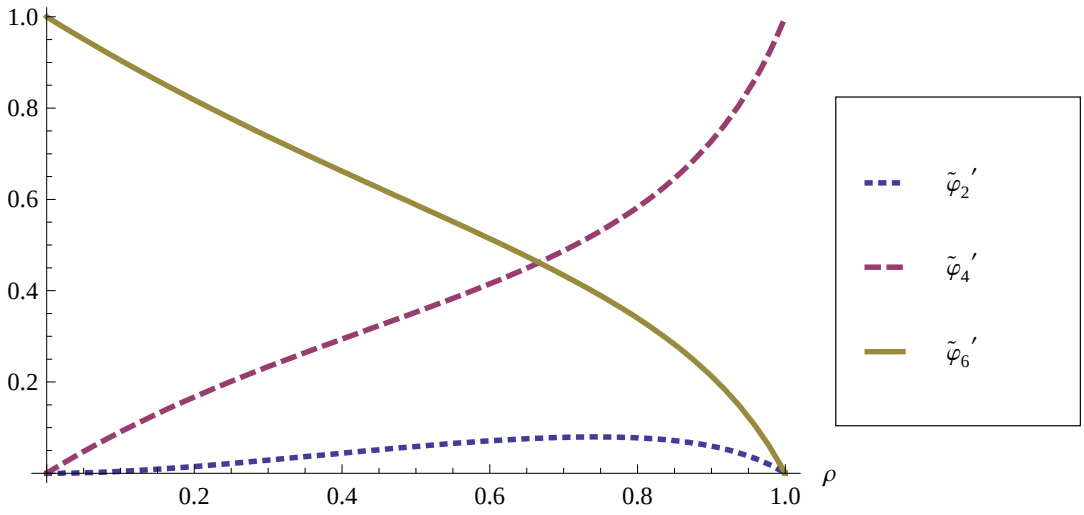

Figure 45: Steady-state probabilities $\tilde{\varphi}_2'$, $\tilde{\varphi}_4'$, $\tilde{\varphi}_6'$ as functions of the parameter ρ.

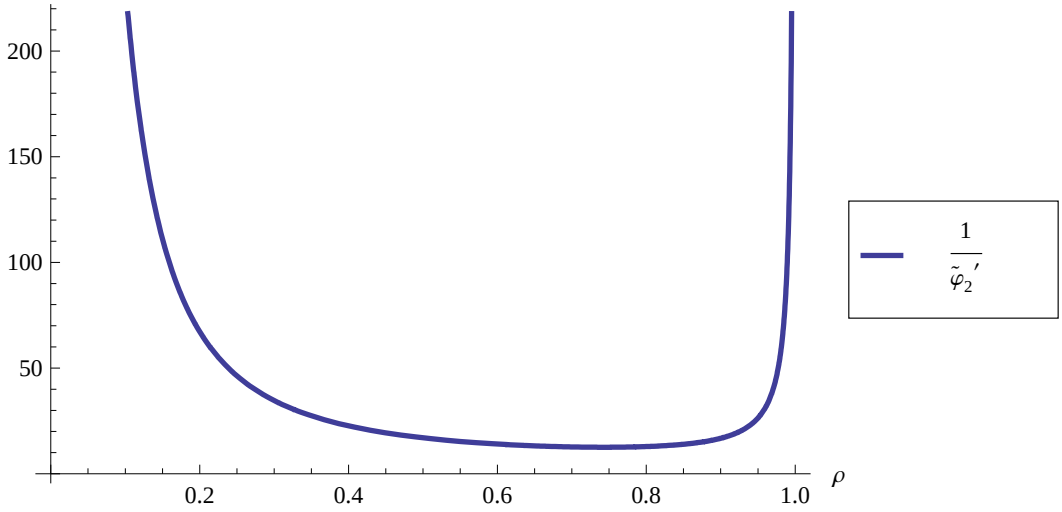

Figure 46: Average system run-through $\frac{1}{\tilde{\varphi}_2'}$ as a function of the parameter ρ.

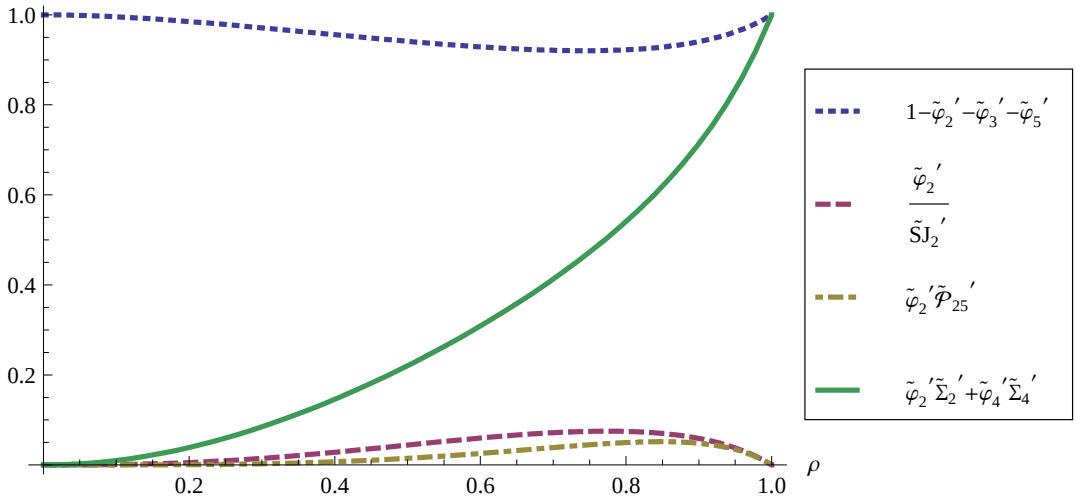

Figure 47: Some performance indices as functions of the parameter ρ.

The fourth curve in Figure 47 represents the steady-state probability of the shared memory request from a processor, calculated as $\tilde{\varphi}'_2\widetilde{\Sigma}'_2 + \tilde{\varphi}'_4\widetilde{\Sigma}'_4$, where $\widetilde{\Sigma}'_i = \sum_{\{A,\widetilde{\mathcal{K}}|\{r\}\in A,\ \widetilde{\mathcal{K}}_i \xrightarrow{A} \widetilde{\mathcal{K}}\}} PM_A(\widetilde{\mathcal{K}}_i, \widetilde{\mathcal{K}}),\ i \in \{2,4\}$, 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 mentioned probability, one should take the parameter ρ closer to 1.

10. Related work

In this section, we consider in detail differences and similarities between dtsiPBC and other well-known or similar SPAs for the purpose of subsequent determining the specific advantages of dtsiPBC.

10.1. Continuous time and interleaving semantics

Let us compare dtsiPBC with classical SPAs: Markovian Timed Processes for Performance Evaluation (MTIPP) [50], Performance Evaluation Process Algebra (PEPA) [52] and Extended Markovian Process Algebra (EMPA) [14].

In MTIPP, every activity is a pair consisting of the action name (including the symbol τ for the *internal*, invisible action) and the parameter of exponential distribution of the action delay (the *rate*). The operations are *prefix*, *choice*, *parallel* composition including *synchronization* on the specified action set and *recursion*. It is possible to specify processes by recursive equations as well. The interleaving semantics is defined on the basis of Markovian (i.e. extended with the specification of rates) labeled transition systems. Note that we have the interleaving behaviour here because the exponential PDF is a continuous one, and a simultaneous execution of any two activities has zero probability according to the properties of continuous distributions. CTMCs can be derived from the mentioned transition systems to analyze performance.

In PEPA, activities are the pairs consisting of action types (including the *unknown*, unimportant type τ) and activity rates. The rate is either the parameter of exponential distribution of the activity duration or it is *unspecified*, denoted by $\top$. An activity with unspecified rate is *passive* by its action type. The set of operations includes *prefix*, *choice*, *cooperation*, *hiding* and constants whose meaning is given by the defining equations including the *recursive* ones. The cooperation is accomplished on the set of action types (the cooperation set) on which the components must *synchronize* or cooperate. If the cooperation set is empty, the cooperation operator turns into the *parallel* combinator. The semantics is interleaving, it is defined via the extension of labeled transition systems with a possibility to specify activity rates. Based on the transition systems, the continuous time Markov processes (CTMPs) are generated which are used for performance evaluation with the help of the embedded continuous time Markov chains (ECTMCs).

In EMPA, each action is a pair consisting of its type and rate. Actions can be *external* or *internal* (denoted by τ) according to types. There are three kinds of actions according to rates: *timed* ones with exponentially distributed

durations (essentially, the actions from MTIPP and PEPA), *immediate* ones with priorities and weights (the actions analogous to immediate transitions of GSPNs) and *passive* ones (similar to passive actions of PEPA). Timed actions specify activities that are relevant for performance analysis. Immediate actions model logical events and the activities that are irrelevant from the performance viewpoint or much faster than others. Passive actions model activities waiting for the synchronization with timed or immediate ones, and express nondeterministic choice. The set of operators consist of *prefix*, functional *abstraction*, functional *relabeling*, *alternative* composition and *parallel* composition ones. Parallel composition includes *synchronization* on the set of action types like in TCSP [53]. The syntax also includes *recursive* definitions given by means of constants. The semantics is interleaving and based on the labeled transition systems enriched with the information about action rates. For the exponentially timed kernel of the algebra (the sublanguage including only exponentially timed and passive actions), it is possible to construct CTMCs from the transition systems of the process terms to analyze the performance.

In dtsiPBC, every activity is a pair consisting of the multiaction (not just an action, as in the classical SPAs) as a first element. The second element is either the probability (not the rate, as in the classical SPAs) to execute the multiaction independently (the activity is called a stochastic multiaction in this case) or the weight expressing how important is the execution of this multiaction (the activity is called an immediate multiaction in this case). Immediate multiactions in dtsiPBC are similar to immediate actions in EMPA, but all the immediate multiactions in dtsiPBC have the same priority 1 (with the purpose to execute them always before stochastic multiactions, all having the same priority 0), whereas the immediate actions in EMPA can have different priority levels. Associating the same priority with all immediate multiactions in dtsiPBC results in the simplified specification and analysis, and such a decision is also appropriate to the calculus. The reason is that, as mentioned in [46], weights (assigned also to immediate actions in EMPA) are enough to denote preferences among immediate multiactions (designating their advantages or prescribing sub-priorities to them) and to produce the conformable probabilistic behaviours when one has to make a choice among several immediate multiactions executable in some state. There are no immediate actions in MTIPP and PEPA. Immediate actions are available only in iPEPA [49], where they are analogous to immediate multiactions in dtsiPBC, and in a variant of TIPP [43] discussed while constructing the calculus PM-TIPP in [90], but there immediate activities are used just to specify probabilistic branching and they cannot be synchronized.

dtsiPBC has the sequence operation, in contrast to the prefix one in the classical SPAs. One can combine arbitrary expressions with the sequence operator, i.e. it is more flexible than the prefix one, where the first argument should be a single activity. The choice operation in dtsiPBC is analogous to that in MTIPP and PEPA, as well as to the alternative composition in EMPA, in the sense that the choice is probabilistic, but a discrete probability function is used in dtsiPBC, unlike continuous ones in the classical calculi. Concurrency and synchronization in dtsiPBC are different operations (this feature is inherited from PBC), unlike the situation in the classical SPAs where parallel composition (combinator) has a synchronization capability. Relabeling in dtsiPBC is analogous to that in EMPA, but it is additionally extended to conjugated actions. The restriction operation in dtsiPBC differs from hiding in PEPA and functional abstraction in EMPA, where the hidden actions are labeled with a symbol of "silent" action τ. In dtsiPBC, restriction by an action means that, for a given expression, any process behaviour containing the action or its conjugate is not allowed. The synchronization on an elementary action in dtsiPBC collects all the pairs consisting of this elementary action and its conjugate which are contained in the multiactions from the synchronized activities. The operation produces new activities such that the first element of every resulting activity is the union of the multiactions from which all the mentioned pairs of conjugated actions are removed. The second element is either the product of the probabilities of the synchronized stochastic multiactions or the sum of the weights of the synchronized immediate multiactions. This differs from the way synchronization is applied in the classical SPAs where it is accomplished over identical action names, and every resulting activity consists of the same action name and the rate calculated via some expression (including sums, minimums and products) on the rates of the initial activities, such as the apparent rate in PEPA. dtsiPBC has no recursion operation or recursive definitions, but it includes the iteration operation to specify infinite looping behaviour with the explicitly defined start and termination.

dtsiPBC has a discrete time semantics, and residence time in the tangible states is geometrically distributed, unlike the classical SPAs with continuous time semantics and exponentially distributed activity delays. As a consequence, the semantics of dtsiPBC is the step one, in contrast to the interleaving semantics of the classical SPAs. The performance is investigated via the underlying SMCs and (reduced) DTMCs extracted from the labeled probabilistic transition systems associated with expressions of dtsiPBC. In the classical SPAs, CTMCs are usually used for performance evaluation. In [40], a denotational semantics of PEPA has been proposed via PEPA nets that are high-level

CTSPNs with coloured tokens (coloured CTSPNs), from which the underlying CTMCs can be retrieved. In [13, 9], a denotational semantics of EMPA based on GSPNs has been defined, from which one can also extract the underlying SMCs and CTMCs (when both immediate and timed transitions are present) or DTMCs (but when there are only immediate transitions). dtsiPBC has a denotational semantics in terms of LDTSIPNs from which the underlying SMCs and (reduced) DTMCs can be derived.

10.2. Continuous time and non-interleaving semantics

Only a few non-interleaving SPAs were considered among non-Markovian ones [60, 22]. The semantics of all Markovian calculi is interleaving and their action delays have exponential distribution, which is the only continuous probability distribution with memoryless (Markovian) property.

In [24], Generalized Stochastic Process Algebra (GSPA) was introduced. It has a true-concurrent denotational semantics in terms of generalized stochastic event structures (GSESs) with non-Markovian stochastic delays of events. In that paper, no operational semantics or performance evaluation methods for GSPA were presented. Later, in [59], generalized semi-Markov processes (GSMPs) were extracted from GSESs to analyze performance.

In [87, 88], generalized Stochastic π-calculus (Sπ) with general continuous distributions of activity delays was defined. It has a proved operational semantics with transitions labeled by encodings of their deduction trees. No well-established underlying performance model for this version of Sπ was described.

In [21, 20], Generalized Semi-Markovian Process Algebra (GSMPA) was developed with an ST-operational semantics and non-Markovian action delays. The performance analysis in GSMPA is accomplished via GSMPs.

Again, the first fundamental difference between dtsiPBC and the calculi GSPA, Sπ and GSMPA is that dtsiPBC is based on PBC, whereas GSPA is an extension of simple Process Algebra (PA) from [24], Sπ extends π-calculus [78, 79] and GSMPA is an enrichment of EMPA. Therefore, both GSPA and GSMPA have *prefixing*, *choice* (*alternative* composition), *parallel* composition, *renaming* (*relabeling*) and *hiding* (*abstraction*) operations, but only GSMPA permits *constants*. Unlike dtsiPBC, GSPA has neither iteration or recursion, GSMPA allows only *recursive* definitions, whereas Sπ additionally has operations to specify *mobility*. Note also that GSPA, Sπ and GSMPA do not specify instantaneous events or activities while dtsiPBC has immediate multiactions.

The second significant difference is that geometrically distributed or zero delays are associated with process states in dtsiPBC, unlike generally distributed delays assigned to events in GSPA or to activities in Sπ and GSMPA. As a consequence, dtsiPBC has a discrete time operational semantics allowing for concurrent execution of activities in steps. GSPA has no operational semantics while Sπ and GSMPA have continuous time ones. In continuous time semantics, concurrency is simulated by interleaving, since simultaneous occurrence of any two events has zero probability according to the properties of continuous probability distributions. Therefore, interleaving transitions are often annotated with an additional information to keep concurrency data. The transition labels in the operational semantics of Sπ encode the action causality information and allow one to derive the enabling relations and the firing distributions of concurrent transitions from the transition sequences. At the same time, abstracting from stochastic delays leads to the classical early interleaving semantics of π-calculus. The ST-operational semantics of GSMPA is based on decorated transition systems governed by transition rules with rather complex preconditions. There are two types of transitions: the choice (action beginning) and the termination (action ending) ones. The choice transitions are labeled by weights of single actions chosen for execution while the termination transitions have no labels. Only single actions can begin, but several actions can end in parallel. Thus, the choice transitions happen just sequentially while the termination transitions can happen simultaneously. As a result, the decorated interleaving / step transition systems are obtained. dtsiPBC has an SPN-based denotational semantics. In comparison with event structures, PNs are more expressive and visually tractable formalism, capable of finitely specifying an infinite behaviour. Recursion in GSPA produces infinite GSESs while dtsiPBC has iteration operation with a finite SPN semantics. Identification of infinite GSESs that can be finitely represented in GSPA was left for a future research.

10.3. Discrete time

In [1], a class of compositional DTSPNs with generally distributed discrete time transition delays was proposed, called dts-nets. The denotational semantics of a stochastic extension (we call it stochastic ACP or sACP) of a subset of Algebra of Communicating Processes (ACP) [8] can be constructed via dts-nets. There are two types of transitions in dts-nets: immediate (timeless) ones, with zero delays, and time ones, whose delays are random variables having

general discrete distributions. The top-down synthesis of dts-nets consists in the substitution of their transitions by blocks (dts-subnets) corresponding to the sequence, choice, parallelism and iteration operators. It was explained how to calculate the throughput time of dts-nets using the service time (defined as holding time or delay) of their transitions. For this, the notions of service distribution for the transitions and throughput distribution for the building blocks were defined. Since the throughput time of the parallelism block was calculated as the maximal service time for its two constituting transitions, the analogue of the step semantics approach was implemented.

In [71, 72], an SPA called Theory of Communicating Processes with discrete stochastic time (TCP^{dst}) was introduced, later in [69] called Theory of Communicating Processes with discrete real and stochastic time (TCP^{drst}). It has discrete real time (deterministic) delays (including zero delays) and discrete stochastic time delays. The algebra generalizes real time processes to discrete stochastic time ones by applying real time properties to stochastic time and imposing race condition to real time semantics. TCP^{dst} has an interleaving operational semantics in terms of stochastic transition systems. The performance is analyzed via discrete time probabilistic reward graphs which are essentially the reward transition systems with probabilistic states having finite number of outgoing probabilistic transitions and timed states having a single outgoing timed transition. The mentioned graphs can be transformed by unfolding or geometrization into discrete time Markov reward chains (DTMRCs) appropriate for transient or stationary analysis.

The first difference between dtsiPBC and the algebras sACP and TCP^{dst} is that dtsiPBC is based on PBC, but sACP and TCP^{dst} are the extensions of ACP [8]. sACP has taken from ACP only *sequence*, *choice*, *parallelism* and *iteration* operations, whereas dtsiPBC has additionally relabeling, restriction and synchronization ones, inherited from PBC. In TCP^{dst}, besides standard action *prefixing*, *alternative*, *parallel* composition, *encapsulation* (similar to *restriction*) and *recursive* variables, there are also *timed delay prefixing*, *dependent delays scope* and the *maximal time progress* operators, which are new both for ACP and dtsiPBC.

The second difference is that dtsiPBC, sACP and TCP^{dst}, all have zero delays, however, discrete time delays in dtsiPBC are zeros or geometrically distributed and associated with process states. The zero delays are possible just in vanishing states while geometrically distributed delays are possible only in tangible states. For each tangible state, the parameter of geometric distribution governing the delay in the state is completely determined by the probabilities of all stochastic multiactions executable from it. In sACP and TCP^{dst}, delays are generally distributed, but they are assigned to transitions in sACP and separated from actions (excepting zero delays) in TCP^{dst}. Moreover, a special attention is given to zero delays in sACP and deterministic delays in TCP^{dst}. In sACP, immediate (timeless) transitions with zero delays serve as source and sink transitions of the dts-subnets corresponding to the choice, parallelism and iteration operators. In TCP^{dst}, zero delays of actions are specified by undelayable action prefixes while positive deterministic delays of processes are specified with timed delay prefixes. Neither formal syntax nor operational semantics for sACP are defined and it is not explained how to derive Markov chains from the algebraic expressions or the corresponding dts-nets to analyze performance. It is not stated explicitly, which type of semantics (interleaving or step) is accommodated in sACP. In spite of the discrete time approach, operational semantics of TCP^{dst} is still interleaving, unlike that of dtsiPBC. In addition, no denotational semantics was defined for TCP^{dst}.

Let us mention other SPAs with discrete time and interleaving semantics. Those without immediate actions are: Weighted Synchronous Calculus of Communicating Systems (WSCCS) [100, 101] and discrete-time variant of stochastic Concurrent Constraint Programming (we call it $dsCCP$) [18].

That with immediate actions is: Interactive Probabilistic Chains (IPC) [34, 47].

The three SPAs are rather specific: unlike standard approach, weights in WSCCS, rates (weights) in dsCCP and probabilities in IPC are not associated with actions. In dsCCP, probabilities are calculated using rates (weights) that are assigned to operations. In IPC, actions are executed instantaneously while probabilistic choices take one unit time.

Table 13 summarizes the SPAs comparison above and that from Section 1 (the calculi sPBC, gsPBC and dtsPBC), by classifying the SPAs according to the concept of time, the presence of immediate (multi)actions and the type of operational semantics. The names of SPAs, whose denotational semantics is based on SPNs, are printed in bold font. The underlying stochastic process (if defined) is specified in parentheses near the name of the corresponding SPA.

11. Discussion

Let us now discuss which advantages has dtsiPBC in comparison with the SPAs described in Section 10.

Table 13: Classification of stochastic process algebras.

Time	Immediate (multi)actions	Interleaving semantics	Non-interleaving semantics
Continuous	No	MTIPP (CTMC), **PEPA** (CTMP), **sPBC** (CTMC)	GSPA (GSMP), Sπ, GSMPA (GSMP)
	Yes	**EMPA** (SMC, CTMC), **gsPBC** (SMC)	—
Discrete	No	WSCCS (DTMC), dsCCP (DTMC)	**dtsPBC** (DTMC)
	Yes	TCP^{dst} (DTMRC), IPC (DTMC)	**sACP**, **dtsiPBC** (SMC, DTMC)

11.1. Analytical solution

An important aspect is the analytical tractability of the underlying stochastic process, used for performance evaluation in SPAs. The underlying CTMCs in MTIPP and PEPA, as well as SMCs in EMPA, are treated analytically, but these continuous time SPAs have interleaving semantics. GSPA, Sπ and GSMPA are the continuous time models, for which a non-interleaving semantics is constructed, but for the underlying GSMPs in GSPA and GSMPA, only simulation and numerical methods are applied, whereas no performance model for Sπ is defined. sACP and TCP^{dst} are the discrete time models with the associated analytical methods for the throughput calculation in sACP or for the performance evaluation based on the underlying DTMRCs in TCP^{dst}, but both models have interleaving semantics. dtsiPBC is a discrete time model with a non-interleaving semantics, where analytical methods are applied to the underlying SMCs. Hence, if an interleaving model is appropriate as a framework for the analytical solution towards performance evaluation then one has a choice between the continuous time SPAs MTIPP, PEPA, EMPA and the discrete time ones sACP, TCP^{dst}. Otherwise, if one needs a non-interleaving model with the associated analytical methods for performance evaluation and the discrete time approach is feasible then dtsiPBC is the right choice.

The existence of an analytical solution also permits to interpret quantitative values (rates, probabilities, weights etc.) from the system specifications as parameters, which can be adjusted to optimize the system performance, like in dtsPBC and dtsiPBC. The DTMCs whose transition probabilities are parameters were introduced in [35]. The parameters can also be adjusted in parametric probabilistic transition systems (i.e. DTMCs whose transition probabilities may be real-valued parameters) [63]. Parametric CTMCs with the transition rates treated as parameters were investigated in [45]. On the other hand, no parameters in formulas of SPAs were considered in the literature so far. In dtsiPBC we can easily construct examples with more parameters than we did in our case study. The performance indices will be then interpreted as functions of several variables. The advantage of our approach is that, unlike of the method from [63], we should not impose to the parameters any special conditions needed to guarantee that the real values, interpreted as the transition probabilities, always lie in the interval $[0; 1]$. To be convinced of this fact, just remember that, as we have demonstrated, the positive probability functions PF, PT, PM, PM^*, $PM^\diamond$ define probability distributions, hence, they always return values belonging to $(0; 1]$ for any probability parameters from $(0; 1)$ and weight parameters from $\boldsymbol{R}_{>0}$. In addition, the transition constraints (their probabilities, rates and guards), calculated using the parameters, in our case should not always be polynomials over variables-parameters, as often required in the mentioned papers, but they may also be fractions of polynomials, like in our case study.

11.2. Concurrency interpretation

One can see that the stochastic process calculi proposed in the literature are based on interleaving, as a rule, and parallelism is simulated by synchronous or asynchronous execution. As a semantic domain, the interleaving formalism of transition systems is often used. However, to properly support intuition of the behaviour of concurrent and distributed systems, their semantics should treat parallelism as a primitive concept that cannot be reduced to nondeterminism. Moreover, in interleaving semantics, some important properties of these systems cannot be expressed, such as simultaneous occurrence of concurrent transitions [36] or local deadlock in the spatially distributed processes [82]. Therefore, investigation of stochastic extensions for more expressive and powerful algebraic calculi is an important issue. The development of step or "true concurrency" (such that parallelism is considered as a causal independence) SPAs is an interesting and nontrivial problem, which has attracted special attention last years. Nevertheless, not so

many formal stochastic models of parallel systems were defined whose underlying stochastic processes are based on DTMCs. As mentioned in [39], such models are more difficult to analyze, since several events can occur simultaneously in discrete time systems (the models have a step semantics) and the probability of a set of events cannot be easily related to the probability of the single ones. Therefore, parallel executions of actions are often not considered also in the discrete time SPAs, such as TCP^{dst}, whose underlying stochastic process is DTMCs with rewards (DTMRCs). As observed in [54], even for stochastic models with generally distributed time delays, some restrictions on the concurrency degree were imposed to simplify their analysis techniques. In particular, the enabling restriction requires that no two generally distributed transitions are enabled in any reachable marking. Hence, their activity periods do not intersect and no two such transitions can fire simultaneously, this results in interleaving semantics of the model.

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

Example 11.1. *In Figure 48, the interleaving transition system of the generalized shared memory system $ts(\overline{K})$ from Section 9 is presented. The transition system $ts(\overline{K})$ is constructed from the (step) one $TS(\overline{K})$ in Figure 37 as follows. First, all the transitions due to executing more than one activity are removed. Second, the states that become non-reachable (from the initial state) in the absence of such "parallel" transitions are deleted, together with all the transitions from these states. Third, for each of the states left, the remaining outgoing transition probabilities are normalized. Formally, the probabilities of the remaining transitions are defined as follows. Let G be a dynamic expression, $s \in DR(G)$, $\Upsilon \in Exec(s)$ and $|\Upsilon| \leq 1$. The* probability to execute the multiset of activities Υ in s, when only zero-element steps (i.e. empty loops) or one-element steps are allowed, *is $pt(\Upsilon, s) = \frac{PT(\Upsilon,s)}{\sum_{\{\Xi \mid |\Xi| \leq 1\}} PT(\Xi,s)}$. As a result, one can see many serious differences between $ts(\overline{K})$ and $TS(\overline{K})$, i.e. between the system's behaviour in the interleaving and step semantics. One can define interleaving stochastic bisimulation equivalence $\underline{\leftrightarrow}_{is}$ analogously to $\underline{\leftrightarrow}_{ss}$, but using the interleaving transition systems of expressions instead of the standard (step) ones. Then, from $ts(\overline{K})$, the interleaving quotient (by $\underline{\leftrightarrow}_{is}$) transition system of the abstract generalized shared memory system $ts_{\underline{\leftrightarrow}_{is}}(\overline{L})$ can be constructed, depicted in Figure 49. Again, there exist substantial differences between $ts_{\underline{\leftrightarrow}_{is}}(\overline{L})$ and $TS_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ in Figure 41. Next, from $ts_{\underline{\leftrightarrow}_{is}}(\overline{L})$, the interleaving reduced quotient DTMC of the abstract generalized shared memory system $rdtmc_{\underline{\leftrightarrow}_{is}}(\overline{L})$ can be obtained, shown in Figure 50. Clearly, there are severe differences between $rdtmc_{\underline{\leftrightarrow}_{is}}(\overline{L})$ and $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ in Figure 44. Then the steady-state PMF for $rdtmc_{\underline{\leftrightarrow}_{is}}(\overline{L})$ is $\tilde{\phi}'^{\diamond} = \frac{1}{2+4\rho+3\rho^2+3\rho^3}(0, \rho^2(1+\rho), 2\rho(1+\rho+\rho^2), 2(1+\rho))$, whereas the steady-state PMF for $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is $\tilde{\psi}'^{\diamond} = \frac{1}{2+\rho-\rho^2-\rho^3}(0, \rho^2(1-\rho), \rho(2-\rho), (2+\rho)(1-\rho))$. From $ts_{\underline{\leftrightarrow}_{is}}(\overline{L})$, the interleaving quotient SMC of the abstract generalized shared memory system $smc_{\underline{\leftrightarrow}_{is}}(\overline{L})$ can also be extracted, depicted in Figure 51. There are serious differences between $smc_{\underline{\leftrightarrow}_{is}}(\overline{L})$ and $SMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ in Figure 42. Then the steady-state PMF for $smc_{\underline{\leftrightarrow}_{is}}(\overline{L})$ is $\tilde{\phi}' = \frac{1}{2+4\rho+3\rho^2+3\rho^3}(0, \rho^2(1+\rho), 0, 2\rho(1+\rho+\rho^2), 0, 2(1+\rho))$, whereas the steady-state PMF for $SMC_{\underline{\leftrightarrow}_{ss}}(\overline{L})$ is $\tilde{\varphi}' = \frac{1}{2+\rho-\rho^2-\rho^3}(0, \rho^2(1-\rho), 0, \rho(2-\rho), 0, (2+\rho)(1-\rho))$. In Figure 52, the plots of*

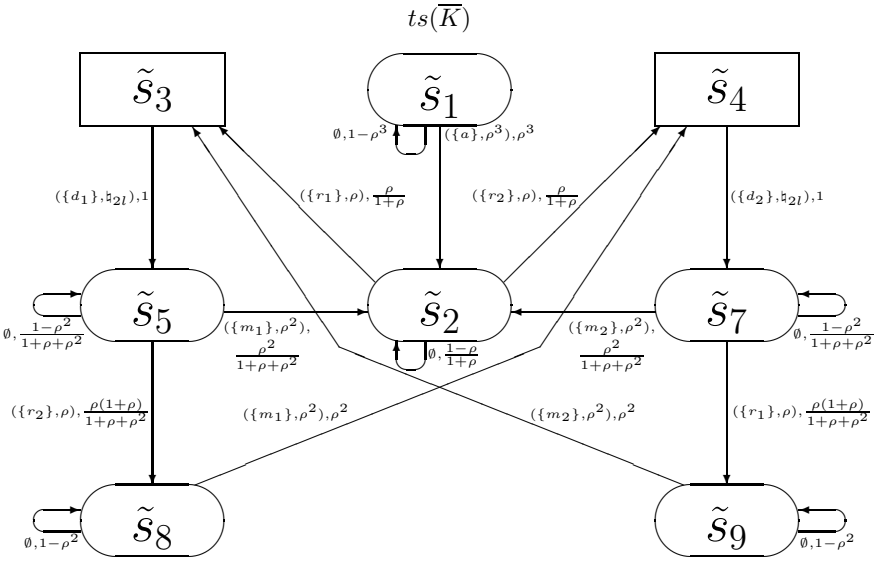

Figure 48: The interleaving transition system of the generalized shared memory system.

$\tilde{\phi}'_2$, $\tilde{\phi}'_4$, $\tilde{\phi}'_6$ as functions of ρ are depicted. One cane see substantial differences between plots in Figure 52 and those in Figure 45. The differences indicate that when ρ tends to 1*, the increase of performance (treated as the time fraction when the memory is allocated to a processor and not required by another one) is much quicker in step semantics (the function $\tilde{\varphi}'_4$) than in the interleaving semantics ($\tilde{\phi}'_4$). Moreover, when $\rho = 1$, the memory is allocated and not requested ($\tilde{\varphi}'_4$) with probability* 1 *in the step case while all the three variants exist with positive probabilities in the interleaving case: the memory is not allocated and not requested ($\tilde{\phi}'_2$) with probability $\frac{1}{6}$ or the memory is allocated and not requested ($\tilde{\phi}'_4$) with probability $\frac{1}{2}$ or the memory is allocated and requested ($\tilde{\phi}'_6$) with probability $\frac{1}{3}$. On the other hand, the dependence of the steady-state PMF on ρ is more complex and interesting in step semantics than in interleaving one, since the functions $\tilde{\varphi}'_2$, $\tilde{\varphi}'_4$, $\tilde{\varphi}'_6$ have local extremes and more inflections than $\tilde{\phi}'_2$, $\tilde{\phi}'_4$, $\tilde{\phi}'_6$.*

Example 11.2. *Let us take $\rho = \frac{1}{2}$ and $l = 1$ in the interleaving transition systems and Markov chains from Example 11.1. Then we obtain the interleaving transition system $ts(\overline{E})$, quotient transition system $ts_{\underline{\leftrightarrow}_{is}}(\overline{F})$ and reduced quotient DTMC $rdtmc_{\underline{\leftrightarrow}_{is}}(\overline{F})$ of the concrete and abstract* standard *shared memory system, specified in Section 9 by the static expressions E and F, respectively. In Table 14, the transient and the steady-state probabilities $\phi'^{\diamond}_i[k]$ $(1 \leq i \leq 4)$ for the interleaving reduced quotient DTMC of the abstract shared memory system at the time moments $k \in \{0, 5, 10, \ldots, 50\}$ and $k = \infty$ are presented, and in Figure 53, the alteration diagram (evolution in time) for the transient probabilities is depicted. The steady-state PMF for $rdtmc_{\underline{\leftrightarrow}_{is}}(\overline{F})$ is $\phi'^{\diamond} = \left(0, \frac{3}{41}, \frac{14}{41}, \frac{24}{41}\right)$, whereas the steady-state PMF for $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$ is $\psi'^{\diamond} = \left(0, \frac{1}{17}, \frac{6}{17}, \frac{10}{17}\right)$. One can see that with k growing, $\phi'^{\diamond}_4[k] = \phi'^{\diamond}[k](\mathcal{K}_6)$ stabilizes slower (especially for the small values of k) than $\psi'^{\diamond}_4[k] = \psi'^{\diamond}[k](\mathcal{K}_6)$ from Table 12 and Figure 36, since $rdtmc_{\underline{\leftrightarrow}_{is}}(\overline{F})$ has no transition from $\mathcal{K}_2$ to $\mathcal{K}_6$, unlike $RDTMC_{\underline{\leftrightarrow}_{ss}}(\overline{F})$. For instance, the absolute relative differences for $k = 5$ are $\left|\frac{\phi'^{\diamond}_4 - \phi'^{\diamond}_4[5]}{\phi'^{\diamond}_4}\right| = \left|\frac{0.5854 - 0.1379}{0.5854}\right| = \frac{0.4475}{0.5854} \approx 0.7644$ (about 76%) and $\left|\frac{\psi'^{\diamond}_4 - \psi'^{\diamond}_4[5]}{\psi'^{\diamond}_4}\right| = \left|\frac{0.5882 - 0.1901}{0.5882}\right| = \frac{0.3981}{0.5882} \approx 0.6768$ (about 68%, i.e. 8% less).*

11.3. Application area

From the application viewpoint, one considers what kind of systems are more appropriate to be modeled and analyzed within SPAs. MTIPP and PEPA are well-suited for the interleaving continuous time systems such that the activity rates or the average sojourn time in the states are known in advance and exponential distribution approximates well the activity delay distributions, whereas EMPA can be used to model the mentioned systems with the activity

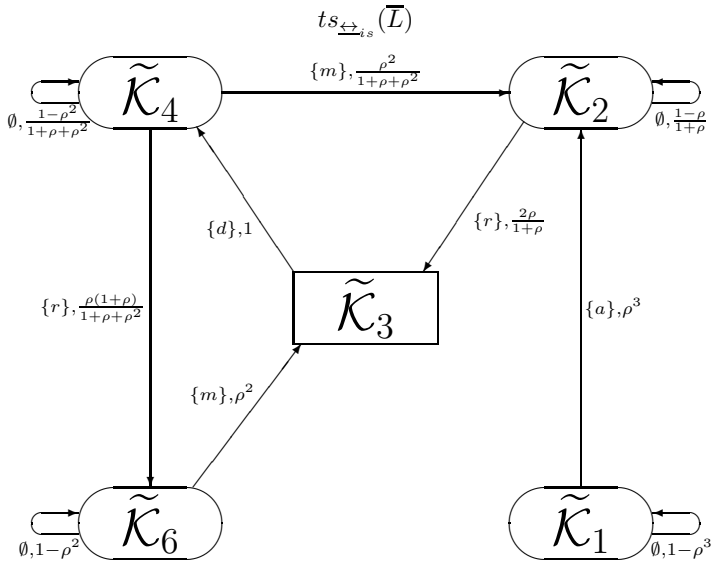

Figure 49: The interleaving quotient transition system of the abstract generalized shared memory system.

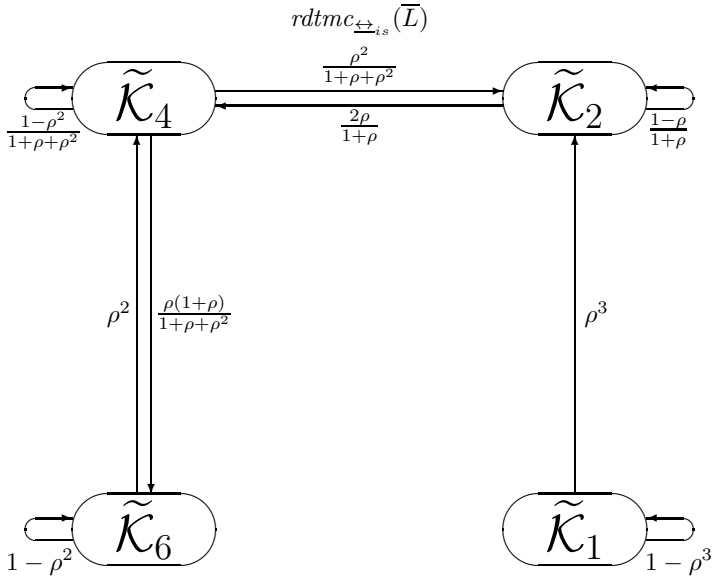

Figure 50: The interleaving reduced quotient DTMC of the abstract generalized shared memory system.

Table 14: Transient and steady-state probabilities for the interleaving reduced quotient DTMC of the abstract shared memory system

k	0	5	10	15	20	25	30	35	40	45	50	∞
$\phi'^{\diamond}_1[k]$	1	0.5129	0.2631	0.1349	0.0692	0.0355	0.0182	0.0093	0.0048	0.0025	0.0013	0
$\phi'^{\diamond}_2[k]$	0	0.1499	0.1155	0.0950	0.0844	0.0789	0.0761	0.0747	0.0739	0.0736	0.0734	0.0732
$\phi'^{\diamond}_3[k]$	0	0.1992	0.2722	0.3061	0.3233	0.3322	0.3367	0.3390	0.3402	0.3408	0.3411	0.3415
$\phi'^{\diamond}_4[k]$	0	0.1379	0.3493	0.4640	0.5231	0.5534	0.5690	0.5770	0.5811	0.5832	0.5842	0.5854

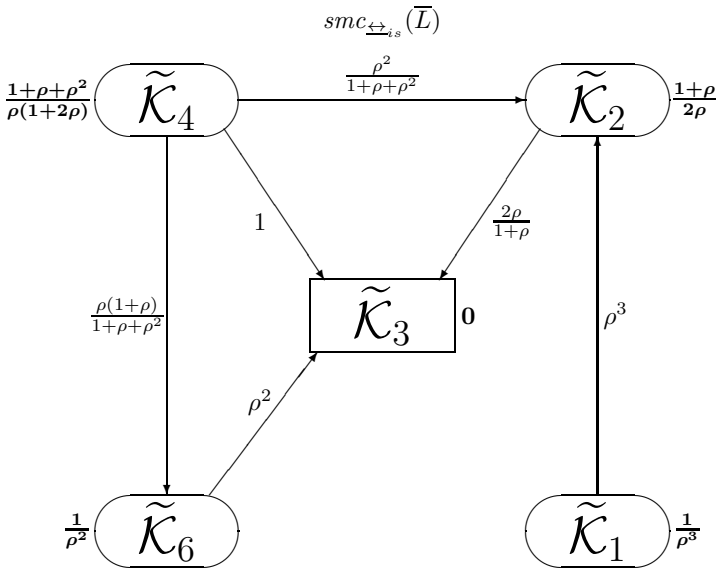

Figure 51: The interleaving quotient underlying SMC of the abstract generalized shared memory system.

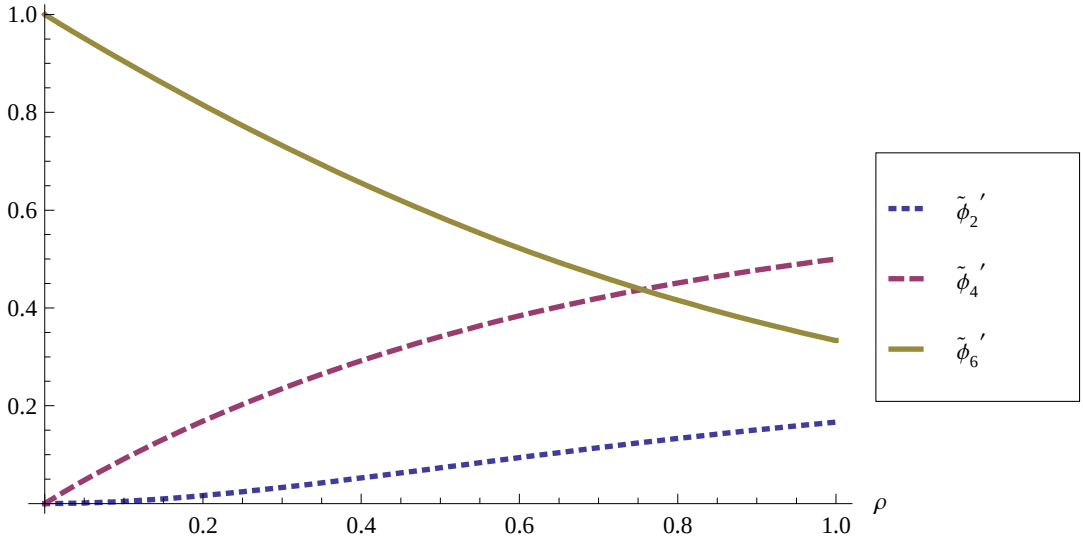

Figure 52: Interleaving steady-state probabilities $\tilde{\phi}'_2$, $\tilde{\phi}'_4$, $\tilde{\phi}'_6$ as functions of the parameter ρ.

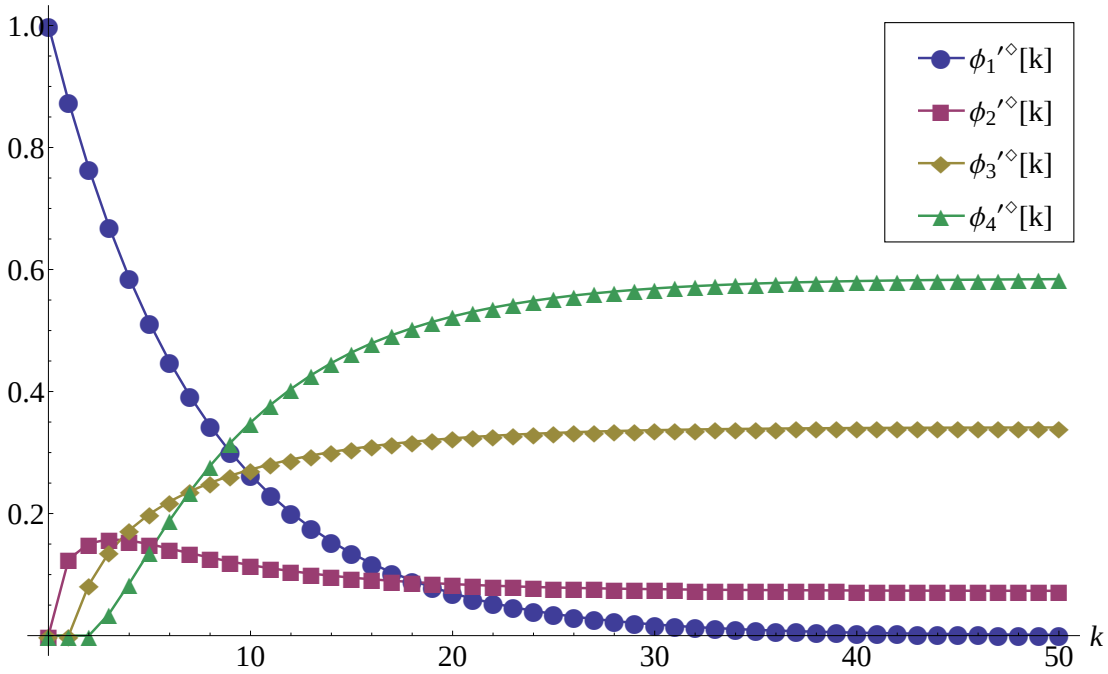

Figure 53: Transient probabilities alteration diagram for the interleaving reduced quotient DTMC of the abstract shared memory system

delays of different duration order or the extended systems, in which purely probabilistic choices or urgent activities must be implemented. GSPA and GSMPA fit well for modeling the continuous time systems with a capability to keep the activity causality information, and with the known activity delay distributions, which cannot be approximated accurately by exponential distribution, while Sπ can additionally model mobility in such systems. TCP^{dst} is a good choice for interleaving discrete time systems with deterministic (fixed) and generalized stochastic delays, whereas sACP is capable to model non-interleaving systems as well, but it offers not enough performance analysis methods. dtsiPBC is consistent for the step discrete time systems such that the independent execution probabilities of activities are known and geometrical distribution approximates well the state residence time distributions. In addition, dtsiPBC can model these systems featuring very scattered activity delays or even more complex systems with instantaneous probabilistic choice or urgency, hence, dtsiPBC can be taken as a non-interleaving discrete time counterpart of EMPA.

11.4. Advantages of our approach

The main advantages of dtsiPBC are the flexible multiaction labels, immediate multiactions, powerful operations, as well as a step operational and a Petri net denotational semantics allowing for concurrent execution of activities (transitions), together with an ability for analytical and parametric performance evaluation. The uniqueness of our approach consists in applying a parallel semantics for the process expressions and preserving the concurrency level in the extracted performance models (SMC, DTMC and RDTMC) through their state changes corresponding to the simultaneous executions.

12. Conclusion

In this paper, we have proposed a discrete time stochastic extension dtsiPBC of a finite part of PBC enriched with iteration and immediate multiactions. In the presented version of dtsiPBC, we have used positive reals as the weights of immediate multiactions, with a goal to enhance its specification capabilities. The calculus has a concurrent step operational semantics based on labeled probabilistic transition systems and a denotational semantics in terms of a subclass of LDTSIPNs. A method of performance evaluation in the framework of the calculus has been presented that explores the underlying stochastic process, which is a semi-Markov chain (SMC). In such an SMC, the sojourn time

in every tangible state is geometrically distributed (with the special case of being infinity) while the sojourn time in every vanishing state is zero. It has been proved that the corresponding discrete time Markov chain (DTMC) or its reduction (RDTMC) by eliminating vanishing states may alternatively and suitably be studied for that purpose. We have also established coincidence of the reduced SMC (RSMC) and RDTMC. We have demonstrated that for DTMCs of the process expressions, an additional embedding (into RSMC) of the reduced embedded DTMC is needed to coincide with the embedded reduced DTMC. Thus, reduction before embedding (constructing the RDTMC instead of the RSMC) requires less computations.

Further, step stochastic bisimulation equivalence of process expressions has been defined and its interrelations with other equivalences of the calculus have been investigated. We have explained how to reduce transition systems and Markov chains (SMCs, DTMCs and RDTMCs) by building their quotients with respect to the introduced equivalence. We have studied an effect of the quotienting to extraction, embedding and reduction, in terms of the transition probability matrices (TPMs) of the quotient DTMCs, EDTMCs and RDTMCs. We have demonstrated that for DTMCs of the process expressions, the quotienting is permutable (commute) with both extraction and reduction, whereas an additional embedding of the quotient embedded DTMC is needed to coincide with the embedded quotient DTMC. Thus, making extraction before the quotienting permits to start reasoning from the Markov chain level. Applying reduction before the quotienting simplifies quantitative analysis in case of many non-equivalent vanishing states. The quotienting before embedding requires less computations. We have proved that the mentioned equivalence guarantees identity of the stationary behaviour and the sojourn time properties, and thus preserves performance measures. A case study of the shared memory system has been presented as an example of modeling, performance evaluation and performance preserving reduction within the calculus. Finally, we have determined the advantages of dtsiPBC by comparing it with other SPAs. In particular, by examining the interleaving transition system of the generalized shared memory system, we have demonstrated that step semantics is preferable to the interleaving one for the specification and analysis, as in our context, as within other discrete time SPAs.

The advantage of our framework is twofold. First, one can specify in it concurrent composition and synchronization of (multi)actions, whereas this is not possible in classical Markov chains. Second, algebraic formulas represent processes in a more compact way than Petri nets and allow one to apply syntactic transformations and comparisons. Process algebras are compositional by definition and their operations naturally correspond to operators of programming languages. Hence, it is much easier to construct a complex model in the algebraic setting than in PNs. The complexity of PNs generated for practical models in the literature demonstrates that it is not straightforward to construct such PNs directly from the system specifications. dtsiPBC is well suited for the discrete time applications, whose discrete states change with a global time tick, such as business processes, neural and transportation networks, computer and communication systems, timed web services [102], as well as for those, in which the distributed architecture or the concurrency level should be preserved while modeling and analysis (remember that, in step semantics, we have additional transitions due to concurrent executions).

Future work will consist in constructing a congruence for dtsiPBC, i.e. the equivalence that withstands application of all operations of the algebra. The first possible candidate is a stronger version of $\underline{\leftrightarrow}_{ss}$ defined via transition systems equipped with two extra transitions skip and redo, like those from [66]. We also plan to extend the calculus with deterministically timed multiactions having a fixed time delay (including the zero one which is the case of immediate multiactions) to enhance expressiveness of the calculus and to extend application area of the associated analysis techniques. The resulting SPA will be a concurrent discrete time analogue of SM-PEPA [19], whose underlying stochastic model is a semi-Markov chain. Moreover, recursion could be added to dtsiPBC to increase further 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 (2000) 237–255. http://tmitwww.tm.tue.nl/staff/hreijers/H.A. Reijers Bestanden/Statistica.pdf.

[2] C. Autant, P. Schnoebelen, Place bisimulations in Petri nets, in: Proc. 13^{th} ICATPN 1992, volume 616 of *Lect. Notes Comp. Sci.*, Springer, 1992, pp. 45–61.

[3] C. Baier, Polynomial time algorithms for testing probabilistic bisimulation and simulation, in: Proc. 8^{th} CAV 1996, volume 1102 of *Lect. Notes Comp. Sci.*, Springer, 1996, pp. 50–61. http://www.inf.tu-dresden.de/content/institutes/thi/algi/publikationen/texte/27_00_old.pdf.

[4] C. Baier, B. Engelen, M. Majster-Cederbaum, Deciding bisimilarity and similarity for probabilistic processes, Journal of Computer and System Sciences 60 (2000) 187–231.

[5] G. Balbo, Introduction to stochastic Petri nets, in: Proc. 1st EEF/Euro Summer School of Trends in Comp. Sci. 2000, volume 2090 of *Lect. Notes Comp. Sci.*, Springer, 2001, pp. 84–155.
[6] G. Balbo, Introduction to generalized stochastic Petri nets, in: Proc. 7th SFM 2007, volume 4486 of *Lect. Notes Comp. Sci.*, Springer, 2007, pp. 83–131.
[7] F. Bause, P.S. Kritzinger, Stochastic Petri nets: an introduction to the theory, Vieweg Verlag, 2002. 2nd edition, 218 pages, http://ls4-www.cs.tu-dortmund.de/cms/de/home/bause/bause_kritzinger_spn_book_print.pdf.
[8] J.A. Bergstra, J.W. Klop, Algebra of communicating processes with abstraction, Theor. Comput. Sci. 37 (1985) 77–121.
[9] M. Bernardo, Theory and application of extended Markovian process algebra, University of Bologna, Italy, 1999. Ph. D. thesis, 276 pages, http://www.sti.uniurb.it/bernardo/documents/phdthesis.pdf.
[10] M. Bernardo, A survey of Markovian behavioral equivalences, in: Proc. 7th SFM 2007, volume 4486 of *Lect. Notes Comp. Sci.*, Springer, 2007, pp. 180–219. http://www.sti.uniurb.it/bernardo/documents/sfm07pe.tuto.pdf.
[11] M. Bernardo, On the tradeoff between compositionality and exactness in weak bisimilarity for integrated-time Markovian process calculi, Theor. Comput. Sci. 563 (2015) 99–143. http://www.sti.uniurb.it/bernardo/documents/tcs563.pdf.
[12] M. Bernardo, M. Bravetti, Reward based congruences: can we aggregate more?, in: Proc. PAPM-PROBMIV 2001, volume 2165 of *Lect. Notes Comp. Sci.*, Springer, 2001, pp. 136–151. http://www.cs.unibo.it/~bravetti/papers/papm01b.ps.
[13] M. Bernardo, L. Donatiello, R. Gorrieri, A formal approach to the integration of performance aspects in the modeling and analysis of concurrent systems, Inform. Comput. 144 (1998) 83–154. http://www.sti.uniurb.it/bernardo/documents/ic144.pdf.
[14] M. Bernardo, R. Gorrieri, A tutorial on EMPA: a theory of concurrent processes with nondeterminism, priorities, probabilities and time, Theor. Comput. Sci. 202 (1998) 1–54. http://www.sti.uniurb.it/bernardo/documents/tcs202.pdf.
[15] E. Best, R. Devillers, J.G. Hall, The box calculus: a new causal algebra with multi-label communication, in: Advances in Petri Nets 1992, volume 609 of *Lect. Notes Comp. Sci.*, Springer, 1992, pp. 21–69.
[16] E. Best, R. Devillers, M. Koutny, Petri net algebra, EATCS Monographs on Theor. Comput. Sci., Springer, 2001. 378 pages.
[17] E. Best, M. Koutny, A refined view of the box algebra, in: Proc. 16th ICATPN 1995, volume 935 of *Lect. Notes Comp. Sci.*, Springer, 1995, pp. 1–20. http://parsys.informatik.uni-oldenburg.de/~best/publications/pn95.ps.gz.
[18] L. Bortolussi, Stochastic concurrent constraint programming, Electronic Notes in Theor. Comp. Sci. 164 (2006) 65–80.
[19] J.T. Bradley, Semi-Markov PEPA: modelling with generally distributed actions, International Journal of Simulation 6 (2005) 43–51. http://pubs.doc.ic.ac.uk/semi-markov-pepa/semi-markov-pepa.pdf.
[20] M. Bravetti, Specification and analysis of stochastic real-time systems, University of Bologna, Italy, 2002. Ph. D. thesis, 432 pages, http://www.cs.unibo.it/~bravetti/papers/phdthesis.ps.gz.
[21] M. Bravetti, M. Bernardo, R. Gorrieri, Towards performance evaluation with general distributions in process algebras, in: Proc. 9th CONCUR 1998, volume 1466 of *Lect. Notes Comp. Sci.*, Springer, 1998, pp. 405–422. http://www.cs.unibo.it/~bravetti/papers/concur98.ps.
[22] M. Bravetti, P.R. D'Argenio, Tutte le algebre insieme: concepts, discussions and relations of stochastic process algebras with general distributions, in: Validation of Stochastic Systems: A Guide to Current Research, volume 2925 of *Lect. Notes Comp. Sci.*, Springer, 2004, pp. 44–88. http://www.cs.unibo.it/~bravetti/papers/voss03.ps.
[23] E. Brinksma, H. Hermanns, Process algebra and Markov chains, in: Proc. 1st EEF/Euro Summer School of Trends in Comp. Sci. 2000, volume 2090 of *Lect. Notes Comp. Sci.*, Springer, 2001, pp. 183–231.
[24] E. Brinksma, J.P. Katoen, R. Langerak, D. Latella, A stochastic causality-based process algebra, Comp. J. 38 (1995) 552–565. http://eprints.eemcs.utwente.nl/6387/01/552.pdf.
[25] G. Bucci, L. Sassoli, E. Vicario, Correctness verification and performance analysis of real-time systems using stochastic preemptive time Petri nets, IEEE Transactions on Software Engineering 31 (2005) 913–927. http://www.dsi.unifi.it/~vicario/Research/TSE05.pdf.
[26] P. Buchholz, Exact and ordinary lumpability in finite Markov chains, Journal of Applied Probability 31 (1994) 59–75.
[27] P. Buchholz, Markovian process algebra: composition and equivalence, in: Proc. 2nd Int. Workshop on Process Algebras and Performance Modelling (PAPM) 1994, number 27 in Arbeitsberichte des IMMD, University of Erlangen, Germany, 1994, pp. 11–30.
[28] P. Buchholz, A notion of equivalence for stochastic Petri nets, in: Proc. 16th ICATPN 1995, volume 935 of *Lect. Notes Comp. Sci.*, Springer, 1995, pp. 161–180.
[29] P. Buchholz, Iterative decomposition and aggregation of labeled GSPNs, in: Proc. 19th ICATPN 1998, volume 1420 of *Lect. Notes Comp. Sci.*, Springer, 1998, pp. 226–245.
[30] P. Buchholz, I.V. Tarasyuk, Net and algebraic approaches to probabilistic modeling, Joint Novosibirsk Computing Center and Institute of Informatics Systems Bulletin, Series Computer Science 15 (2001) 31–64. Novosibirsk, Russia, http://itar.iis.nsk.su/files/itar/pages/spnpancc.pdf.
[31] S. Cattani, R. Segala, Decision algorithms for probabilistic bisimulation, in: Proc. 13th CONCUR 2002, volume 2421 of *Lect. Notes Comp. Sci.*, Springer, 2002, pp. 371–385. http://www.cs.bham.ac.uk/~dxp/papers/CS02.pdf.
[32] I. Christoff, Testing equivalence and fully abstract models of probabilistic processes, in: Proc. 1st CONCUR 1990, volume 458 of *Lect. Notes Comp. Sci.*, Springer, 1990, pp. 126–140.
[33] G. Ciardo, J.K. Muppala, K.S. Trivedi, On the solution of GSPN reward models, Performance Evaluation 12 (1991) 237–253. http://people.ee.duke.edu/~kst/spn_papers/gspnrew.ps.
[34] N. Coste, H. Hermanns, E. Lantreibecq, W. Serwe, Towards performance prediction of compositional models in industrial GALS designs, in: Proc. 21st CAV 2009, volume 5643 of *Lect. Notes Comp. Sci.*, Springer, 2009, pp. 204–218.
[35] C. Daws, Symbolic and parametric model checking of discrete-time Markov chains, in: Proc. 1st ICTAC 2004, volume 3407 of *Lect. Notes Comp. Sci.*, Springer, 2005, pp. 280–294.
[36] P. Degano, C. Priami, Non-interleaving semantics for mobile processes, Theoretical Computer Science 216 (1999) 237–270.
[37] S. Derisavi, H. Hermanns, W.H. Sanders, Optimal state-space lumping of Markov chains, Information Processing Letters 87 (2003) 309–315.
[38] C. Eisentraut, H. Hermanns, J. Schuster, A. Turrini, L. Zhang, The quest for minimal quotients for probabilistic automata, in: Proc. 19th

TACAS 2013, volume 7795 of *Lect. Notes Comp. Sci.*, Springer, 2013, pp. 16–31.
[39] J.M. Fourneau, Collaboration of discrete-time Markov chains: Tensor and product form, Performance Evaluation 67 (2010) 779–796.
[40] S. Gilmore, J. Hillston, L. Kloul, M. Ribaudo, PEPA nets: a structured performance modelling formalism, Performance Evaluation 54 (2003) 79–104. http://www.dcs.ed.ac.uk/pepa/pepanetsJournal.pdf.
[41] R.J. van Glabbeek, The linear time – branching time spectrum II: the semantics of sequential systems with silent moves. Extended abstract, in: Proc. 4^{th} CONCUR 1993, volume 715 of *Lect. Notes Comp. Sci.*, Springer, 1993, pp. 66–81.
[42] R.J. van Glabbeek, S.A. Smolka, B. Steffen, Reactive, generative, and stratified models of probabilistic processes, Inform. Comput. 121 (1995) 59–80. http://boole.stanford.edu/pub/prob.ps.gz.
[43] N. Götz, U. Herzog, M. Rettelbach, Multiprocessor and distributed system design: the integration of functional specification and performance analysis using stochastic process algebras, in: Proc. 16^{th} Performance 1993, volume 729 of *Lect. Notes Comp. Sci.*, Springer, 1993, pp. 121–146.
[44] M.C. Guenther, N.J. Dingle, J.T. Bradley, W.J. Knottenbelt, Passage-time computation and aggregation strategies for large semi-Markov processes, Performance Evaluation 68 (2011) 221–236.
[45] T. Han, J.P. Katoen, A. Mereacre, Approximate parameter synthesis for probabilistic time-bounded reachability, in: Proc. 29^{th} IEEE Real-Time Systems Symposium (RTSS) 2008, IEEE Computer Society Press, New York, USA, 2008, pp. 173–182.
[46] V. Hashemi, H. Hermanns, L. Song, Reward-bounded reachability probability for uncertain weighted MDPs, in: Proc. 17^{th} VMCAI 2016, volume 9583 of *Lect. Notes Comp. Sci.*, Springer, 2016, pp. 351–371.
[47] H. Hatefi, H. Hermanns, Improving time bounded reachability computations in interactive Markov chains, in: Proc. 5^{th} FSEN 2013, volume 8161 of *Lect. Notes Comp. Sci.*, Springer, 2013, pp. 250–266.
[48] B.R. Haverkort, Markovian models for performance and dependability evaluation, in: Proc. 1^{st} EEF/Euro Summer School of Trends in Comp. Sci. 2000, volume 2090 of *Lect. Notes Comp. Sci.*, Springer, 2001, pp. 38–83. http://www-i2.informatik.rwth-aachen.de/Teaching/Seminar/VOSS2005/have01.pdf.
[49] R.A. Hayden, J.T. Bradley, A. Clark, Performance specification and evaluation with unified stochastic probes and fluid analysis, IEEE Transactions on Software Engineering 39 (2013) 97–118. http://pubs.doc.ic.ac.uk/fluid-unified-stochastic-probes/fluid-unified-stochastic-probes.pdf.
[50] H. Hermanns, M. Rettelbach, Syntax, semantics, equivalences and axioms for MTIPP, in: Proc. 2^{nd} Int. Workshop on Process Algebras and Performance Modelling (PAPM) 1994, number 27 in Arbeitsberichte des IMMD, University of Erlangen, Germany, 1994, pp. 71–88.
[51] J. Hillston, The nature of synchronisation, in: Proc. 2^{nd} Int. Workshop on Process Algebras and Performance Modelling (PAPM) 1994, number 27 in Arbeitsberichte des IMMD, University of Erlangen, Germany, 1994, pp. 51–70. http://www.dcs.ed.ac.uk/pepa/synchronisation.pdf.
[52] J. Hillston, A compositional approach to performance modelling, Cambridge University Press, UK, 1996. 158 pages, http://www.dcs.ed.ac.uk/pepa/book.pdf.
[53] C.A.R. Hoare, Communicating sequential processes, Prentice-Hall, London, UK, 1985. http://www.usingcsp.com/cspbook.pdf.
[54] A. Horváth, M. Paolieri, L. Ridi, E. Vicario, Transient analysis of non-Markovian models using stochastic state classes, Performance Evaluation 69 (2012) 315–335.
[55] L. Jategaonkar, A.R. Meyer, Deciding true concurrency equivalences on safe, finite nets, Theor. Comput. Sci. 154 (1996) 107–143.
[56] C.C. Jou, S.A. Smolka, Equivalences, congruences and complete axiomatizations for probabilistic processes, in: Proc. 1^{st} CONCUR 1990, volume 458 of *Lect. Notes Comp. Sci.*, Springer, 1990, pp. 367–383.
[57] G. Kahn, Natural semantics, in: Proc. 4^{th} STACS 1987, Springer, London, UK, 1987, pp. 22–39.
[58] J.P. Katoen, Quantinative and qualitative extensions of event structures, CTIT Ph. D.-thesis series 96-09, Centre for Telematics and Information Technology, University of Twente, Enschede, The Netherlands, 1996. Ph. D. thesis, 303 pages.
[59] J.P. Katoen, E. Brinksma, D. Latella, R. Langerak, Stochastic simulation of event structures, in: Proc. 4^{th} Int. Workshop on Process Algebra and Performance Modelling (PAPM) 1996, CLUT Press, Torino, Italy, 1996, pp. 21–40. http://eprints.eemcs.utwente.nl/6487/01/263_KLLB96b.pdf.
[60] J.P. Katoen, P.R. D'Argenio, General distributions in process algebra, in: Proc. 1^{st} EEF/Euro Summer School of Trends in Comp. Sci. 2000, volume 2090 of *Lect. Notes Comp. Sci.*, Springer, 2001, pp. 375–429.
[61] M. Koutny, A compositional model of time Petri nets, in: Proc. 21^{st} ICATPN 2000, volume 1825 of *Lect. Notes Comp. Sci.*, Springer, 2000, pp. 303–322.
[62] V.G. Kulkarni, Modeling and analysis of stochastic systems, Texts in Statistical Science, Chapman and Hall / CRC Press, 2010.
[63] R. Lanotte, A. Maggiolo-Schettini, A. Troina, Parametric probabilistic transition systems for system design and analysis, Formal Asp. Comput. 19 (2007) 93–109.
[64] K.G. Larsen, A. Skou, Bisimulation through probabilistic testing, Inform. Comput. 94 (1991) 1–28.
[65] H. Macià, V. Valero, D. Cazorla, F. Cuartero, Introducing the iteration in sPBC, in: Proc. 24^{th} FORTE 2004, volume 3235 of *Lect. Notes Comp. Sci.*, Springer, 2004, pp. 292–308. http://www.info-ab.uclm.es/retics/publications/2004/forte04.pdf.
[66] H. Macià, V. Valero, F. Cuartero, D. de-Frutos, A congruence relation for sPBC, Formal Methods in System Design 32 (2008) 85–128. Springer, The Netherlands.
[67] H. Macià, V. Valero, F. Cuartero, M.C. Ruiz, sPBC: a Markovian extension of Petri box calculus with immediate multiactions, Fundamenta Informaticae 87 (2008) 367–406. IOS Press, Amsterdam, The Netherlands.
[68] H. Macià, V. Valero, D. de-Frutos, sPBC: a Markovian extension of finite Petri box calculus, in: Proc. 9^{th} IEEE Int. Workshop on Petri Nets and Performance Models (PNPM) 2001, IEEE Computer Society Press, Aachen, Germany, 2001, pp. 207–216. http://www.info-ab.uclm.es/retics/publications/2001/pnpm01.ps.
[69] J. Markovski, P.R. D'Argenio, J.C.M. Baeten, E.P. de Vink, Reconciling real and stochastic time: the need for probabilistic refinement, Formal Asp. Comput. 24 (2012) 497–518.
[70] J. Markovski, A. Sokolova, N. Trčka, E.P. de Vink, Compositionality for Markov reward chains with fast and silent transitions, Performance Evaluation 66 (2009) 435–452.

[71] J. Markovski, E.P. de Vink, Extending timed process algebra with discrete stochastic time, in: Proc. 12^{th} AMAST 2008, volume 5140 of *Lect. Notes Comp. Sci.*, Springer, 2008, pp. 268–283.
[72] J. Markovski, E.P. de Vink, Performance evaluation of distributed systems based on a discrete real- and stochastic-time process algebra, Fundamenta Informaticae 95 (2009) 157–186. IOS Press, Amsterdam, The Netherlands.
[73] O. Marroquín, D. de-Frutos, Extending the Petri box calculus with time, in: Proc. 22^{nd} ICATPN 2001, volume 2075 of *Lect. Notes Comp. Sci.*, Springer, 2001, pp. 303–322.
[74] M.A. Marsan, Stochastic Petri nets: an elementary introduction, in: Advances in Petri Nets 1989, volume 424 of *Lect. Notes Comp. Sci.*, Springer, 1990, pp. 1–29.
[75] M.A. Marsan, G. Balbo, G. Conte, S. Donatelli, G. Franceschinis, Modelling with generalized stochastic Petri nets, Wiley Series in Parallel Computing, John Wiley and Sons, 1995. 316 pages, http://www.di.unito.it/~greatspn/GSPN-Wiley.
[76] P.M. Merlin, D.J. Farber, Recoverability of communication protocols: implications of a theoretical study, IEEE Transactions on Communications 24 (1976) 1036–1043.
[77] R.A.J. Milner, Communication and concurrency, Prentice-Hall, Upper Saddle River, NJ, USA, 1989. 260 pages.
[78] R.A.J. Milner, J. Parrow, D. Walker, A calculus of mobile processes, I, Inform. Comput. 100 (1992) 1–40.
[79] R.A.J. Milner, J. Parrow, D. Walker, A calculus of mobile processes, II, Inform. Comput. 100 (1992) 41–77.
[80] M.K. Molloy, On the integration of the throughput and delay measures in distributed processing models, Report CSD-810-921, University of California, Los Angeles, CA, USA, 1981. Ph. D. thesis.
[81] M.K. Molloy, Discrete time stochastic Petri nets, IEEE Transactions on Software Engineering 11 (1985) 417–423.
[82] U. Montanari, M. Pistore, D. Yankelevich, Efficient minimization up to location equivalence, in: Proc. 6^{th} ESOP 1996, volume 1058 of *Lect. Notes Comp. Sci.*, Springer, 1996, pp. 265–279.
[83] T.N. Mudge, H.B. Al-Sadoun, A semi-Markov model for the performance of multiple-bus systems, IEEE Transactions on Computers C-34 (1985) 934–942. http://www.eecs.umich.edu/~tnm/papers/SemiMarkov.pdf.
[84] A. Niaouris, An algebra of Petri nets with arc-based time restrictions, in: Proc. 1^{st} ICTAC 2004, volume 3407 of *Lect. Notes Comp. Sci.*, Springer, 2005, pp. 447–462.
[85] R. Paige, R.E. Tarjan, Three partition refinement algorithms, SIAM J. Comput. 16 (1987) 973–989.
[86] G.D. Plotkin, A structural approach to operational semantics, Technical Report DAIMI FN-19, Computer Science Department, Aarhus University, Aarhus, Denmark, 1981.
[87] C. Priami, Stochastic π-calculus with general distributions, in: Proc. 4^{th} Int. Workshop on Process Algebra and Performance Modelling (PAPM) 1996, CLUT Press, Torino, Italy, 1996, pp. 41–57.
[88] C. Priami, Language-based performance prediction for distributed and mobile systems, Inform. Comput. 175 (2002) 119–145.
[89] C. Ramchandani, Perfomance evaluation of asynchronous concurrent systems by timed Petri nets, Massachusetts Institute of Technology, Cambridge, USA, 1973. Ph. D. thesis.
[90] M. Rettelbach, Probabilistic branching in Markovian process algebras, The Computer Journal 38 (1995) 590–599.
[91] S.M. Ross, Stochastic processes, John Wiley and Sons, New York, USA, 1996. 528 pages, 2^{nd} edition.
[92] I.V. Tarasyuk, Discrete time stochastic Petri box calculus, Berichte aus dem Department für Informatik 3/05, Carl von Ossietzky Universität Oldenburg, Germany, 2005. 25 pages, http://itar.iis.nsk.su/files/itar/pages/dtspbcib_cov.pdf.
[93] 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. NCC Publisher, Novosibirsk, Russia, http://itar.iis.nsk.su/files/itar/pages/dtsitncc.pdf.
[94] I.V. Tarasyuk, Stochastic Petri box calculus with discrete time, Fundamenta Informaticae 76 (2007) 189–218. IOS Press, Amsterdam, The Netherlands, http://itar.iis.nsk.su/files/itar/pages/dtspbcfi.pdf.
[95] I.V. Tarasyuk, Equivalence relations for modular performance evaluation in dtsPBC, Math. Struct. Comp. Sci. 24 (2014) 78–154 (e240103). Cambridge University Press, Cambridge, UK, http://itar.iis.nsk.su/files/itar/pages/dtsdphms.pdf.
[96] 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. 25 pages, http://itar.iis.nsk.su/files/itar/pages/dtsipbc.pdf.
[97] I.V. Tarasyuk, H. Macià, V. Valero, Discrete time stochastic Petri box calculus with immediate multiactions dtsiPBC, in: Proc. 6^{th} Int. Workshop on Practical Applications of Stochastic Modelling (PASM) 2012, London, UK, volume 296 of *Electronic Notes in Theor. Comp. Sci.*, Elsevier, 2013, pp. 229–252. http://itar.iis.nsk.su/files/itar/pages/dtsipbcentcs.pdf.
[98] I.V. Tarasyuk, H. Macià, V. Valero, Performance analysis of concurrent systems in algebra dtsiPBC, Programming and Computer Software 40 (2014) 229–249. Pleiades Publishing, Ltd.
[99] I.V. Tarasyuk, H. Macià, V. Valero, Stochastic process reduction for performance evaluation in dtsiPBC, Siberian Electronic Mathematical Reports 12 (2015) 513–551. Sobolev Institute of Mathematics, Novosibirsk, Russia, http://itar.iis.nsk.su/files/itar/pages/dtsipbcsemr.pdf.
[100] C. Tofts, Processes with probabilities, priority and time, Formal Asp. Comput. 6 (1994) 536–564.
[101] C. Tofts, Symbolic approaches to probability distributions in process algebra, Formal Asp. Comput. 12 (2000) 392–415.
[102] V. Valero, M.E. Cambronero, Using unified modelling language to model the publish/subscribe paradigm in the context of timed web services with distributed resources, Mathematical and Computer Modelling of Dynamical Systems 23 (2017) 570–594.
[103] R. Wimmer, S. Derisavi, H. Hermanns, Symbolic partition refinement with automatic balancing of time and space, Performance Evaluation 67 (2010) 816–836.
[104] R. Zijal, G. Ciardo, G. Hommel, Discrete deterministic and stochastic Petri nets, in: Proc. 9^{th} ITG/GI Professional Meeting "Messung, Modellierung und Bewertung von Rechen- und Kommunikationssystemen" (MMB) 1997, Freiberg, Germany, VDE-Verlag, Berlin, Germany, 1997, pp. 103–117. http://www.cs.ucr.edu/~ciardo/pubs/1997MMB-DDSPN.pdf.
[105] A. Zimmermann, J. Freiheit, G. Hommel, Discrete time stochastic Petri nets for modeling and evaluation of real-time systems, in: Proc. 9^{th} Int. Workshop on Parallel and Distributed Real Time Systems (WPDRTS) 2001, San Francisco, USA, pp. 282–286. http://pdv.cs.tu-berlin.de/~azi/texte/WPDRTS01.pdf.

Appendix A. Proofs

Appendix A.1. Proof of Theorem 4.2

At some points, the present proof for dtsiPBC goes along the lines from the respective proofs for PBC [17, 16] and sPBC [66].

Let $N = Box_{dtsi}(E)$. We define a mapping $\beta : DR(\overline{E}) \to RS(\overline{N})$ so that $\beta([G]_\approx) = M_G$ iff $[G]_\approx \in DR(\overline{E})$ and $(N, M_G) = Box_{dtsi}(G)$. Then, like in PBC [17, 16], one can see that β is a bijection, since for each dynamic expression G, its structural equivalence class $[G]_\approx$ defines a single corresponding marking M_G in the dtsi-box $Box_{dtsi}(G)$ and vice versa.

Clearly, $[\overline{E}]_\approx \in DR(\overline{E})$ and $Box_{dtsi}(\overline{E}) = \overline{Box_{dtsi}(E)} = \overline{N} = (N, {}^\circ N) = (N, M_{\overline{E}})$. Hence, $\beta([\overline{E}]_\approx) = M_{\overline{E}}$. Thus, β binds the initial states of the transition system $TS(\overline{E})$ and the corresponding reachability graph $RG(\overline{N})$.

Let $[G]_\approx \in DR(\overline{E})$ and $\beta([G]_\approx) = M_G \in RS(\overline{N})$. We now prove by induction on the structure of dynamic expressions and corresponding dtsi-boxes that $Exec([G]_\approx)$ and $Fire(M_G)$ are isomorphic. This means that for every $\Upsilon \in Exec([G]_\approx)$ there exists $U \in Fire(M_G)$ such that U consists of the transitions *corresponding* to the activities from Υ and vice versa: $(\alpha, \kappa)_\iota \in \Upsilon \Leftrightarrow t_\iota \in U$, where $\Lambda_N(t_\iota) = \varrho_{(\alpha,\kappa)}$. Thus, the *corresponding* activities and transitions have the same probabilities (in case of stochastic multiactions and transitions) or weights (in case of immediate multiactions and transitions), as well as the same multiaction labels and numberings. We can write $U = U(\Upsilon)$ and $\Upsilon = \Upsilon(U)$, to indicate such a correspondence.

Actually, each Υ and the *corresponding* U are completely defined by the sets of their numberings $Num(\Upsilon) = \{\iota \mid (\alpha, \kappa)_\iota \in \Upsilon\} = \{\iota \mid t_\iota \in U\} = Num(U)$, since each activity and transition have a unique numbering. Moreover, $Exec([G]_\approx)$ and $Fire(M_G)$ are completely defined by the sets of their numberings $Num(Exec([G]_\approx)) = \{Num(\Upsilon) \mid \Upsilon \in Exec([G]_\approx)\} = \{Num(U) \mid U \in Fire(M_G)\} = Num(Fire(M_G))$.

- If $final(G)$ then $G \approx \underline{E}$, $tang([G]_\approx)$ and $Exec([G]_\approx) = Exec([\underline{E}]_\approx) = \{\emptyset\}$. On the other hand, $Box_{dtsi}(G) = Box_{dtsi}(\underline{E}) = \underline{N} = (N, N^\circ) = (N, M_{\underline{E}})$ and $Fire(M_G) = Fire(M_{\underline{E}}) = \{\emptyset\} = Exec([G]_\approx)$.

- If $G = \overline{(\alpha, \rho)_\iota}$ and $\rho \in (0; 1)$ then $tang([G]_\approx)$ and $Exec([G]_\approx) = \{\emptyset, \{(\alpha, \rho)_\iota\}\}$. On the other hand, $Box_{dtsi}(G) = (N_{(\alpha,\rho)_\iota}, {}^\bullet t_\iota)$, where $\Lambda_N(t_\iota) = \varrho_{(\alpha,\rho)}$, and $Fire(M_G) = Fire({}^\bullet t_\iota) = \{\emptyset, \{t_\iota\}\}$, which is isomorphic to $Exec([G]_\approx)$.

- If $G = \overline{(\alpha, \natural_l^0)_\iota}$ and $l \in \boldsymbol{R}_{>0}$ then $vanish([G]_\approx)$ and $Exec([G]_\approx) = \{\{(\alpha, \natural_l^0)_\iota\}\}$. On the other hand, $Box_{dtsi}(G) = (N_{(\alpha,\natural_l^0)_\iota}, {}^\bullet t_\iota)$, where $\Lambda_N(t_\iota) = \varrho_{(\alpha,\natural_l^0)}$, and $Fire(M_G) = Fire({}^\bullet t_\iota) = \{\{t_\iota\}\}$, which is isomorphic to $Exec([G]_\approx)$.

- If $G = H; E$, where $H \in OpRegDynExpr$, $E \in RegStatExpr$, then

$$Exec([H; E]_\approx) = \begin{cases} Exec([H]_\approx), & \neg final(H); \\ Exec([\overline{E}]_\approx) & final(H). \end{cases}$$

  On the other hand, $Box_{dtsi}(G) = Box_{dtsi}(H; E) = (Box_{dtsi}(\lfloor H \rfloor; E), M_{H;E})$, and for $Box_{dtsi}(H) = (Box_{dtsi}(\lfloor H \rfloor), M_H)$, $Box_{dtsi}(\overline{E}) = \overline{N_E} = (N_E, {}^\circ N_E) = (N_E, M_{\overline{E}})$, we have

$$Fire(M_{H;E}) = \begin{cases} Fire(M_H), & M_H \neq N_H^\circ; \\ Fire(M_{\overline{E}}), & M_H = N_H^\circ; \end{cases}$$

  which is isomorphic to $Exec([H; E]_\approx)$.

- If $G = E; H$, where $E \in RegStatExpr$, $H \in OpRegDynExpr$, then

$$Exec([E; H]_\approx) = Exec([H]_\approx).$$

  On the other hand, $Box_{dtsi}(G) = Box_{dtsi}(E; H) = (Box_{dtsi}(E; \lfloor H \rfloor), M_{E;H})$, and for $Box_{dtsi}(H) = (Box_{dtsi}(\lfloor H \rfloor), M_H)$, we have

$$Fire(M_{E;H}) = Fire(M_H);$$

  which is isomorphic to $Exec([E; H]_\approx)$.

- If $G = H[]E$, where $H \in OpRegDynExpr,\ E \in RegStatExpr$, then

$$Exec([H[]E]_{\approx}) = \begin{cases} Exec([H]_{\approx}), & \neg init(H) \vee \\ & (init(H) \wedge vanish([H]_{\approx}) \wedge tang([\overline{E}]_{\approx})); \\ Exec([\overline{E}]_{\approx}), & (init(H) \wedge tang([H]_{\approx}) \wedge vanish([\overline{E}]_{\approx})); \\ Exec([H]_{\approx}) \cup Exec([\overline{E}]_{\approx}), & (init(H) \wedge tang([H]_{\approx}) \wedge tang([\overline{E}]_{\approx})) \vee \\ & (init(H) \wedge vanish([H]_{\approx}) \wedge vanish([\overline{E}]_{\approx})). \end{cases}$$

  On the other hand, $Box_{dtsi}(G) = Box_{dtsi}(H[]E) = (Box_{dtsi}(\lfloor H \rfloor[]E), M_{H[]E})$, and for $Box_{dtsi}(H) = (Box_{dtsi}(\lfloor H \rfloor), M_H),\ Box_{dtsi}(\overline{E}) = \overline{N_E} = (N_E, {}^{\circ}N_E) = (N_E, M_{\overline{E}})$, we have

$$Fire(M_{H[]E}) = \begin{cases} Fire(M_H), & M_H \neq {}^{\circ}N_H \vee \\ & (M_H = {}^{\circ}N_H \wedge vanish(M_H) \wedge tang(M_{\overline{E}})); \\ Fire(M_{\overline{E}}), & (M_H = {}^{\circ}N_H \wedge tang(M_H) \wedge vanish(M_{\overline{E}})); \\ Fire(M_H) \cup Fire(M_{\overline{E}}), & (M_H = {}^{\circ}N_H \wedge tang(M_H) \wedge tang(M_{\overline{E}})) \vee \\ & (M_H = {}^{\circ}N_H \wedge vanish(M_H) \wedge vanish(M_{\overline{E}})); \end{cases}$$

  which is isomorphic to $Exec([H[]E]_{\approx})$.

  If $G = E[]H$, where $E \in RegStatExpr,\ H \in OpRegDynExpr$, then the constructions are similar.

- If $G = H\|Z$, where $H, Z \in OpRegDynExpr$, then

$$Exec([H\|Z]_{\approx}) = \begin{cases} Exec([H]_{\approx}), & (vanish([H]_{\approx}) \wedge tang([Z]_{\approx})); \\ Exec([Z]_{\approx}), & (tang([H]_{\approx}) \wedge vanish([Z]_{\approx})); \\ Exec([H]_{\approx}) \cup Exec([Z]_{\approx}) \cup & \\ (Exec([H]_{\approx}) \odot Exec([Z]_{\approx})), & (tang([H]_{\approx}) \wedge tang([Z]_{\approx})) \vee \\ & (vanish([H]_{\approx}) \wedge vanish([Z]_{\approx})), \end{cases}$$

  where $Exec([H]_{\approx}) \odot Exec([Z]_{\approx}) = \{\Upsilon + \Phi \mid \Upsilon \in Exec([H]_{\approx}),\ \Phi \in Exec([Z]_{\approx})\}$.

  On the other hand, $Box_{dtsi}(G) = Box_{dtsi}(H\|Z) = (Box_{dtsi}(\lfloor H \rfloor\|Z), M_{H\|Z})$, and for $Box_{dtsi}(H) = (Box_{dtsi}(\lfloor H \rfloor), M_H)$, $Box_{dtsi}(Z) = (Box_{dtsi}(\lfloor Z \rfloor), M_Z)$, we have

$$Fire(M_{H\|Z}) = \begin{cases} Fire(M_H), & (vanish(M_H) \wedge tang(M_Z)); \\ Fire(M_Z), & (tang(M_H) \wedge vanish(M_Z)); \\ Fire(M_H) \cup Fire(M_Z) \cup & \\ (Fire(M_H) \odot Fire(M_Z)), & (tang(M_H) \wedge tang(M_Z)) \vee (vanish(M_H) \wedge vanish(M_Z)), \end{cases}$$

  where $Fire(M_H) \odot Fire(M_Z) = \{U \cup T \mid U \in Fire(M_H),\ T \in Fire(M_Z)\}$; which is isomorphic to $Exec([H\|Z]_{\approx})$.

- If $G = H[f]$, where $H \in OpRegDynExpr$, then

$$Exec([H[f]]_{\approx}) = \{f(\Upsilon) \mid \Upsilon \in Exec([H]_{\approx})\}.$$

  On the other hand, $Box_{dtsi}(G) = Box_{dtsi}(H[f]) = (Box_{dtsi}(\lfloor H \rfloor[f]), M_{H[f]})$, and for $Box_{dtsi}(H) = (Box_{dtsi}(\lfloor H \rfloor), M_H)$, we have

$$Fire(M_{H[f]}) = \{f(U) \mid U \in Fire(M_H)\},$$

  where $f(U) = \{t_\iota \in U \mid \Lambda_H(t_\iota) = \varrho_{(\alpha,\kappa)},\ \Lambda_{H[f]}(t_\iota) = \varrho_{(f(\alpha),\kappa)}\}$; which is isomorphic to $Exec([H[f]]_{\approx})$.

- If $G = H$ rs a, where $H \in OpRegDynExpr$, then

$$Exec([H \text{ rs } a]_{\approx}) = \{\Upsilon - \Upsilon_a \mid \Upsilon \in Exec([H]_{\approx})\},$$

where $\Upsilon_a = \{(\alpha, \kappa)_\iota \in \Upsilon \mid (a \in \alpha) \vee (\hat{a} \in \alpha)\},\ a \in Act$.

On the other hand, $Box_{dtsi}(G) = Box_{dtsi}(H \text{ rs } a) = (Box_{dtsi}(\lfloor H \rfloor \text{ rs } a), M_{H \text{ rs } a})$, and for $Box_{dtsi}(H) = (Box_{dtsi}(\lfloor H \rfloor), M_H)$, we have

$$Fire(M_{H \text{ rs } a}) = \{U \setminus U_a \mid U \in Fire(M_H)\},$$

where $U_a = \{t_\iota \in U \mid \Lambda_H(t_\iota) = \varrho_{(\alpha,\kappa)},\ (a \in \alpha) \vee (\hat{a} \in \alpha)\},\ a \in Act$; which is isomorphic to $Exec([H \text{ rs } a]_{\approx})$.

- If $G = H$ sy a, where $H \in OpRegDynExpr$, then

$$Exec([H \text{ sy } a]_{\approx}) = \begin{cases} Exec([H]_{\approx}) \cup \{\Upsilon + \{(\alpha \oplus_a \beta, \rho \cdot \chi)_{(\iota_1)(\iota_2)}\} \mid \\ \Upsilon + \{(\alpha, \rho)_{\iota_1}\} + \{(\beta, \chi)_{\iota_2}\} \in Exec([H]_{\approx}),\ a \in \alpha,\ \hat{a} \in \beta\}, & tang([H]_{\approx}); \\ Exec([H]_{\approx}) \cup \{\Upsilon + \{(\alpha \oplus_a \beta, \natural^0_{l+m})_{(\iota_1)(\iota_2)}\} \mid \\ \Upsilon + \{(\alpha, \natural^0_l)_{\iota_1}\} + \{(\beta, \natural^0_m)_{\iota_2}\} \in Exec([H]_{\approx}),\ a \in \alpha,\ \hat{a} \in \beta\}, & vanish([H]_{\approx}). \end{cases}$$

On the other hand, $Box_{dtsi}(G) = Box_{dtsi}(H \text{ sy } a) = (Box_{dtsi}(\lfloor H \rfloor \text{ sy } a), M_{H \text{ sy } a})$, and for $Box_{dtsi}(H) = (Box_{dtsi}(\lfloor H \rfloor), M_H)$, we have

$$Fire(M_{H \text{ sy } a}) = \begin{cases} Fire(M_H) \cup \{U \cup \{t_{(\iota_1)(\iota_2)}\} \mid \Lambda_{H \text{ sy } a}(t_{(\iota_1)(\iota_2)}) = \varrho_{(\alpha \oplus_a \beta, \rho \cdot \chi)}, \\ U \cup \{v_{\iota_1}, w_{\iota_2}\} \in Fire(M_H),\ \Lambda_H(v_{\iota_1}) = \varrho_{(\alpha,\rho)},\ \Lambda_H(w_{\iota_2}) = \varrho_{(\beta,\chi)}, \\ a \in \alpha,\ \hat{a} \in \beta\}, & tang(M_H); \\ Fire(M_H) \cup \{U \cup \{t_{(\iota_1)(\iota_2)}\} \mid \Lambda_{H \text{ sy } a}(t_{(\iota_1)(\iota_2)}) = \varrho_{(\alpha \oplus_a \beta, \natural^0_{l+m})}, \\ U \cup \{v_{\iota_1}, w_{\iota_2}\} \in Fire(M_H),\ \Lambda_H(v_{\iota_1}) = \varrho_{(\alpha, \natural^0_l)},\ \Lambda_H(w_{\iota_2}) = \varrho_{(\beta, \natural^0_m)}, \\ a \in \alpha,\ \hat{a} \in \beta\}, & vanish(M_H); \end{cases}$$

which is isomorphic to $Exec([H \text{ sy } a]_{\approx})$.

- If $G = [H * E * F]$, where $H \in OpRegDynExpr,\ E, F \in RegStatExpr$, then

$$Exec([[H * E * F]]_{\approx}) = \begin{cases} Exec([H]_{\approx}), & \neg final(H); \\ Exec([\overline{E}]_{\approx}), & (final(H) \wedge vanish([\overline{E}]_{\approx}) \wedge tang([\overline{F}]_{\approx})); \\ Exec([\overline{F}]_{\approx}), & (final(H) \wedge tang([\overline{E}]_{\approx}) \wedge vanish([\overline{F}]_{\approx})); \\ Exec([\overline{E}]_{\approx}) \cup Exec([\overline{F}]_{\approx}), & (final(H) \wedge tang([\overline{E}]_{\approx}) \wedge tang([\overline{F}]_{\approx})) \vee \\ & (final(H) \wedge vanish([\overline{E}]_{\approx}) \wedge vanish([\overline{F}]_{\approx})). \end{cases}$$

On the other hand, $Box_{dtsi}(G) = Box_{dtsi}([H * E * F]) = (Box_{dtsi}(\lfloor H \rfloor * E * F), M_{[H*E*F]})$, and for $Box_{dtsi}(H) = (Box_{dtsi}(\lfloor H \rfloor), M_H)$, $Box_{dtsi}(\overline{E}) = \overline{N_E} = (N_E, {}^\circ N_E) = (N_E, M_{\overline{E}})$, $Box_{dtsi}(\overline{F}) = \overline{N_F} = (N_F, {}^\circ N_F) = (N_F, M_{\overline{F}})$, we have

$$Fire(M_{[H*E*F]}) = \begin{cases} Fire(M_H), & M_H \neq N^\circ_H; \\ Fire(M_{\overline{E}}), & (M_H = N^\circ_H \wedge vanish(M_{\overline{E}}) \wedge tang(M_{\overline{F}})); \\ Fire(M_{\overline{F}}), & (M_H = N^\circ_H \wedge tang(M_{\overline{E}}) \wedge vanish(M_{\overline{F}})); \\ Fire(M_{\overline{E}}) \cup Fire(M_{\overline{F}}), & (M_H = N^\circ_H \wedge tang(M_{\overline{E}}) \wedge tang(M_{\overline{F}})) \vee \\ & (M_H = N^\circ_H \wedge vanish(M_{\overline{E}}) \wedge vanish(M_{\overline{F}})); \end{cases}$$

which is isomorphic to $Exec([[H * E * F]]_{\approx})$.

- If $G = [E * H * F]$, where $E, F \in RegStatExpr,\ H \in OpRegDynExpr$, then

$$Exec([[E*H*F]]_\approx)=\begin{cases} Exec([H]_\approx), & (\neg init(H) \wedge \neg final(H)) \vee \\ & ((init(H) \vee final(H)) \wedge vanish([H]_\approx) \wedge tang([\overline{F}]_\approx)); \\ Exec([\overline{F}]_\approx), & ((init(H) \vee final(H)) \wedge tang([H]_\approx) \wedge vanish([\overline{F}]_\approx)); \\ Exec([H]_\approx) \cup Exec([\overline{F}]_\approx), & ((init(H) \vee final(H)) \wedge tang([H]_\approx) \wedge tang([\overline{F}]_\approx)) \vee \\ & ((init(H) \vee final(H)) \wedge vanish([H]_\approx) \wedge vanish([\overline{F}]_\approx)). \end{cases}$$

On the other hand, $Box_{dtsi}(G) = Box_{dtsi}([E * H * F]) = (Box_{dtsi}(E * \lfloor H \rfloor * F), M_{[E*H*F]})$, and for $Box_{dtsi}(H) = (Box_{dtsi}(\lfloor H \rfloor), M_H)$, $Box_{dtsi}(\overline{F}) = \overline{N_F} = (N_F, {}^\circ N_F) = (N_F, M_{\overline{F}})$, we have

$$Fire(M_{[E*H*F]})=\begin{cases} Fire(M_H), & (M_H \neq {}^\circ N_H \wedge M_H \neq N_H^\circ) \vee \\ & ((M_H = {}^\circ N_H \vee M_H = N_H^\circ) \wedge vanish(M_H) \wedge tang(M_{\overline{F}})); \\ Fire(M_{\overline{F}}), & ((M_H = {}^\circ N_H \vee M_H = N_H^\circ) \wedge tang(M_H) \wedge vanish(M_{\overline{F}})); \\ Fire(M_H) \cup Fire(M_{\overline{F}}), & ((M_H = {}^\circ N_H \vee M_H = N_H^\circ) \wedge tang(M_H) \wedge tang(M_{\overline{F}})) \vee \\ & ((M_H = {}^\circ N_H \vee M_H = N_H^\circ) \wedge vanish(M_H) \wedge vanish(M_{\overline{F}})); \end{cases}$$

which is isomorphic to $Exec([[E * H * F]]_\approx)$.

- If $G = [E * F * H]$, where $E, F \in RegStatExpr,\ H \in OpRegDynExpr$, then

$$Exec([[E*F*H]]_\approx) = \begin{cases} Exec([\overline{F}]_\approx), & (vanish([\overline{F}]_\approx) \wedge init(H) \wedge tang([H]_\approx)); \\ Exec([H]_\approx), & \neg init(H) \vee \\ & (tang([\overline{F}]_\approx) \wedge init(H) \wedge vanish([H]_\approx)); \\ Exec([\overline{F}]_\approx) \cup Exec([H]_\approx), & (tang([\overline{F}]_\approx) \wedge init(H) \wedge tang([H]_\approx)) \vee \\ & (vanish([\overline{F}]_\approx) \wedge init(H) \wedge vanish([H]_\approx)). \end{cases}$$

On the other hand, $Box_{dtsi}(G) = Box_{dtsi}([E * F * H]) = (Box_{dtsi}(E * F * \lfloor H \rfloor), M_{[E*F*H]})$, and for $Box_{dtsi}(\overline{F}) = \overline{N_F} = (N_F, {}^\circ N_F) = (N_F, M_{\overline{F}})$, $Box_{dtsi}(H) = (Box_{dtsi}(\lfloor H \rfloor), M_H)$, we have

$$Fire(M_{[E*F*H]}) = \begin{cases} Fire(M_{\overline{F}}), & (vanish(M_{\overline{F}}) \wedge M_H = {}^\circ N_H \wedge tang(M_H)); \\ Fire(M_H), & M_H \neq {}^\circ N_H \vee \\ & (tang(M_{\overline{F}}) \wedge M_H = {}^\circ N_H \wedge vanish(M_H)); \\ Fire(M_{\overline{F}}) \cup Fire(M_H), & (tang(M_{\overline{F}}) \wedge M_H = {}^\circ N_H \wedge tang(M_H)) \vee \\ & (vanish(M_{\overline{F}}) \wedge M_H = {}^\circ N_H \wedge vanish(M_H)); \end{cases}$$

which is isomorphic to $Exec([[E * F * H]]_\approx)$.

Thus, we have proved that $Exec([G]_\approx)$ and $Fire(M_G)$ are isomorphic. It remains to check the homomorphism property, stating that for all $[G]_\approx, [\widetilde{G}]_\approx \in DR(\overline{E})$ and for all *corresponding* $\Upsilon \in Exec([G]_\approx),\ U \in Fire(M_G)$ it holds $[G]_\approx \xrightarrow{\Upsilon}_{\mathcal{P}} [\widetilde{G}]_\approx \ \Leftrightarrow\ M_G = \beta([G]_\approx) \xrightarrow{U}_{\mathcal{P}} \beta([\widetilde{G}]_\approx) = M_{\widetilde{G}}$.

Note that the probability functions $PF(\Upsilon, [G]_\approx)$ and $PT(\Upsilon, [G]_\approx)$ depend only on the structure of $Exec([G]_\approx)$, as well as on the probabilities of stochastic multiactions and weights of immediate multiactions from its elements. Analogously, $PF(U, M_G)$ and $PT(U, M_G)$ depend only on the structure of $Fire(M_G)$, as well as the probabilities of stochastic transitions and weights of immediate transitions from its elements. Further, $PF(\Upsilon, [G]_\approx)$ and $PT(\Upsilon, [G]_\approx)$ are respectively defined in the same way (using the same formulas and cases) as $PF(U, M_G)$ and $PT(U, M_G)$, for each pair of the *corresponding* (multi)set of activities Υ and transition set U. Obviously, the isomorphism of $Exec([G]_\approx)$ and $Fire(M_G)$ guarantees coincidence of their structure as well as the mentioned probabilities and weights. Hence, if U *corresponds* to Υ then $PF(\Upsilon, [G]_\approx) = PF(U, M_G)$ and $PT(\Upsilon, [G]_\approx) = PT(U, M_G)$.

We also have $\mathcal{L}(\Upsilon) = \mathcal{L}(U)$, where $\mathcal{L}(U) = \sum_{\{t \in U | \Lambda_G(t) = \varrho_{(\alpha,\kappa)}\}} \alpha$ is the *multiaction part* of a set of transitions $U \subseteq T_N$. Thus, each transition $[G]_\approx \xrightarrow{\Upsilon}_{\mathcal{P}} \tilde{s}$ in $TS(\overline{E})$ has a corresponding one $M_G \xrightarrow{U}_{\mathcal{P}} \widetilde{M}$ in $RG(\overline{N})$ with $\mathcal{L}(\Upsilon) = \mathcal{L}(U)$ and vice versa. Observe that the structure of the plain and operator dtsi-boxes in dtsiPBC is similar to that of the plain and operator boxes in PBC. Hence, like in PBC [17, 16], we can prove that $\tilde{s} = [\widetilde{G}]_\approx$ and $\widetilde{M} = M_{\widetilde{G}}$ with $(N, M_{\widetilde{G}}) = Box_{dtsi}(\widetilde{G})$ for the dynamic expression $\widetilde{G}$ such that $G \xrightarrow{\Upsilon} \widetilde{G}$. Therefore, by construction of β, we get $\beta([\widetilde{G}]_\approx) = M_{\widetilde{G}}$. □

Appendix A.2. Proof of Theorem 5.2

Let $\mathbf{P}_r$ be the reordered (by moving vanishing states to the first positions) TPM for $DTMC(G)$. Like in Section 5, we reorder the states from $DR(G)$ so that the first rows and columns of $\mathbf{P}_r$ will correspond to the states from $DR_V(G)$ and the last ones will correspond to the states from $DR_T(G)$. Let $|DR(G)| = n$ and $|DR_T(G)| = m$. Then the reordered TPM for $DTMC(G)$ can be decomposed as

$$\mathbf{P}_r = \begin{pmatrix} \mathbf{C} & \mathbf{D} \\ \mathbf{E} & \mathbf{F} \end{pmatrix}.$$

The elements of the $(n-m) \times (n-m)$ submatrix $\mathbf{C}$ are the probabilities to move from vanishing to vanishing states, and those of the $(n-m) \times m$ submatrix $\mathbf{D}$ are the probabilities to move from vanishing to tangible states. The elements of the $m \times (n-m)$ submatrix $\mathbf{E}$ are the probabilities to move from tangible to vanishing states, and those of the $m \times m$ submatrix $\mathbf{F}$ are the probabilities to move from tangible to tangible states.

The TPM $\mathbf{P}^\diamond$ for $RDTMC(G)$ is the $m \times m$ matrix, calculated as

$$\mathbf{P}^\diamond = \mathbf{F} + \mathbf{EGD},$$

where the elements of the matrix $\mathbf{G} = \sum_{k=0}^{\infty} \mathbf{C}^k$ are the probabilities to move from vanishing to vanishing states in any number of state changes, without traversal of tangible states, in $DTMC(G)$. We define the matrix $\mathbf{H} = \mathbf{EGD}$. For $s, \tilde{s} \in DR_T(G)$, let $PM_F(s, \tilde{s})$ and $PM_H(s, \tilde{s})$ be the probabilities to change from s to $\tilde{s}$ for the submatrix $\mathbf{F}$ and matrix $\mathbf{H}$, respectively.

In a similar way, the reordered TPM for $EDTMC(G)$ can be decomposed as

$$\mathbf{P}_r^* = \begin{pmatrix} \mathbf{C}^* & \mathbf{D}^* \\ \mathbf{E}^* & \mathbf{F}^* \end{pmatrix}.$$

The elements of the submatrices of $\mathbf{P}_r^*$ are described like those of the submatrices of $\mathbf{P}_r$.

The TPM $(\mathbf{P}^*)^\diamond$ for $REDTMC(G)$ is the $m \times m$ matrix, calculated as

$$(\mathbf{P}^*)^\diamond = \mathbf{F}^* + \mathbf{E}^*\mathbf{G}'\mathbf{D}^*,$$

where the elements of the matrix $\mathbf{G}' = \sum_{k=0}^{\infty} (\mathbf{C}^*)^k$ are the probabilities to move from vanishing to vanishing states in any number of state changes, without traversal of tangible states, in $EDTMC(G)$. We define the matrix $\mathbf{H}' = \mathbf{E}^*\mathbf{G}'\mathbf{D}^*$. For $s, \tilde{s} \in DR_T(G)$, let $PM_{H'}(s, \tilde{s})$ be the probability to change from s to $\tilde{s}$ for the matrix $\mathbf{H}'$.

From the proof of Theorem 5.1, we have $\mathbf{P}_r^* = Diag(SL_r)(\mathbf{P}_r - \mathbf{I}) + \mathbf{I}$, where SL_r is the reordered (by moving vanishing states to the first positions) self-loops abstraction vector of G in $DTMC(G)$. Let SL_C and SL_F be the self-loops abstraction subvectors of G for the submatrices $\mathbf{C}$ and $\mathbf{F}$, respectively, i.e. the "head" of length $n-m$ and the "tail" of length m, taken from the vector SL_r, with the following elements: $\forall s \in DR_V(G)\ SL_C(s) = SL_r(s)$ and $\forall s \in DR_T(G)\ SL_F(s) = SL_r(s)$. Then we have

$$\mathbf{P}_r^* = \begin{pmatrix} Diag(SL_C) & \mathbf{0} \\ \mathbf{0} & Diag(SL_F) \end{pmatrix} \begin{pmatrix} \mathbf{C} - \mathbf{I} & \mathbf{D} \\ \mathbf{E} & \mathbf{F} - \mathbf{I} \end{pmatrix} + \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} = \begin{pmatrix} Diag(SL_C)(\mathbf{C} - \mathbf{I}) + \mathbf{I} & Diag(SL_C)\mathbf{D} \\ Diag(SL_F)\mathbf{E} & Diag(SL_F)(\mathbf{F} - \mathbf{I}) + \mathbf{I} \end{pmatrix}.$$

Hence, $\mathbf{C}^* = Diag(SL_C)(\mathbf{C} - \mathbf{I}) + \mathbf{I}$, $\mathbf{D}^* = Diag(SL_C)\mathbf{D}$, $\mathbf{E}^* = Diag(SL_F)\mathbf{E}$, $\mathbf{F}^* = Diag(SL_F)(\mathbf{F} - \mathbf{I}) + \mathbf{I}$.

Then $(\mathbf{P}^*)^\diamond = \mathbf{F}^* + \mathbf{E}^*\mathbf{G}'\mathbf{D}^* = Diag(SL_F)(\mathbf{F}-\mathbf{I}) + \mathbf{I} + Diag(SL_F)\mathbf{E}\mathbf{G}'Diag(SL_C)\mathbf{D} = Diag(SL_F)((\mathbf{F} + \mathbf{E}\mathbf{G}'Diag(SL_C)\mathbf{D})-\mathbf{I})+\mathbf{I}$. Let us explore the matrix $\mathbf{G}'Diag(SL_C)$. The matrix $\mathbf{G}'$ can have two different forms, depending on whether the loops among vanishing states exist in $EDTMC(G)$, hence, we consider the two following cases.

1. There exist *no loops among vanishing states* in $EDTMC(G)$. We have $\exists l \in \mathbb{N}\ \forall k > l\ (\mathbf{C}^*)^k = \mathbf{0}$ and $\mathbf{G}' = \sum_{k=0}^{l}(\mathbf{C}^*)^k$.
   Then there are no loops among *different* vanishing states in $DTMC(G)$ (but self-loops may exist in vanishing states), since no loop among *different* states is removed and all self-loops (in the non-absorbing states) are removed in $EDTMC(G)$, with respect to $DTMC(G)$.
   Let there are no self-loops in vanishing states in $DTMC(G)$. In such a case, $\forall s \in DT_V(G)\ SL_C(s) = SL(s) = 1$ and $Diag(SL_C) = \mathbf{I}$. We have $\mathbf{C}^* = Diag(SL_C)(\mathbf{C}-\mathbf{I})+\mathbf{I} = \mathbf{I}(\mathbf{C}-\mathbf{I})+\mathbf{I} = \mathbf{C}$ and $\mathbf{G}' = \sum_{k=0}^{l}(\mathbf{C}^*)^k = \sum_{k=0}^{l}\mathbf{C}^k = \mathbf{G}$. Thus, $\mathbf{G}'Diag(SL_C) = \mathbf{G}\mathbf{I} = \mathbf{G}$.
   Let there are self-loops in vanishing states in $DTMC(G)$. In such a case, $\mathbf{G} = (\mathbf{I}-\mathbf{C})^{-1}$. Note that $\mathbf{C} \neq \mathbf{I} \neq \mathbf{C}^*$, since there exist no absorbing vanishing states in $DTMC(G)$. It is easy to prove by induction on $l \in \mathbb{N}$ that $\mathbf{G}'(\mathbf{I}-\mathbf{C}^*) = \left(\sum_{k=0}^{l}(\mathbf{C}^*)^k\right)(\mathbf{I}-\mathbf{C}^*) = \mathbf{I} - (\mathbf{C}^*)^{l+1}$. Since $(\mathbf{C}^*)^{l+1} = \mathbf{0}$, we get $\mathbf{G}'(\mathbf{I}-\mathbf{C}^*) = \mathbf{I}-\mathbf{0} = \mathbf{I}$. In a similar way, we show that $(\mathbf{I}-\mathbf{C}^*)\mathbf{G}' = \mathbf{I}$. We have $\lim_{k\to\infty}(\mathbf{C}^*)^k = \mathbf{0}$. Hence, $\mathbf{G}' = (\mathbf{I}-\mathbf{C}^*)^{-1} = (\mathbf{I} - Diag(SL_C)(\mathbf{C}-\mathbf{I}) - \mathbf{I})^{-1} = (Diag(SL_C)(\mathbf{I}-\mathbf{C}))^{-1} = (\mathbf{I}-\mathbf{C})^{-1}Diag(SL_C)^{-1} = \mathbf{G}Diag(SL_C)^{-1}$. Thus, $\mathbf{G}'Diag(SL_C) = \mathbf{G}Diag(SL_C)^{-1}Diag(SL_C) = \mathbf{G}$.
2. There exist *loops among vanishing states* in $EDTMC(G)$. We have $\lim_{k\to\infty}(\mathbf{C}^*)^k = \mathbf{0}$ and $\mathbf{G}' = (\mathbf{I}-\mathbf{C}^*)^{-1}$.
   Then there are loops among vanishing states in $DTMC(G)$, since no loop among states is removed and self-loops are possibly added in $DTMC(G)$, with respect to $EDTMC(G)$. Hence, $\lim_{k\to\infty}(\mathbf{C})^k = \mathbf{0}$ and $\mathbf{G} = (\mathbf{I}-\mathbf{C})^{-1}$.
   We have $\mathbf{G}' = (\mathbf{I}-\mathbf{C}^*)^{-1} = (\mathbf{I} - Diag(SL_C)(\mathbf{C}-\mathbf{I}) - \mathbf{I})^{-1} = (Diag(SL_C)(\mathbf{I}-\mathbf{C}))^{-1} =$
   $(\mathbf{I}-\mathbf{C})^{-1}Diag(SL_C)^{-1} = \mathbf{G}Diag(SL_C)^{-1}$. Thus, $\mathbf{G}'Diag(SL_C) = \mathbf{G}Diag(SL_C)^{-1}Diag(SL_C) = \mathbf{G}$.

In the both cases above, we get $\mathbf{G}'Diag(SL_C) = \mathbf{G}$. Hence, $(\mathbf{P}^*)^\diamond = Diag(SL_F)((\mathbf{F} + \mathbf{E}\mathbf{G}'Diag(SL_C)\mathbf{D}) - \mathbf{I}) + \mathbf{I} = Diag(SL_F)((\mathbf{F} + \mathbf{E}\mathbf{G}\mathbf{D}) - \mathbf{I}) + \mathbf{I} = Diag(SL_F)(\mathbf{P}^\diamond - \mathbf{I}) + \mathbf{I}$.

Let $s, \tilde{s} \in DR_T(G)$. The EDTMC for $RDTMC(G)$ is denoted by $ERDTMC(G)$ and has the probabilities $(PM^\diamond)^*(s,\tilde{s})$ to change from s to $\tilde{s}$. The RDTMC for $EDTMC(G)$ is denoted by $REDTMC(G)$ and has the probabilities $(PM^*)^\diamond(s,\tilde{s})$ to change from s to $\tilde{s}$. The EDTMC for $REDTMC(G)$ is denoted by $EREDTMC(G)$ and has the probabilities $((PM^*)^\diamond)^*(s,\tilde{s})$ to change from s to $\tilde{s}$.

Further, let SL_H and $SL_{H'}$ be the self-loops abstraction vectors of G for the matrices $\mathbf{H}$ and $\mathbf{H}'$, respectively. We have $(\mathbf{P}^*)^\diamond = \mathbf{F}^* + \mathbf{H}' = \mathbf{F}^* + Diag(SL_F)\mathbf{E}\mathbf{G}\mathbf{D} = \mathbf{F}^* + Diag(SL_F)\mathbf{H}$. Hence, $\mathbf{H}' = Diag(SL_F)\mathbf{H}$ and $\forall s, \tilde{s} \in DR_T(G)\ PM_{H'}(s,\tilde{s}) = SL_F(s)PM_H(s,\tilde{s})$. Since there are no self-loops in $\mathbf{F}^*$, we conclude that $(SL^*)^\diamond = SL_{H'}$ is the self-loops abstraction vector of G in $REDTMC(G)$.

- Let $PM_F(s,s)+PM_H(s,s) = PM^\diamond(s,s) < 1$ and $PM_F(s,s), PM_H(s,s) > 0$, i.e. s is non-absorbing in $RDTMC(G)$ and there exist self-loops associated with s in $DTMC(G)$ and *extra* self-loops (in addition to those inherited from $DTMC(G)$) in $RDTMC(G)$.

  In $ERDTMC(G)$, we have $(PM^\diamond)^*(s,\tilde{s}) = SL^\diamond(s)PM^\diamond(s,\tilde{s}) = \frac{PM^\diamond(s,\tilde{s})}{1-PM^\diamond(s,s)} = \frac{PM^\diamond(s,\tilde{s})}{1-PM_F(s,s)-PM_H(s,s)} = \frac{\frac{PM^\diamond(s,\tilde{s})}{1-PM_F(s,s)}}{1-\frac{PM_H(s,\tilde{s})}{1-PM_F(s,s)}} = \frac{SL_F(s)PM^\diamond(s,\tilde{s})}{1-SL_F(s)PM_H(s,s)}$. Then the self-loops abstraction factor in s in $RDTMC(G)$ is $SL^\diamond(s) = \frac{SL_F(s)}{1-SL_F(s)PM_H(s,s)} = SL_F(s)SL_{H'}(s)$, where $SL_{H'}(s) = \frac{1}{1-SL_F(s)PM_H(s,s)}$ is the self-loops abstraction factor in s in $REDTMC(G)$. Thus, $(PM^\diamond)^*(s,\tilde{s}) = SL_F(s)SL_{H'}(s)PM^\diamond(s,\tilde{s})$.

  In $EREDTMC(G)$, we have $((PM^*)^\diamond)^*(s,\tilde{s}) = (SL^*)^\diamond(s)(PM^*)^\diamond(s,\tilde{s}) = SL_{H'}(s)(PM^*)^\diamond(s,\tilde{s}) =$
  $SL_{H'}(s)SL_F(s)PM^\diamond(s,\tilde{s}) = (PM^\diamond)^*(s,\tilde{s})$.

  The other three cases (no self-loops associated with s in $DTMC(G)$, no extra self-loops associated with s in $RDTMC(G)$, or no *any* self-loops associated with s in $RDTMC(G)$) are treated analogously, by replacing $PM_F(s,s)$ or/and $PM_H(s,s)$ with zeros.

- Let $PM_F(s,s) + PM_H(s,s) = PM^\diamond(s,s) = 1$ and $PM_F(s,s), PM_H(s,s) > 0$, i.e. s is absorbing in $RDTMC(G)$ and there exist self-loops associated with s in $DTMC(G)$ and *extra* self-loops (in addition to those inherited from $DTMC(G)$) in $RDTMC(G)$.

In $ERDTMC(G)$, we have $(PM^{\circ})^{*}(s,s) = 1$ by definition of the EDTMC, since $PM^{\circ}(s,s) = 1$.

In $REDTMC(G)$, the probability of a self-loop associated with s is $(PM^{*})^{\circ}(s,s) = PM_{H'}(s,s) = SL_F(s)PM_H(s,s) = \frac{PM_H(s,s)}{1-PM_F(s,s)} = \frac{1-PM_F(s,s)}{1-PM_F(s,s)} = 1$.

In $EREDTMC(G)$, we have $((PM^{*})^{\circ})^{*}(s,s) = 1 = (PM^{\circ})^{*}(s,s)$ by definition of the EDTMC, since $(PM^{*})^{\circ}(s,s) = 1$.

The other three cases (no self-loops associated with s in $DTMC(G)$, no extra self-loops associated with s in $RDTMC(G)$, or no *any* self-loops associated with s in $RDTMC(G)$) are treated analogously, by replacing $PM_F(s,s)$ or/and $PM_H(s,s)$ with zeros.

Thus, $((\mathbf{P}^{*})^{\circ})^{*} = (\mathbf{P}^{\circ})^{*}$ and $EREDTMC(G) = ERDTMC(G)$. □

Appendix A.3. Proof of Proposition 6.2

Like it has been done for strong equivalence in Proposition 8.2.1 from [52], we shall prove the following fact about step stochastic bisimulation. Let us have $\forall j \in \mathcal{J},\ \mathcal{R}_j : G \underline{\leftrightarrow}_{ss} G'$ for some index set $\mathcal{J}$. Then the transitive closure of the union of all relations $\mathcal{R} = (\cup_{j\in\mathcal{J}}\mathcal{R}_j)^{+}$ is also an equivalence and $\mathcal{R} : G \underline{\leftrightarrow}_{ss} G'$.

Since $\forall j \in \mathcal{J},\ \mathcal{R}_j$ is an equivalence, by definition of $\mathcal{R}$, we get that $\mathcal{R}$ is also an equivalence.

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

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

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

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

- $n = 1$

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

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

- $n \to n+1$

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

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

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

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

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

Appendix A.4. Proof of Proposition 7.2

Let $\mathcal{K}, \widetilde{\mathcal{K}} \in DR(G)/_{\mathcal{R}_{ss}(G)}$ and $s \in \mathcal{K}$. The EDTMC for the quotient of $EDTMC(G)$ is denoted by $EDTMC'(G)$ and has the probabilities $PM'(\mathcal{K}, \widetilde{\mathcal{K}})$ to change from $\mathcal{K}$ to $\widetilde{\mathcal{K}}$.

- Let $PM(s,s) + PM(s, \mathcal{K} \setminus \{s\}) = PM(s, \mathcal{K}) < 1$ and $PM(s,s), PM(s, \mathcal{K} \setminus \{s\}) > 0$, i.e. $s, \mathcal{K}$ are non-absorbing and there exist self-loops associated with s in $DTMC(G)$ and $\mathcal{K}$ in the quotient of $EDTMC(G)$.

  In $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$, we have $PM^*(\mathcal{K}, \widetilde{\mathcal{K}}) = SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) PM(\mathcal{K}, \widetilde{\mathcal{K}}) = \frac{PM(\mathcal{K}, \widetilde{\mathcal{K}})}{1 - PM(\mathcal{K}, \mathcal{K})} = \frac{PM(s, \widetilde{\mathcal{K}})}{1 - PM(s, \mathcal{K})} = \frac{PM(s, \widetilde{\mathcal{K}})}{1 - PM(s,s) - PM(s, \mathcal{K} \setminus \{s\})} = \frac{\frac{PM(s, \widetilde{\mathcal{K}})}{1 - PM(s,s)}}{1 - \frac{PM(s, \mathcal{K} \setminus \{s\})}{1 - PM(s,s)}} = \frac{SL(s) PM(s, \widetilde{\mathcal{K}})}{1 - SL(s) PM(s, \mathcal{K} \setminus \{s\})}$. Then $SL_{\underline{\leftrightarrow}_{ss}}(\mathcal{K}) = \frac{SL(s)}{1 - SL(s) PM(s, \mathcal{K} \setminus \{s\})} = SL(s) SL'(s, \mathcal{K})$, where $SL'(s, \mathcal{K}) = \frac{1}{1 - SL(s) PM(s, \mathcal{K} \setminus \{s\})}$ is the self-loops abstraction factor in the equivalence class $\mathcal{K}$ with respect to the state $s \in \mathcal{K}$ for the quotient of $EDTMC(G)$.

  In $EDTMC'(G)$, we have $PM'(\mathcal{K}, \widetilde{\mathcal{K}}) = \frac{\sum_{\tilde{s} \in \widetilde{\mathcal{K}}} PM^*(s, \tilde{s})}{1 - \sum_{s' \in \mathcal{K} \setminus \{s\}} PM^*(s, s')} = \frac{\sum_{\tilde{s} \in \widetilde{\mathcal{K}}} SL(s) PM(s, \tilde{s})}{1 - \sum_{s' \in \mathcal{K} \setminus \{s\}} SL(s) PM(s, s')} = \frac{SL(s) \sum_{\tilde{s} \in \widetilde{\mathcal{K}}} PM(s, \tilde{s})}{1 - SL(s) \sum_{s' \in \mathcal{K} \setminus \{s\}} PM(s, s')} = \frac{SL(s) PM(s, \widetilde{\mathcal{K}})}{1 - SL(s) PM(s, \mathcal{K} \setminus \{s\})} = PM^*(\mathcal{K}, \widetilde{\mathcal{K}})$.

  The other three cases (no self-loops associated with s in $DTMC(G)$, with $\mathcal{K}$ in the quotient of $EDTMC(G)$, or with both) are treated analogously, by replacing $PM(s,s)$ or/and $PM(s, \mathcal{K} \setminus \{s\})$ with zeros.

- Let $PM(s,s) + PM(s, \mathcal{K} \setminus \{s\}) = PM(s, \mathcal{K}) = 1$ and $PM(s,s), PM(s, \mathcal{K} \setminus \{s\}) > 0$, i.e. $\mathcal{K}$ is absorbing in $DTMC_{\underline{\leftrightarrow}_{ss}}(G)$ and there exist self-loops associated with s in $DTMC(G)$ and $\mathcal{K}$ in the quotient of $EDTMC(G)$.

  In $EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$, we have $PM^*(\mathcal{K}, \mathcal{K}) = 1$ by definition of the EDTMC, since $PM(\mathcal{K}, \mathcal{K}) = PM(s, \mathcal{K}) = 1$.

  In the quotient of $EDTMC(G)$, the probability of a self-loop associated with $\mathcal{K}$ is $\sum_{s' \in \mathcal{K} \setminus \{s\}} PM^*(s, s') = \sum_{s' \in \mathcal{K} \setminus \{s\}} SL(s) PM(s, s') = SL(s) \sum_{s' \in \mathcal{K} \setminus \{s\}} PM(s, s') = SL(s) PM(s, \mathcal{K} \setminus \{s\}) = SL(s)(1 - PM(s,s)) = \frac{1 - PM(s,s)}{1 - PM(s,s)} = 1$. In $EDTMC'(G)$, we have $PM'(\mathcal{K}, \widetilde{\mathcal{K}}) = 1 = PM^*(\mathcal{K}, \mathcal{K})$ by definition of the EDTMC, since in the quotient of $EDTMC(G)$, the probability of a self-loop associated with $\mathcal{K}$ is 1.

  The other two cases (no self-loops associated with s in $DTMC(G)$ or with $\mathcal{K}$ in the quotient of $EDTMC(G)$) are treated analogously, by replacing $PM(s,s)$ with zero or taking $\mathcal{K} = \{s\}$ when $PM(s, \mathcal{K} \setminus \{s\}) = 0$.

Thus, $(\mathbf{P}^*)^*_{\underline{\leftrightarrow}_{ss}} = \mathbf{P}^*_{\underline{\leftrightarrow}_{ss}}$ and $EDTMC'(G) = EDTMC_{\underline{\leftrightarrow}_{ss}}(G)$. □

Appendix A.5. Proof of Proposition 7.3

Let $\mathbf{P}_r$ be the reordered (by moving vanishing states to the first positions) TPM for $DTMC(G)$. Like in Section 5, we reorder the states from $DR(G)$ so that the first rows and columns of $\mathbf{P}_r$ will correspond to the states from $DR_V(G)$ and the last ones will correspond to the states from $DR_T(G)$. Let $|DR(G)| = n$ and $|DR_T(G)| = m$. Then the reordered TPM for $DTMC(G)$ can be decomposed as

$$\mathbf{P}_r = \begin{pmatrix} \mathbf{C} & \mathbf{D} \\ \mathbf{E} & \mathbf{F} \end{pmatrix}.$$

The elements of the $(n-m) \times (n-m)$ submatrix $\mathbf{C}$ are the probabilities to move from vanishing to vanishing states, and those of the $(n-m) \times m$ submatrix $\mathbf{D}$ are the probabilities to move from vanishing to tangible states. The elements of the $m \times (n-m)$ submatrix $\mathbf{E}$ are the probabilities to move from tangible to vanishing states, and those of the $m \times m$ submatrix $\mathbf{F}$ are the probabilities to move from tangible to tangible states.

The TPM $\mathbf{P}^\diamond$ for $RDTMC(G)$ is the $m \times m$ matrix, calculated as

$$\mathbf{P}^\diamond = \mathbf{F} + \mathbf{EGD},$$

where the elements of the matrix $\mathbf{G} = \sum_{k=0}^{\infty} \mathbf{C}^k$ are the probabilities to move from vanishing to vanishing states in any number of state changes, without traversal of tangible states.

By the note after Proposition 6.1, $\mathcal{R}_{ss}(G) \subseteq (DR_T(G))^2 \uplus (DR_V(G))^2$. Hence, $\forall \mathcal{K} \in DR(G)/_{\mathcal{R}_{ss}(G)}$, all states from $\mathcal{K}$ are tangible, when $\mathcal{K} \in DR_T(G)/_{\mathcal{R}_{ss}(G)}$, or all of them are vanishing, when $\mathcal{K} \in DR_V(G)/_{\mathcal{R}}$.

Let $\mathbf{V}_r$ be the reordered (by moving vanishing states and their equivalence classes to the first positions) collector matrix for $\mathcal{R}_{ss}(\overline{F})$ and $\mathbf{W}_r$ be the (accordingly) reordered distributor matrix for $\mathbf{V}_r$. We reorder the states from $DR(G)$ and the equivalence classes from $DR(G)/_{\mathcal{R}_{ss}(G)}$ as follows. The first rows of $\mathbf{V}_r$ will correspond to the states from $DR_V(G)$ and the first columns of $\mathbf{V}_r$ will correspond to the equivalence classes from $DR_V(G)/_{\mathcal{R}_{ss}(G)}$, whereas the last rows of $\mathbf{V}_r$ will correspond to the states from $DR_T(G)$ and the last columns of $\mathbf{V}_r$ will correspond to the equivalence classes from $DR_T(G)/_{\mathcal{R}_{ss}(G)}$. The first rows of $\mathbf{W}_r$ will correspond to the equivalence classes from $DR_V(G)/_{\mathcal{R}_{ss}(G)}$ and the first columns of $\mathbf{W}_r$ will correspond to the states from $DR_V(G)$, whereas the last rows of $\mathbf{W}_r$ will correspond to the equivalence classes from $DR_T(G)/_{\mathcal{R}_{ss}(G)}$ and the last columns of $\mathbf{W}_r$ will correspond to the states from $DR_T(G)$.

Let $|DR(G)/_{\mathcal{R}_{ss}(G)}|{=}l$ and $|DR_T(G)/_{\mathcal{R}_{ss}(G)}|{=}k$. Note that tangible (vanishing) states can only belong to the equivalence classes of tangible (vanishing) states. Then the reordered collector and distributor matrices can be decomposed as

$$\mathbf{V}_r = \begin{pmatrix} \mathbf{V}_C & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_F \end{pmatrix},\ \mathbf{W}_r = \begin{pmatrix} \mathbf{W}_C & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_F \end{pmatrix},$$

where $\mathbf{0}$ are the matrices consisting only of zeros, all those matrices of the appropriate sizes. The elements of the $(n{-}m){\times}(l{-}k)$ submatrix $\mathbf{V}_C$ are the probabilities to move from vanishing states to the equivalence classes of vanishing states, and those of the $m{\times}k$ submatrix $\mathbf{V}_F$ are the probabilities to move from tangible states to the equivalence classes of tangible states. The elements of the $(l{-}k){\times}(n{-}m)$ submatrix $\mathbf{W}_C$ are the probabilities to move from the equivalence classes of vanishing states to vanishing states, and those of the $k{\times}m$ submatrix $\mathbf{W}_F$ are the probabilities to move from the equivalence classes of tangible states to tangible states.

We have

$$\mathbf{W}_r\mathbf{V}_r = \begin{pmatrix} \mathbf{W}_C\mathbf{V}_C & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_F\mathbf{V}_F \end{pmatrix} = \mathbf{I},$$

hence, $\mathbf{W}_C\mathbf{V}_C = \mathbf{I}$ and $\mathbf{W}_F\mathbf{V}_F = \mathbf{I}$.

Since tangible and vanishing states always belong to the equivalence classes of the same kind, the quotienting (by $\underline{\leftrightarrow}_{ss}$) and reordering (by moving vanishing states and their equivalence classes to the first positions) are permutable. The quotiented reordered TPM may only differ from the reordered quotiented TPM up to the order of the equivalence classes of tangible states and the order of the equivalence classes of vanishing states. To avoid such a difference, we rearrange the equivalence classes of the same kind in increasing order of the smallest indices of the states from them while keeping the equivalence classes of vanishing states at the first positions.

Then $\mathbf{P}_r\mathbf{V}_r = \mathbf{V}_r\mathbf{P}_{r\underline{\leftrightarrow}_{ss}}$ and $\mathbf{P}_{r\underline{\leftrightarrow}_{ss}} = \mathbf{W}_r\mathbf{P}_r\mathbf{V}_r$. We have

$$\mathbf{P}_r\mathbf{V}_r = \begin{pmatrix} \mathbf{C} & \mathbf{D} \\ \mathbf{E} & \mathbf{F} \end{pmatrix}\begin{pmatrix} \mathbf{V}_C & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_F \end{pmatrix} = \begin{pmatrix} \mathbf{C}\mathbf{V}_C & \mathbf{D}\mathbf{V}_F \\ \mathbf{E}\mathbf{V}_C & \mathbf{F}\mathbf{V}_F \end{pmatrix},$$
$$\mathbf{V}_r\mathbf{P}_{r\underline{\leftrightarrow}_{ss}} = \begin{pmatrix} \mathbf{V}_C & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_F \end{pmatrix}\begin{pmatrix} \mathbf{C}_{\underline{\leftrightarrow}_{ss}} & \mathbf{D}_{\underline{\leftrightarrow}_{ss}} \\ \mathbf{E}_{\underline{\leftrightarrow}_{ss}} & \mathbf{F}_{\underline{\leftrightarrow}_{ss}} \end{pmatrix} = \begin{pmatrix} \mathbf{V}_C\mathbf{C}_{\underline{\leftrightarrow}_{ss}} & \mathbf{V}_C\mathbf{D}_{\underline{\leftrightarrow}_{ss}} \\ \mathbf{V}_F\mathbf{E}_{\underline{\leftrightarrow}_{ss}} & \mathbf{V}_F\mathbf{F}_{\underline{\leftrightarrow}_{ss}} \end{pmatrix}.$$

Hence, $\mathbf{C}\mathbf{V}_C = \mathbf{V}_C\mathbf{C}_{\underline{\leftrightarrow}_{ss}}$, $\mathbf{D}\mathbf{V}_F = \mathbf{V}_C\mathbf{D}_{\underline{\leftrightarrow}_{ss}}$, $\mathbf{E}\mathbf{V}_C = \mathbf{V}_F\mathbf{E}_{\underline{\leftrightarrow}_{ss}}$, $\mathbf{F}\mathbf{V}_F = \mathbf{V}_F\mathbf{F}_{\underline{\leftrightarrow}_{ss}}$.

Let us show that $\mathbf{G}\mathbf{V}_C = \mathbf{V}_C\mathbf{G}_{\underline{\leftrightarrow}_{ss}}$. Since $\mathbf{G} = \sum_{k=0}^{\infty}\mathbf{C}^k$, it is sufficient to prove $\left(\sum_{k=0}^{l}\mathbf{C}^k\right)\mathbf{V}_C = \mathbf{V}_C\sum_{k=0}^{l}\mathbf{C}^k_{\underline{\leftrightarrow}_{ss}}$ by induction on $l \in \boldsymbol{N}$ and then take a limit $l \to \infty$.

- $l = 0$

  We have $\left(\sum_{k=0}^{0}\mathbf{C}^k\right)\mathbf{V}_C = \mathbf{C}^0\mathbf{V}_C = \mathbf{I}\mathbf{V}_C = \mathbf{V}_C = \mathbf{V}_C\mathbf{I} = \mathbf{V}_C\mathbf{C}^0_{\underline{\leftrightarrow}_{ss}} = \mathbf{V}_C\sum_{k=0}^{0}\mathbf{C}^k_{\underline{\leftrightarrow}_{ss}}$.

- $l \to l+1$

  Suppose that $\left(\sum_{k=0}^{l}\mathbf{C}^k\right)\mathbf{V}_C = \mathbf{V}_C\sum_{k=0}^{l}\mathbf{C}^k_{\underline{\leftrightarrow}_{ss}}$. Then $\left(\sum_{k=0}^{l+1}\mathbf{C}^k\right)\mathbf{V}_C = \left(\mathbf{I} + \mathbf{C}\sum_{k=0}^{l}\mathbf{C}^k\right)\mathbf{V}_C = \mathbf{V}_C{+}\mathbf{C}\mathbf{V}_C\sum_{k=0}^{l}\mathbf{C}^k_{\underline{\leftrightarrow}_{ss}} = \mathbf{V}_C + \mathbf{V}_C\mathbf{C}_{\underline{\leftrightarrow}_{ss}}\sum_{k=0}^{l}\mathbf{C}^k_{\underline{\leftrightarrow}_{ss}} = \mathbf{V}_C\left(\mathbf{I} + \mathbf{C}_{\underline{\leftrightarrow}_{ss}}\sum_{k=0}^{l}\mathbf{C}^k_{\underline{\leftrightarrow}_{ss}}\right) = \mathbf{V}_C\sum_{k=0}^{l+1}\mathbf{C}^k_{\underline{\leftrightarrow}_{ss}}$.

Next, $\mathbf{P}^{\diamond}\mathbf{V}_F = (\mathbf{F} + \mathbf{E}\mathbf{G}\mathbf{D})\mathbf{V}_F = \mathbf{F}\mathbf{V}_F + \mathbf{E}\mathbf{G}\mathbf{D}\mathbf{V}_F = \mathbf{V}_F\mathbf{F}_{\underline{\leftrightarrow}_{ss}} + \mathbf{E}\mathbf{G}\mathbf{V}_C\mathbf{D}_{\underline{\leftrightarrow}_{ss}} = \mathbf{V}_F\mathbf{F}_{\underline{\leftrightarrow}_{ss}} + \mathbf{E}\mathbf{V}_C\mathbf{G}_{\underline{\leftrightarrow}_{ss}}\mathbf{D}_{\underline{\leftrightarrow}_{ss}} = \mathbf{V}_F\mathbf{F}_{\underline{\leftrightarrow}_{ss}} + \mathbf{V}_F\mathbf{E}_{\underline{\leftrightarrow}_{ss}}\mathbf{G}_{\underline{\leftrightarrow}_{ss}}\mathbf{D}_{\underline{\leftrightarrow}_{ss}} = \mathbf{V}_F(\mathbf{F}_{\underline{\leftrightarrow}_{ss}} + \mathbf{E}_{\underline{\leftrightarrow}_{ss}}\mathbf{G}_{\underline{\leftrightarrow}_{ss}}\mathbf{D}_{\underline{\leftrightarrow}_{ss}}) = \mathbf{V}_F\mathbf{P}^{\diamond}_{\underline{\leftrightarrow}_{ss}}$. After left-multiplying by $\mathbf{W}_F$ the resulting equality $\mathbf{P}^{\diamond}\mathbf{V}_F = \mathbf{V}_F\mathbf{P}^{\diamond}_{\underline{\leftrightarrow}_{ss}}$, we finally get

$$(\mathbf{P}^{\diamond})_{\underset{\leftrightarrow}{}_{ss}} = \mathbf{W}_F \mathbf{P}^{\diamond} \mathbf{V}_F = \mathbf{P}^{\diamond}_{\underset{\leftrightarrow}{}_{ss}}.$$

□

Appendix A.6. Proof of Proposition 8.1

By Proposition 6.1, $(DR(G) \cup DR(G'))/_{\mathcal{R}} = ((DR_T(G) \cup DR_T(G'))/_{\mathcal{R}}) \uplus ((DR_V(G) \cup DR_V(G'))/_{\mathcal{R}})$. Hence, $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$, all states from $\mathcal{H}$ are tangible, when $\mathcal{H} \in (DR_T(G) \cup DR_T(G'))/_{\mathcal{R}}$, or all of them are vanishing, when $\mathcal{H} \in (DR_V(G) \cup DR_V(G'))/_{\mathcal{R}}$.

By definition of the steady-state PMFs for SMCs, $\forall s \in DR_V(G),\ \varphi(s) = 0$ and $\forall s' \in DR_V(G'),\ \varphi'(s') = 0$. Thus, $\forall \mathcal{H} \in (DR_V(G) \cup DR_V(G'))/_{\mathcal{R}},\ \sum_{s \in \mathcal{H} \cap DR(G)} \varphi(s) = \sum_{s \in \mathcal{H} \cap DR_V(G)} \varphi(s) = 0 = \sum_{s' \in \mathcal{H} \cap DR_V(G')} \varphi'(s') = \sum_{s' \in \mathcal{H} \cap DR(G')} \varphi'(s')$.

By Proposition 5.2, $\forall s \in DR_T(G),\ \varphi(s) = \frac{\psi(s)}{\sum_{\tilde{s} \in DR_T(G)} \psi(\tilde{s})}$ and $\forall s' \in DR_T(G'),\ \varphi'(s') = \frac{\psi'(s')}{\sum_{\tilde{s}' \in DR_T(G')} \psi'(\tilde{s}')}$, where ψ and ψ' are the steady-state PMFs for $DTMC(G)$ and $DTMC(G')$, respectively. Thus, $\forall \mathcal{H}, \widetilde{\mathcal{H}} \in (DR_T(G) \cup DR_T(G'))/_{\mathcal{R}}$, $\sum_{s \in \mathcal{H} \cap DR(G)} \varphi(s) = \sum_{s \in \mathcal{H} \cap DR_T(G)} \varphi(s) = \sum_{s \in \mathcal{H} \cap DR_T(G)} \left(\frac{\psi(s)}{\sum_{\tilde{s} \in DR_T(G)} \psi(\tilde{s})} \right) = \frac{\sum_{s \in \mathcal{H} \cap DR_T(G)} \psi(s)}{\sum_{\tilde{s} \in DR_T(G)} \psi(\tilde{s})} = \frac{\sum_{s \in \mathcal{H} \cap DR_T(G)} \psi(s)}{\sum_{\widetilde{\mathcal{H}}} \sum_{\tilde{s} \in \widetilde{\mathcal{H}} \cap DR_T(G)} \psi(\tilde{s})}$ and $\sum_{s' \in \mathcal{H} \cap DR(G')} \varphi'(s') = \sum_{s' \in \mathcal{H} \cap DR_T(G')} \varphi'(s') = \sum_{s' \in \mathcal{H} \cap DR_T(G')} \left(\frac{\psi'(s')}{\sum_{\tilde{s}' \in DR_T(G')} \psi'(\tilde{s}')} \right) = \frac{\sum_{s' \in \mathcal{H} \cap DR_T(G')} \psi'(s')}{\sum_{\tilde{s}' \in DR_T(G')} \psi'(\tilde{s}')} = \frac{\sum_{s' \in \mathcal{H} \cap DR_T(G')} \psi'(s')}{\sum_{\widetilde{\mathcal{H}}} \sum_{\tilde{s}' \in \widetilde{\mathcal{H}} \cap DR_T(G')} \psi'(\tilde{s}')}$.

It remains to prove that $\forall \mathcal{H} \in (DR_T(G) \cup DR_T(G'))/_{\mathcal{R}},\ \sum_{s \in \mathcal{H} \cap DR_T(G)} \psi(s) = \sum_{s' \in \mathcal{H} \cap DR_T(G')} \psi'(s')$. Since $(DR(G) \cup DR(G'))/_{\mathcal{R}} = ((DR_T(G) \cup DR_T(G'))/_{\mathcal{R}}) \uplus ((DR_V(G) \cup DR_V(G'))/_{\mathcal{R}})$, the previous equality is a consequence of the following: $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}},\ \sum_{s \in \mathcal{H} \cap DR(G)} \psi(s) = \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'(s')$.

Standard proof continuation.

It is sufficient to prove the previous statement for transient PMFs only, since $\psi = \lim_{k \to \infty} \psi[k]$ and $\psi' = \lim_{k \to \infty} \psi'[k]$. We proceed by induction on k.

- $k = 0$

  The only non-zero values of the initial PMFs of $DTMC(G)$ and $DTMC(G')$ are $\psi[0]([G]_{\approx})$ and $\psi[0]([G']_{\approx})$. Let $\mathcal{H}_0$ be the equivalence class containing $[G]_{\approx}$ and $[G']_{\approx}$. Then $\sum_{s \in \mathcal{H}_0 \cap DR(G)} \psi[0](s) = \psi[0]([G]_{\approx}) = 1 = \psi'[0]([G']_{\approx}) = \sum_{s' \in \mathcal{H}_0 \cap DR(G')} \psi'[0](s')$.

  As for other equivalence classes, $\forall \mathcal{H} \in ((DR(G) \cup DR(G'))/_{\mathcal{R}}) \setminus \mathcal{H}_0$, we have $\sum_{s \in \mathcal{H} \cap DR(G)} \psi[0](s) = 0 = \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'[0](s')$.

- $k \to k + 1$

  Let $\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$ and $s_1, s_2 \in \mathcal{H}$. We have $\forall \widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}},\ \forall A \in \boldsymbol{N}^{\mathcal{L}}_{fin},\ s_1 \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}} \Leftrightarrow s_2 \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}}$. Therefore, $PM(s_1, \widetilde{\mathcal{H}}) = \sum_{\{\Upsilon \mid \exists \tilde{s}_1 \in \widetilde{\mathcal{H}},\ s_1 \xrightarrow{\Upsilon} \tilde{s}_1\}} PT(\Upsilon, s_1) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} \sum_{\{\Upsilon \mid \exists \tilde{s}_1 \in \widetilde{\mathcal{H}},\ s_1 \xrightarrow{\Upsilon} \tilde{s}_1,\ \mathcal{L}(\Upsilon) = A\}} PT(\Upsilon, s_1) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} PM_A(s_1, \widetilde{\mathcal{H}}) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} PM_A(s_2, \widetilde{\mathcal{H}}) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} \sum_{\{\Upsilon \mid \exists \tilde{s}_2 \in \widetilde{\mathcal{H}},\ s_2 \xrightarrow{\Upsilon} \tilde{s}_2,\ \mathcal{L}(\Upsilon) = A\}} PT(\Upsilon, s_2) = \sum_{\{\Upsilon \mid \exists \tilde{s}_2 \in \widetilde{\mathcal{H}},\ s_2 \xrightarrow{\Upsilon} \tilde{s}_2\}} PT(\Upsilon, s_2) = PM(s_2, \widetilde{\mathcal{H}})$. Since we have the previous equality for all $s_1, s_2 \in \mathcal{H}$, we can denote $PM(\mathcal{H}, \widetilde{\mathcal{H}}) = PM(s_1, \widetilde{\mathcal{H}}) = PM(s_2, \widetilde{\mathcal{H}})$. Note that transitions from the states of $DR(G)$ always lead to those from the same set, hence, $\forall s \in DR(G),\ PM(s, \widetilde{\mathcal{H}}) = PM(s, \widetilde{\mathcal{H}} \cap DR(G))$. The same is true for $DR(G')$.

  By induction hypothesis, $\sum_{s \in \mathcal{H} \cap DR(G)} \psi[k](s) = \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'[k](s')$. Further,
  $\sum_{\tilde{s} \in \widetilde{\mathcal{H}} \cap DR(G)} \psi[k+1](\tilde{s}) = \sum_{\tilde{s} \in \widetilde{\mathcal{H}} \cap DR(G)} \sum_{s \in DR(G)} \psi[k](s) PM(s, \tilde{s}) = \sum_{s \in DR(G)} \sum_{\tilde{s} \in \widetilde{\mathcal{H}} \cap DR(G)} \psi[k](s) PM(s, \tilde{s}) =$
  $\sum_{s \in DR(G)} \psi[k](s) \sum_{\tilde{s} \in \widetilde{\mathcal{H}} \cap DR(G)} PM(s, \tilde{s}) = \sum_{\mathcal{H}} \sum_{s \in \mathcal{H} \cap DR(G)} \psi[k](s) \sum_{\tilde{s} \in \widetilde{\mathcal{H}} \cap DR(G)} PM(s, \tilde{s}) =$
  $\sum_{\mathcal{H}} \sum_{s \in \mathcal{H} \cap DR(G)} \psi[k](s) \sum_{\tilde{s} \in \widetilde{\mathcal{H}} \cap DR(G)} \sum_{\{\Upsilon \mid s \xrightarrow{\Upsilon} \tilde{s}\}} PT(\Upsilon, s) = \sum_{\mathcal{H}} \sum_{s \in \mathcal{H} \cap DR(G)} \psi[k](s) \sum_{\{\Upsilon \mid \exists \tilde{s} \in \widetilde{\mathcal{H}} \cap DR(G),\ s \xrightarrow{\Upsilon} \tilde{s}\}} PT(\Upsilon, s) =$

  $\sum_{\mathcal{H}} \sum_{s \in \mathcal{H} \cap DR(G)} \psi[k](s) PM(s, \widetilde{\mathcal{H}}) = \sum_{\mathcal{H}} \sum_{s \in \mathcal{H} \cap DR(G)} \psi[k](s) PM(\mathcal{H}, \widetilde{\mathcal{H}}) =$
  $\sum_{\mathcal{H}} PM(\mathcal{H}, \widetilde{\mathcal{H}}) \sum_{s \in \mathcal{H} \cap DR(G)} \psi[k](s) = \sum_{\mathcal{H}} PM(\mathcal{H}, \widetilde{\mathcal{H}}) \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'[k](s') =$
  $\sum_{\mathcal{H}} \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'[k](s') PM(\mathcal{H}, \widetilde{\mathcal{H}}) = \sum_{\mathcal{H}} \sum_{s' \in \mathcal{H}' \cap DR(G')} \psi'[k](s') PM(s', \widetilde{\mathcal{H}}) =$
  $\sum_{\mathcal{H}} \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'[k](s') \sum_{\{\Upsilon \mid \exists \tilde{s}' \in \widetilde{\mathcal{H}} \cap DR(G'),\ s' \xrightarrow{\Upsilon} \tilde{s}'\}} PT(\Upsilon, s') =$
  $\sum_{\mathcal{H}} \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'[k](s') \sum_{\tilde{s}' \in \widetilde{\mathcal{H}} \cap DR(G')} \sum_{\{\Upsilon \mid \exists \tilde{s}',\ s' \xrightarrow{\Upsilon} \tilde{s}'\}} PT(\Upsilon, s') =$
  $\sum_{\mathcal{H}} \sum_{s' \in \mathcal{H} \cap DR(G')} \psi'[k](s') \sum_{\tilde{s}' \in \widetilde{\mathcal{H}} \cap DR(G')} PM(s', \tilde{s}') = \sum_{s' \in DR(G')} \psi'[k](s') \sum_{\tilde{s}' \in \widetilde{\mathcal{H}} \cap DR(G')} PM(s', \tilde{s}') =$

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

Alternative proof continuation.

Thus, we should now prove that $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}},\ \sum_{\{i | s_i \in \mathcal{H} \cap DR(G)\}} \psi_i = \sum_{\{j | s'_j \in \mathcal{H} \cap DR(G')\}} \psi'_j$.

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

$$\begin{cases} \psi \mathbf{P} = \psi \\ \psi \mathbf{1}^T = 1 \end{cases}.$$

Then, for all i $(1 \le i \le n)$, we have

$$\begin{cases} \sum_{j=1}^{n} \mathcal{P}_{ji} \psi_j = \psi_i \\ \sum_{j=1}^{n} \psi_j = 1 \end{cases}.$$

By definition of $\mathcal{P}_{ij}$ $(1 \le i, j \le n)$ we have

$$\begin{cases} \sum_{j=1}^{n} PM(s_j, s_i) \psi_j = \psi_i \\ \sum_{j=1}^{n} \psi_j = 1 \end{cases}.$$

Let $\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$ and $s_1, s_2 \in \mathcal{H}$. We have $\forall \widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}},\ \forall A \in \boldsymbol{N}^{\mathcal{L}}_{fin},\ s_1 \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}} \Leftrightarrow$ $s_2 \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}}$. Therefore, $PM(s_1, \widetilde{\mathcal{H}}) = \sum_{\{\Upsilon | \exists \tilde{s}_1 \in \widetilde{\mathcal{H}}\ s_1 \xrightarrow{\Upsilon} \tilde{s}_1\}} PT(\Upsilon, s_1) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} \sum_{\{\Upsilon | \exists \tilde{s}_1 \in \widetilde{\mathcal{H}}\ s_1 \xrightarrow{\Upsilon} \tilde{s}_1,\ \mathcal{L}(\Upsilon) = A\}} PT(\Upsilon, s_1) =$
$\sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} PM_A(s_1, \widetilde{\mathcal{H}}) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} PM_A(s_2, \widetilde{\mathcal{H}}) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} \sum_{\{\Upsilon | \exists \tilde{s}_2 \in \widetilde{\mathcal{H}}\ s_2 \xrightarrow{\Upsilon} \tilde{s}_2,\ \mathcal{L}(\Upsilon) = A\}} PT(\Upsilon, s_2) = \sum_{\{\Upsilon | \exists \tilde{s}_2 \in \widetilde{\mathcal{H}}\ s_2 \xrightarrow{\Upsilon} \tilde{s}_2\}} PT(\Upsilon, s_2) =$
$PM(s_2, \widetilde{\mathcal{H}})$. Since we have the previous equality for all $s_1, s_2 \in \mathcal{H}$, we can denote $PM(\mathcal{H}, \widetilde{\mathcal{H}}) = PM(s_1, \widetilde{\mathcal{H}}) = PM(s_2, \widetilde{\mathcal{H}})$. Note that transitions from the states of $DR(G)$ always lead to those from the same set, hence, $\forall s \in DR(G),\ PM(s, \widetilde{\mathcal{H}}) = PM(s, \widetilde{\mathcal{H}} \cap DR(G))$. The same is true for $DR(G')$.

Let $\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$. We sum the left and right sides of the first equation from the system above for all i such that $s_i \in \mathcal{H} \cap DR(G)$. The resulting equation is

$$\sum_{\{i | s_i \in \mathcal{H} \cap DR(G)\}} \sum_{j=1}^{n} PM(s_j, s_i) \psi_j = \sum_{\{i | s_i \in \mathcal{H} \cap DR(G)\}} \psi_i.$$

Let us denote the aggregate steady-state PMF for $DTMC(G)$ by $\psi_{\mathcal{H} \cap DR(G)} = \sum_{\{i | s_i \in \mathcal{H} \cap DR(G)\}} \psi_i$. Then, for the left-hand side of the equation above, we get
$\sum_{\{i | s_i \in \mathcal{H} \cap DR(G)\}} \sum_{j=1}^{n} PM(s_j, s_i) \psi_j = \sum_{j=1}^{n} \psi_j \sum_{\{i | s_i \in \mathcal{H} \cap DR(G)\}} PM(s_j, s_i) = \sum_{j=1}^{n} PM(s_j, \mathcal{H}) \psi_j =$
$\sum_{\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}} \sum_{\{j | s_j \in \widetilde{\mathcal{H}} \cap DR(G)\}} PM(s_j, \mathcal{H}) \psi_j = \sum_{\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}} \sum_{\{j | s_j \in \widetilde{\mathcal{H}} \cap DR(G)\}} PM(\widetilde{\mathcal{H}}, \mathcal{H}) \psi_j =$
$\sum_{\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}} PM(\widetilde{\mathcal{H}}, \mathcal{H}) \sum_{\{j | s_j \in \widetilde{\mathcal{H}} \cap DR(G)\}} \psi_j = \sum_{\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}} PM(\widetilde{\mathcal{H}}, \mathcal{H}) \psi_{\widetilde{\mathcal{H}} \cap DR(G)}$.

For the left-hand side of the second equation from the system above, we have
$\sum_{j=1}^{n} \psi_j = \sum_{\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}} \sum_{\{j | s_j \in \widetilde{\mathcal{H}} \cap DR(G)\}} \psi_j = \sum_{\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}} \psi_{\widetilde{\mathcal{H}} \cap DR(G)}$.

Thus, the aggregate linear equation system for $DTMC(G)$ is

$$\begin{cases} \sum_{\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}} PM(\widetilde{\mathcal{H}}, \mathcal{H}) \psi_{\widetilde{\mathcal{H}} \cap DR(G)} = \psi_{\mathcal{H} \cap DR(G)} \\ \sum_{\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}} \psi_{\widetilde{\mathcal{H}} \cap DR(G)} = 1 \end{cases}.$$

Let us denote the aggregate steady-state PMFs for $DTMC(G')$ by $\psi'_{\mathcal{H} \cap DR(G')} = \sum_{\{j | s'_j \in \mathcal{H} \cap DR(G')\}} \psi'_j$. Then, in a similar way, the aggregate linear equation system for $DTMC(G')$ is

$$\begin{cases} \sum_{\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}} PM(\widetilde{\mathcal{H}}, \mathcal{H}) \psi'_{\widetilde{\mathcal{H}} \cap DR(G')} = \psi'_{\mathcal{H} \cap DR(G')} \\ \sum_{\widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}} \psi'_{\widetilde{\mathcal{H}} \cap DR(G')} = 1 \end{cases}.$$

Let $(DR(G) \cup DR(G'))/_{\mathcal{R}} = \{\mathcal{H}_1, \ldots, \mathcal{H}_l\}$. Then the aggregate steady-state PMFs $\psi_{\mathcal{H}_k \cap DR(G)}$ and $\psi'_{\mathcal{H}_k \cap DR(G')}$ $(1 \le k \le l)$ satisfy the same aggregate system of $l + 1$ linear equations with l independent equations and l unknowns. The

aggregate linear equation system has a unique solution, when a single aggregate steady-state PMF exists. This is the case here, since in Section 5 we have demonstrated that $DTMC(G)$ has a single steady state iff $SMC(G)$ has, and aggregation preserves this property [26]. Hence, $\psi_{\mathcal{H}_k \cap DR(G)} = \psi'_{\mathcal{H}_k \cap DR(G')}$ $(1 \le k \le l)$. □

Appendix A.7. Proof of Theorem 8.1

Let $\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$ and $s, \bar{s} \in \mathcal{H}$. We have $\forall \widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}},\ \forall A \in \boldsymbol{N}^{\mathcal{L}}_{fin},\ s \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}} \ \Leftrightarrow$ $\bar{s} \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}}$. The previous equality is valid for all $s, \bar{s} \in \mathcal{H}$, hence, we can rewrite it as $\mathcal{H} \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}}$ and denote $PM_A(\mathcal{H}, \widetilde{\mathcal{H}}) = PM_A(s, \widetilde{\mathcal{H}}) = PM_A(\bar{s}, \widetilde{\mathcal{H}})$. Note that transitions from the states of $DR(G)$ always lead to those from the same set, hence, $\forall s \in DR(G),\ PM_A(s, \widetilde{\mathcal{H}}) = PM_A(s, \widetilde{\mathcal{H}} \cap DR(G))$. The same is true for $DR(G')$.

Let $\Sigma = A_1 \cdots A_n$ be a derived step trace of G and G'. Then $\exists \mathcal{H}_0, \ldots, \mathcal{H}_n \in (DR(G) \cup DR(G'))/_{\mathcal{R}},\ \mathcal{H}_0 \xrightarrow{A_1}_{\mathcal{P}_1} \mathcal{H}_1 \xrightarrow{A_2}_{\mathcal{P}_2} \cdots \xrightarrow{A_n}_{\mathcal{P}_n} \mathcal{H}_n$. We now intend to prove that the sum of probabilities of all the paths starting in every $s_0 \in \mathcal{H}_0$ and going through the states from $\mathcal{H}_1, \ldots, \mathcal{H}_n$ is equal to the product of $\mathcal{P}_1, \ldots, \mathcal{P}_n$:

$$\sum_{\{\Upsilon_1,\ldots,\Upsilon_n | s_0 \xrightarrow{\Upsilon_1} \cdots \xrightarrow{\Upsilon_n} s_n,\ \mathcal{L}(\Upsilon_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \le i \le n)\}} \prod_{i=1}^{n} PT(\Upsilon_i, s_{i-1}) = \prod_{i=1}^{n} PM_{A_i}(\mathcal{H}_{i-1}, \mathcal{H}_i).$$

We prove this equality by induction on the derived step trace length n.

- $n = 1$

  $\sum_{\{\Upsilon_1 | s_0 \xrightarrow{\Upsilon_1} s_1,\ \mathcal{L}(\Upsilon_1)=A_1,\ s_1 \in \mathcal{H}_1\}} PT(\Upsilon_1, s_0) = PM_{A_1}(s_0, \mathcal{H}_1) = PM_{A_1}(\mathcal{H}_0, \mathcal{H}_1).$

- $n \to n + 1$

  $\sum_{\{\Upsilon_1,\ldots,\Upsilon_n,\Upsilon_{n+1} | s_0 \xrightarrow{\Upsilon_1} \cdots \xrightarrow{\Upsilon_n} s_n \xrightarrow{\Upsilon_{n+1}} s_{n+1},\ \mathcal{L}(\Upsilon_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \le i \le n+1)\}} \prod_{i=1}^{n+1} PT(\Upsilon_i, s_{i-1}) =$
  $\sum_{\{\Upsilon_1,\ldots,\Upsilon_n | s_0 \xrightarrow{\Upsilon_1} \cdots \xrightarrow{\Upsilon_n} s_n,\ \mathcal{L}(\Upsilon_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \le i \le n)\}} \sum_{\{\Upsilon_{n+1} | s_n \xrightarrow{\Upsilon_{n+1}} s_{n+1},\ \mathcal{L}(\Upsilon_{n+1})=A_{n+1},\ s_n \in \mathcal{H}_n,\ s_{n+1} \in \mathcal{H}_{n+1}\}} \prod_{i=1}^{n} PT(\Upsilon_i, s_{i-1}) PT(\Upsilon_{n+1}, s_n) =$
  $\sum_{\{\Upsilon_1,\ldots,\Upsilon_n | s_0 \xrightarrow{\Upsilon_1} \cdots \xrightarrow{\Upsilon_n} s_n,\ \mathcal{L}(\Upsilon_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \le i \le n)\}} \left[\prod_{i=1}^{n} PT(\Upsilon_i, s_{i-1}) \sum_{\{\Upsilon_{n+1} | s_n \xrightarrow{\Upsilon_{n+1}} s_{n+1},\ \mathcal{L}(\Upsilon_{n+1})=A_{n+1},\ s_n \in \mathcal{H}_n,\ s_{n+1} \in \mathcal{H}_{n+1}\}} PT(\Upsilon_{n+1}, s_n) \right] =$
  $\sum_{\{\Upsilon_1,\ldots,\Upsilon_n | s_0 \xrightarrow{\Upsilon_1} \cdots \xrightarrow{\Upsilon_n} s_n,\ \mathcal{L}(\Upsilon_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \le i \le n)\}} \prod_{i=1}^{n} PT(\Upsilon_i, s_{i-1}) PM_{A_{n+1}}(s_n, \mathcal{H}_{n+1}) =$
  $\sum_{\{\Upsilon_1,\ldots,\Upsilon_n | s_0 \xrightarrow{\Upsilon_1} \cdots \xrightarrow{\Upsilon_n} s_n,\ \mathcal{L}(\Upsilon_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \le i \le n)\}} \prod_{i=1}^{n} PT(\Upsilon_i, s_{i-1}) PM_{A_{n+1}}(\mathcal{H}_n, \mathcal{H}_{n+1}) =$
  $PM_{A_{n+1}}(\mathcal{H}_n, \mathcal{H}_{n+1}) \sum_{\{\Upsilon_1,\ldots,\Upsilon_n | s_0 \xrightarrow{\Upsilon_1} \cdots \xrightarrow{\Upsilon_n} s_n,\ \mathcal{L}(\Upsilon_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \le i \le n)\}} \prod_{i=1}^{n} PT(\Upsilon_i, s_{i-1}) =$
  $PM_{A_{n+1}}(\mathcal{H}_n, \mathcal{H}_{n+1}) \prod_{i=1}^{n} PM_{A_i}(\mathcal{H}_{i-1}, \mathcal{H}_i) = \prod_{i=1}^{n+1} PM_{A_i}(\mathcal{H}_{i-1}, \mathcal{H}_i).$

Let $s_0, \bar{s}_0 \in \mathcal{H}_0$. We have $PT(A_1 \cdots A_n, s_0) = \sum_{\{\Upsilon_1,\ldots,\Upsilon_n | s_0 \xrightarrow{\Upsilon_1} \cdots \xrightarrow{\Upsilon_n} s_n,\ \mathcal{L}(\Upsilon_i)=A_i,\ (1 \le i \le n)\}} \prod_{i=1}^{n} PT(\Upsilon_i, s_{i-1}) =$
$\sum_{\mathcal{H}_1,\ldots,\mathcal{H}_n} \sum_{\{\Upsilon_1,\ldots,\Upsilon_n | s_0 \xrightarrow{\Upsilon_1} \cdots \xrightarrow{\Upsilon_n} s_n,\ \mathcal{L}(\Upsilon_i)=A_i,\ s_i \in \mathcal{H}_i\ (1 \le i \le n)\}} \prod_{i=1}^{n} PT(\Upsilon_i, s_{i-1}) = \sum_{\mathcal{H}_1,\ldots,\mathcal{H}_n} \prod_{i=1}^{n} PM_{A_i}(\mathcal{H}_{i-1}, \mathcal{H}_i) =$
$\sum_{\mathcal{H}_1,\ldots,\mathcal{H}_n} \sum_{\{\overline{\Upsilon}_1,\ldots,\overline{\Upsilon}_n | \bar{s}_0 \xrightarrow{\overline{\Upsilon}_1} \cdots \xrightarrow{\overline{\Upsilon}_n} \bar{s}_n,\ \mathcal{L}(\overline{\Upsilon}_i)=A_i,\ \bar{s}_i \in \mathcal{H}_i\ (1 \le i \le n)\}} \prod_{i=1}^{n} PT(\overline{\Upsilon}_i, \bar{s}_{i-1}) =$
$\sum_{\{\overline{\Upsilon}_1,\ldots,\overline{\Upsilon}_n | \bar{s}_0 \xrightarrow{\overline{\Upsilon}_1} \cdots \xrightarrow{\overline{\Upsilon}_n} \bar{s}_n,\ \mathcal{L}(\overline{\Upsilon}_i)=A_i,\ (1 \le i \le n)\}} \prod_{i=1}^{n} PT(\overline{\Upsilon}_i, \bar{s}_{i-1}) = PT(A_1 \cdots A_n, \bar{s}_0).$
Since we have the previous equality for all $s_0, \bar{s}_0 \in \mathcal{H}_0$, we can denote $PT(A_1 \cdots A_n, \mathcal{H}_0) = PT(A_1 \cdots A_n, s_0) = PT(A_1 \cdots A_n, \bar{s}_0)$.

By Proposition 8.1, $\sum_{s \in \mathcal{H} \cap DR(G)} \varphi(s) = \sum_{s' \in \mathcal{H} \cap DR(G')} \varphi'(s')$. We now can complete the proof:
$\sum_{s \in \mathcal{H} \cap DR(G)} \varphi(s) PT(\Sigma, s) = \sum_{s \in \mathcal{H} \cap DR(G)} \varphi(s) PT(\Sigma, \mathcal{H}) = PT(\Sigma, \mathcal{H}) \sum_{s \in \mathcal{H} \cap DR(G)} \varphi(s) =$
$PT(\Sigma, \mathcal{H}) \sum_{s' \in \mathcal{H} \cap DR(G')} \varphi'(s') = \sum_{s' \in \mathcal{H} \cap DR(G')} \varphi'(s') PT(\Sigma, \mathcal{H}) = \sum_{s' \in \mathcal{H} \cap DR(G')} \varphi'(s') PT(\Sigma, s').$ □

Appendix A.8. Proof of Proposition 8.2

Let us present two facts, which will be used in the proof.

1. By Proposition 6.1, $(DR(G) \cup DR(G'))/_{\mathcal{R}} = ((DR_T(G) \cup DR_T(G'))/_{\mathcal{R}}) \uplus ((DR_V(G) \cup DR_V(G'))/_{\mathcal{R}})$. Hence, $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$, all states from $\mathcal{H}$ are tangible, when $\mathcal{H} \in (DR_T(G) \cup DR_T(G'))/_{\mathcal{R}}$, or all of them are vanishing, when $\mathcal{H} \in (DR_V(G) \cup DR_V(G'))/_{\mathcal{R}}$.

2. Let $\mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$ and $s_1, s_2 \in \mathcal{H}$. We have $\forall \widetilde{\mathcal{H}} \in (DR(G) \cup DR(G'))/_{\mathcal{R}},\ \forall A \in \boldsymbol{N}^{\mathcal{L}}_{fin},\ s_1 \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}} \Leftrightarrow s_2 \xrightarrow{A}_{\mathcal{P}} \widetilde{\mathcal{H}}$. Therefore, $PM(s_1, \widetilde{\mathcal{H}}) = \sum_{\{\Upsilon \mid \exists \tilde{s}_1 \in \widetilde{\mathcal{H}},\ s_1 \xrightarrow{\Upsilon} \tilde{s}_1\}} PT(\Upsilon, s_1) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} \sum_{\{\Upsilon \mid \exists \tilde{s}_1 \in \widetilde{\mathcal{H}},\ s_1 \xrightarrow{\Upsilon} \tilde{s}_1,\ \mathcal{L}(\Upsilon)=A\}} PT(\Upsilon, s_1) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} PM_A(s_1, \widetilde{\mathcal{H}}) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} PM_A(s_2, \widetilde{\mathcal{H}}) = \sum_{A \in \boldsymbol{N}^{\mathcal{L}}_{fin}} \sum_{\{\Upsilon \mid \exists \tilde{s}_2 \in \widetilde{\mathcal{H}},\ s_2 \xrightarrow{\Upsilon} \tilde{s}_2,\ \mathcal{L}(\Upsilon)=A\}} PT(\Upsilon, s_2) = \sum_{\{\Upsilon \mid \exists \tilde{s}_2 \in \widetilde{\mathcal{H}},\ s_2 \xrightarrow{\Upsilon} \tilde{s}_2\}} PT(\Upsilon, s_2) = PM(s_2, \widetilde{\mathcal{H}})$. Since we have the previous equality for all $s_1, s_2 \in \mathcal{H}$, we can denote $PM(\mathcal{H}, \widetilde{\mathcal{H}}) = PM(s_1, \widetilde{\mathcal{H}}) = PM(s_2, \widetilde{\mathcal{H}})$. The transitions from the states of $DR(G)$ always lead to those from the same set, hence, $\forall s \in DR(G),\ PM(s, \widetilde{\mathcal{H}}) = PM(s, \widetilde{\mathcal{H}} \cap DR(G))$. The same is true for $DR(G')$. Hence, for all $s \in \mathcal{H} \cap DR(G)$, we obtain $PM(\mathcal{H}, \mathcal{H}) = PM(s, \mathcal{H}) = PM(s, \mathcal{H} \cap DR(G)) = PM(\mathcal{H} \cap DR(G), \mathcal{H} \cap DR(G))$. The same is true for $DR(G')$. Finally, $PM(\mathcal{H} \cap DR(G), \widetilde{\mathcal{H}} \cap DR(G)) = PM(\mathcal{H}, \widetilde{\mathcal{H}}) = PM(\mathcal{H} \cap DR(G'), \widetilde{\mathcal{H}} \cap DR(G'))$.

Let us now prove the proposition statement for the sojourn time averages.

- Let $\mathcal{H} \in (DR_V(G) \cup DR_V(G'))/_{\mathcal{R}}$.

  Then we have $\mathcal{H} \cap DR(G) = \mathcal{H} \cap DR_V(G) \in DR_V(G)/_{\mathcal{R}}$ and $\mathcal{H} \cap DR(G') = \mathcal{H} \cap DR_V(G') \in DR_V(G')/_{\mathcal{R}}$.
  By definition of the average sojourn time in an equivalence class of states, we get $SJ_{\mathcal{R} \cap (DR(G))^2}(\mathcal{H} \cap DR(G)) = SJ_{\mathcal{R} \cap (DR(G))^2}(\mathcal{H} \cap DR_V(G)) = 0 = SJ_{\mathcal{R} \cap (DR(G'))^2}(\mathcal{H} \cap DR_V(G')) = SJ_{\mathcal{R} \cap (DR(G'))^2}(\mathcal{H} \cap DR(G'))$.

- Let $\mathcal{H} \in (DR_T(G) \cup DR_T(G'))/_{\mathcal{R}}$.

  Then we have $\mathcal{H} \cap DR(G) = \mathcal{H} \cap DR_T(G) \in DR_T(G)/_{\mathcal{R}}$ and $\mathcal{H} \cap DR(G') = \mathcal{H} \cap DR_T(G') \in DR_T(G')/_{\mathcal{R}}$.
  By definition of the average sojourn time in an equivalence class of states, we get $SJ_{\mathcal{R} \cap (DR(G))^2}(\mathcal{H} \cap DR(G)) = SJ_{\mathcal{R} \cap (DR(G))^2}(\mathcal{H} \cap DR_T(G)) = \frac{1}{1 - PM(\mathcal{H} \cap DR_T(G), \mathcal{H} \cap DR_T(G))} = \frac{1}{1 - PM(\mathcal{H} \cap DR(G), \mathcal{H} \cap DR(G))} = \frac{1}{1 - PM(\mathcal{H}, \mathcal{H})} = \frac{1}{1 - PM(\mathcal{H} \cap DR(G'), \mathcal{H} \cap DR(G'))} = \frac{1}{1 - PM(\mathcal{H} \cap DR_T(G'), \mathcal{H} \cap DR_T(G'))} = SJ_{\mathcal{R} \cap (DR(G'))^2}(\mathcal{H} \cap DR_T(G')) = SJ_{\mathcal{R} \cap (DR(G'))^2}(\mathcal{H} \cap DR(G'))$.

Thus, $\forall \mathcal{H} \in (DR(G) \cup DR(G'))/_{\mathcal{R}}$ we have $SJ_{\mathcal{R} \cap (DR(G))^2}(\mathcal{H} \cap DR(G)) = SJ_{\mathcal{R} \cap (DR(G'))^2}(\mathcal{H} \cap DR(G'))$.
The proposition statement for the sojourn time variances is proved similarly to that for the averages. □