Process Algebra for Performance Evaluation [1]

Holger Hermanns [a] Ulrich Herzog [b] Joost-Pieter Katoen [b,2]

[a] *Systems Validation Centre, FMT/CTIT, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands*

[b] *Lehrstuhl für Informatik 7, IMMD, Friedrich-Alexander-Universität Erlangen-Nürnberg, Martensstrasse 3, 91058 Erlangen, Germany*

Abstract

This paper surveys the theoretical developments in the field of stochastic process algebras, process algebras where action occurrences may be subject to a delay that is determined by a random variable. A huge class of resource-sharing systems — like large-scale computers, client-server architectures, networks — can accurately be described using such stochastic specification formalisms.

The main emphasis of this paper is the treatment of operational semantics, notions of equivalence, and (sound and complete) axiomatisations of these equivalences for different types of Markovian process algebras, where delays are governed by exponential distributions. Starting from a simple action-less algebra for describing time-homogeneous continuous-time Markov chains, we consider the integration of actions and random delays both as a single entity (like in known Markovian process algebras like TIPP, PEPA and EMPA) and as separate entities (like in the timed process algebras timed CSP and TCCS). In total we consider four related calculi and investigate their relationship to existing Markovian process algebras. We also briefly indicate how one can profit from the separation of time and actions when incorporation more general, non-Markovian distributions.

Key words: axiomatisation; bisimulation; continuous-time Markov chain; lumpability; performance evaluation; process algebra; resource-sharing systems; semantics

[1] Supported by the German Research Council (Deutsche Forschungsgemeinschaft, SFB-Project 182), the European Commission (ESPRIT BRA QMIPS-Project), and the German Academic Exchange Council (Deutscher Akademischer Austauschdienst, BC/Vigoni/ARC-Project).

[2] Currently at the Computer Science Department of the University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands.

1 Introduction

1.1 Motivation

Performance evaluation means to describe, to analyse, and to optimise the dynamic, time-dependent behaviour of systems. However, it is rather common for a system to be fully designed and functionally tested before any attempt is made to determine its performance characteristics. Redesign of both, hardware and software, is costly and may cause late system delivery. Therefore, performance evaluation has to be integrated into the design process from the very beginning [46,31] (cf. Figure 1). The saw-tooth curve schematically shows the system development process when temporal aspects are considered at a too late stage. Early integration of both functional and temporal behaviour in a unified methodology allows to diminish these setbacks and improves the design productivity.

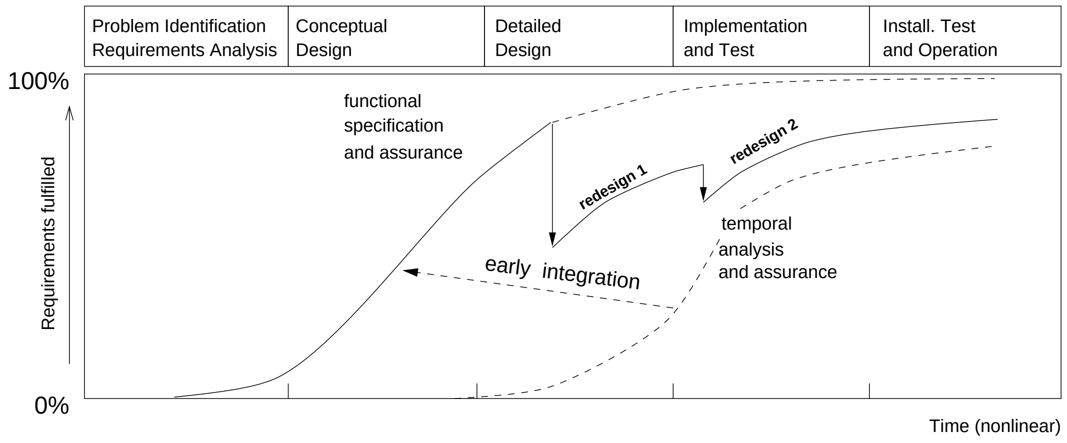

Fig. 1. System life cycle and quality assurance

The need for integrated modelling techniques was already recognised in the seventies. The most successful examples are stochastic Petri nets [3], stochastic graph models [91] and stochastic automata networks [85]. Nevertheless, performance evaluation of complex transportation and processing systems remained an art mastered only by a small group of specialists. Now, process algebras with their unique features offer means which promise a major progress for systematic modelling of complex systems. Therefore, some ten years ago a small group of researchers started to deal intensively with stochastic extensions of process algebras. In this paper we survey the state-of-the-art in the theoretical achievements for these stochastic process algebras, argue the benefits for performance evaluation (including reliability aspects) and indicate the future challenges.

1.2 Real-time versus resource-sharing systems

Transportation and processing systems may be split into two classes:

- *real-time systems* (e.g. systems for process control, manufacturing systems, robots, avionic control systems), and
- *resource-sharing systems* (e.g. time-sharing computers, mainframes, telephone and data-communication systems, production lines with work-over).

While the main objective of real-time system design is to guarantee the correctness of process interaction, in resource-sharing systems the economical use of resources is of prime importance.[3] Therefore, safety and liveness properties are most interesting for real-time systems while the analysis of throughput, utilisation, loss-probabilities and delays are the key features when planning resource sharing systems. Consequently, we also distinguish between different classes of timing and process models:

- *deterministic* timing for real-time systems, i.e. actions take place at distinct time instants or within fixed time intervals. Adequate process models for such systems are extended finite-state machines, timed Petri nets, timed automata, and timed process algebras, and the like,
- *probabilistic* timing for resource-sharing systems, i.e. contention, faults and mass phenomena lead to randomly varying time instants and time intervals. Well-known process models are queueing networks, stochastic Petri nets, and stochastic graphs models; the advantages of stochastic process algebras are becoming apparent step by step.

We focus on resource-sharing systems and their modelling by means of stochastic process algebras while being aware, however, that real-time system models include quite some potential for resource-sharing systems too.

1.3 Performance evaluation of resource-sharing systems

The purpose of performance evaluation is to investigate and optimise the dynamic, time-varying behaviour within and between the individual components of transportation and processing systems. We measure and model the temporal behaviour of real systems, define and determine characteristic performance measures, and develop design rules which guarantee an adequate quality of

[3] Note that depending on the level of abstraction, the same technical system may be viewed in one case as real-time system, in another as resource sharing system. This is particularly true for communication networks. Note also that a second, common requirement of both types of systems is fault-tolerance.

service.

As we have seen, sharing of resources is mandatory because of economical reasons, i.e. there is a varying number of demands (customers) competing for the same resources. The consequences are mutual interference, delays due to contention and varying service quality. Additionally, transmission errors and resource-failures do also influence significantly the system behaviour.

The concept of stochastic processes allows to accurately model and investigate these phenomena. Already in the beginning of this century first fundamental results were obtained for dimensioning telephone systems. Then, the advent of computer systems, computer networks and operations research required a tremendous lift of both theoretical advances and practical use of these techniques. However, despite the solid theoretical foundation on the one side and rich practical experience on the other, performance evaluation is still an art mastered by a small group of specialists. This is particularly true when the systems are large and when there are sophisticated interdependencies.

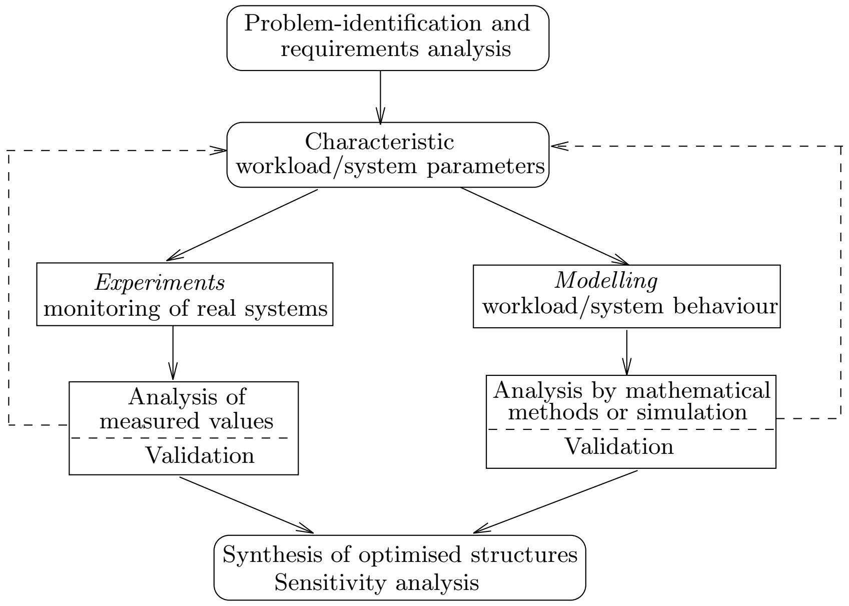
Problem-identification and requirements analysis
Characteristic workload/system parameters
Experiments monitoring of real systems
Modelling workload/system behaviour
Analysis of measured values
Validation
Analysis by mathematical methods or simulation
Validation
Synthesis of optimised structures
Sensitivity analysis

Fig. 2. Overview of performance evaluation methodology.

Process algebras with their unique features offer a framework and various concepts which can help to overcome these major problems of performance evaluation (PE) methodology. The main steps of today's PE-methodology are sketched in Figure 2, while Figure 3 shows a typical example aimed at optimising the throughput of a pick-and-place robot for electronic board equipping. Workload characterisation and system parameter specification are the first sensitive steps in the PE-methodology. Determining these values needs care and knowledge about both the application and the technical system components. Next, the design methodology distinguishes between two totally different but complementary approaches: *experiments* on the real system (measure-

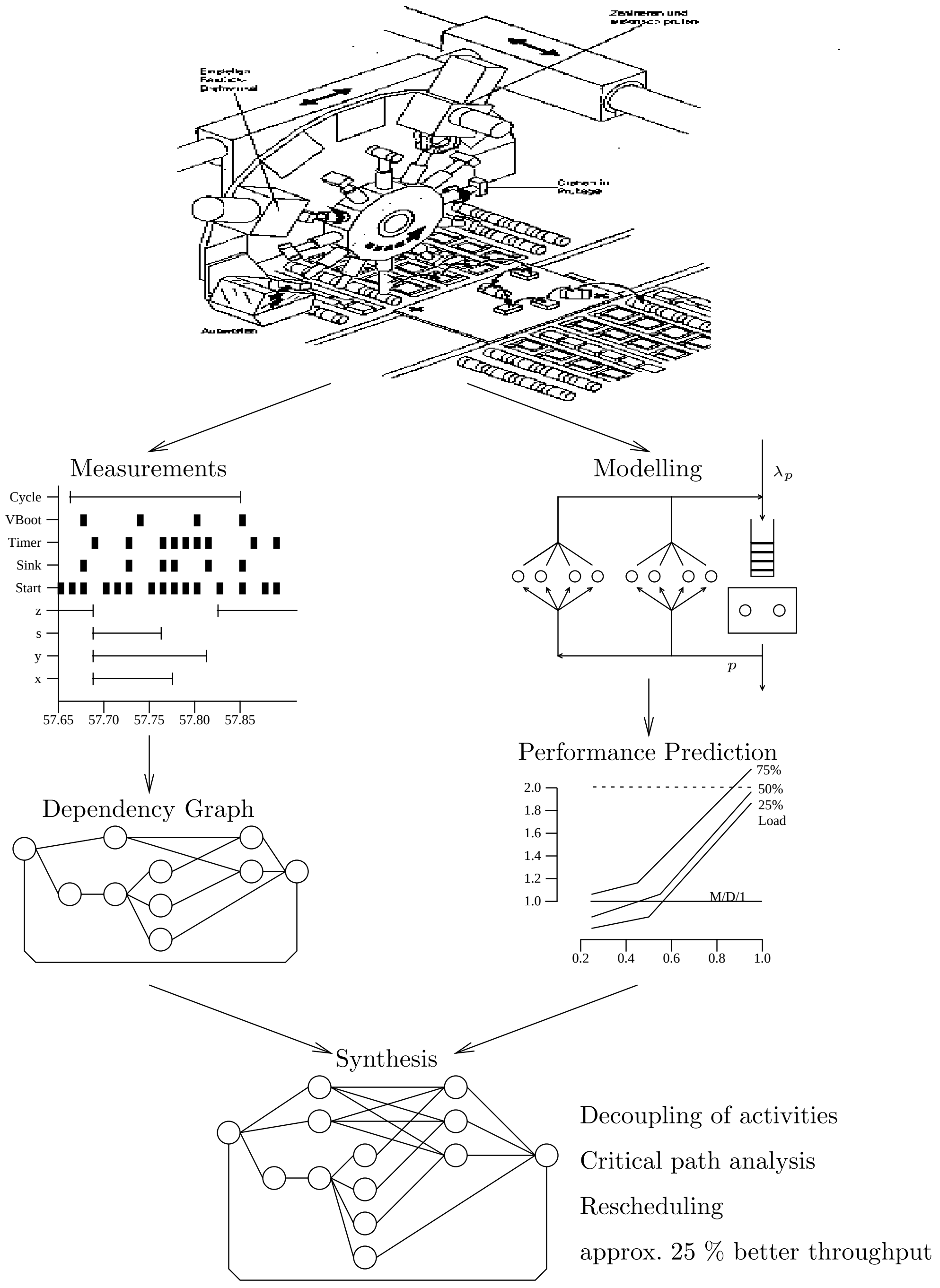

Fig. 3. PE-methodology applied to throughput optimisation of a pick-and-place robot.

ments) and *modelling*. Both are followed by analysis steps using methods of statistics, stochastic processes and simulation. Finally, system structures and operating modes are synthesised; systematic parameter variation (in case of experimentation and simulation) and mathematical optimisation techniques (in combination with stochastic models) guarantee good or even optimal system design considering costs and/or performance and a variety of optimisation constraints.

By following all steps of the PE-methodology, the throughput of an industrial pick-and-place robot could be improved by about 25%. This considerable gain was facilitated by careful measurements and skillful modelling of the system behaviour, decoupling of activities, critical path analysis and rescheduling strategies [105]. This is not a unique example for the success of PE-methodology and even more dramatic gains are known for e.g. integrated protocol design [98]. However, despite the high standard of performance monitoring, PE-theory and many supporting tools, very important problem-classes are not yet solved satisfactorily:

- For designing complex systems, *hierarchical and modular modelling* is mandatory. However, interfacing PE-sub-models accurately is still an art. Moreover, state-space exploration and reduction is often done by hand. Only a few PE-specialists are sufficiently experienced to reach reliable results. In order to improve design productivity and accuracy, much more automatic support is necessary.
- In modelling the performance of parallel and distributed systems, *task dependencies* are mostly neglected.
- Separate teams address the functional design and performance evaluation (cf. Fig. 1).

Stochastic process algebras offer a solid theoretical foundation and new prospects. They can help to solve the aforementioned technical problems in a systematic way, and can assist in diminishing the related organisational problems.

1.4 Stochastic process algebras

The main motivation behind the development of stochastic process algebras has been — as already mentioned — to accurately describe and investigate the behaviour of resource-sharing systems (in contrast to timed process algebras for real-time systems). To achieve this goal, temporal information has been attached to process descriptions in the form of continuous time random variables. These random variables allow to represent time instants as well as durations of activities.

The concept of stochastic process algebras follows the lines of classical process algebras: the main ingredients are a formal mapping from system description to a semantic model and substitutive notions of equivalence. Equational laws reflect these equivalences on the system description level. Rather than considering only the functional behaviour we add stochastic timing information. This additional information in the semantic model allows the evaluation of various system aspects:

- functional behaviour (e.g. liveness or deadlocks)
- temporal behaviour (e.g. throughput, waiting times, reliability)
- combined properties (e.g. probability of timeout, duration of certain event sequences)

The stochastic process associated with every specification is the source for the derivation of performance results. Its characteristics clearly depend on the class of random distributions that are incorporated in the system description. Several attempts have been made to incorporate generally distributed random variables in the model. However, the general approach suffers from the following problem: general distributions lead to intractable stochastic processes and it is often impossible to efficiently analyse them. This problem disappears, if only a certain class of random distributions, so called (negative) exponential distributions, is allowed.

Models with exponential distributions are the basis of contemporary performance evaluation methodologies. They allow an accurate description of many real situations in resource-sharing systems. Moreover, it becomes straightforward to derive a continuous-time Markov chain (CTMC) out of a given specification. These performance models have been extensively studied in the literature and various efficient evaluation strategies exist, see e.g. [97].

1.5 Organisation of the paper

This paper surveys the theoretical developments in the field of stochastic process algebras. Section 2 presents the genealogy of stochastic process algebras, and relates their development to work in probabilistic and timed process algebras. In the subsequent 4 sections of the paper, 4 different calculi are introduced that show how random (exponential) delays can be incorporated into a process algebra. For each calculus an operational semantics, equivalences and axiomatisations are presented. Section 4 presents a simple action-less algebra for describing continuous-time Markov chains. An extension of this algebra with actions is presented in Section 5; this calculus can be considered as the core of languages like TIPP [39], PEPA [63] and EMPA [11]. The incorporation of time-less actions is considered in Section 6. In the latter two approaches delays and actions are treated as a single entity. The separation of exponential delays and actions is treated in Section 7. Obtaining performance models from process algebraic specifications is discussed in Section 8. Section 9 shows how the idea of separating random delays and actions can be used effectively to support arbitrary probability distributions. Finally, Section 10 summarises the paper, provides a taxonomy of the presented calculi and presents current trends and future challenges.

2 Genealogy

Traditionally, process algebras have concentrated on the functional aspects of systems such as their observable behaviour, control flow and synchronisation as properties in relative time. In the late eighties the interest grew in extending process algebras with quantitative information like time and (discrete) probabilities. These extensions are know as *timed* and *probabilistic* process algebras, respectively, and can be considered as the logical predecessors of stochastic process algebras where in fact time and probability are integrated by considering delays of a continuous probabilistic nature. Research on all these quantitative extensions of process algebras started around the same time.

2.1 Timed process algebras

Timed extensions of process algebras have received considerable attention in the last decade. To mention a few, extensions of languages like ACP [5], CSP [92], CCS [80,104] and LOTOS [15,75] have been defined. The main idea of these calculi is to extend the language with a timed prefix like $(t) \, . \, P$ which denotes that process P is reached after a delay of t time units. Both discrete-time and real-time variants are considered, depending on the domain of t. A main distinction between the several timed process algebras concerns the interpretation of when actions can occur. In a may-timing interpretation an action may occur after a certain time delay, but may be subject to a further delay, for instance, since it has to synchronise with its context. In a must-timing semantics, though, an action must occur as soon as it is enabled. The latter interpretation is usually applied to internal actions, since they are not subject to interaction, and thus there is no reason to delay them after becoming enabled. This interpretation is known as maximal progress. There are also more liberal approaches with respect to must-timing, where for instance, arbitrary actions can be made subject to must-timing using specific syntactical constructs (like timeout-operators or urgency operators [15]). An overview of the main issues in defining a timed process algebra can be found in [82]. Usually, the underlying semantical model is a form of timed transition system, where either action-transitions and time-advancing are combined or are treated separately. A more recent trend is to use timed automata [4] as semantical model, see e.g. [24].

Probabilistic extensions of the main process algebras have also been extensively considered in the last decade. Probabilistic extensions of ACP [6], CCS [43], CSP [76] and LOTOS [79] have been published, amongst others. For an overview of probabilistic process algebras we refer to [42]. The basic idea of probabilistic process algebras is to incorporate a probabilistic choice operator that allows terms like $P +_p Q$ (with $p \in (0, 1)$) where P can be selected with probability p and Q with $1-p$. Different semantical models are used for probabilistic process calculi, depending on whether they allow non-determinism — either occurring explicitly as operator, or as a consequence of interpreting parallel composition by interleaving — or not. Avoiding non-determinism is popular for probabilistic extensions of SCCS, Milner's synchronous variant of CCS, where probabilistic choice replaces non-deterministic choice [35]. The pleasant advantage of using a synchronous context for a probabilistic process algebra is the clear interpretation of parallel composition: since processes evolve in "lock-step" fashion the probability of a transition in the composition is just the product of the individual probabilities. For these calculi, the underlying semantical model is in fact a *discrete-time Markov chain*, where transitions are additionally equipped with actions. Larsen and Skou have defined an appropriate notion of (strong) bisimulation for these models [74], while weak bisimulation has been covered in [7], although the latter is not a congruence with respect to the standard parallel composition.

In the non-synchronous case, parallel composition becomes less trivial, and several proposals have been made. For an overview and comparison for the generative case we refer to [28]. Basically there are two approaches: either the non-determinism introduced by interleaving is resolved by parameterising the parallel composition operator with appropriate information or it occurs in the resulting semantical model and is subsequently resolved by a scheduler or adversary [103]. When, in addition to probabilistic branching non-deterministic branching may occur in the semantical model, so-called *Markov decision processes* [30] are obtained. Probabilistic transition systems of Segala and Lynch are rather similar to such decision processes and have been equipped with a notion of weak and strong bisimulation [94].

The use of a probabilistic, discrete-time synchronous process algebra WSCCS for performance evaluation has been proposed by Tofts [100,101]. Other approaches where probabilities and time are combined are, for instance, TPCCS of Hansson and Jonsson [43,42] and the (non-interleaving) extension of LOTOS proposed by Brinksma et al. [17].

The idea to define process algebras where actions may be delayed by time periods of a stochastic nature goes back to the work of Nounou and Yemini [83,84]. They proposed to delay actions by means of exponential distributions, but did not provide a formal semantics of their ideas. In the early nineties, Herzog [58] proposed a predecessor of the stochastic process algebra TIPP (**TI**med **P**rocesses and **P**erformance analysis), an extension of CSP, which was subsequently thoroughly defined in [39,38,54]. For an overview of TIPP we refer to [50]. Independently of this work, Hillston extended a CSP-like language with stochastic delays [62] which resulted in the language PEPA (**P**erformance **E**valuation **P**rocess **A**lgebra), defined in [63]. In 1993, Hillston initiated the first of a series of annual workshops called "Process Algebra for Performance Modelling" (PAPM). Other proposals that evolved shortly thereafter are the process algebras of Buchholz [20] and Bernardo et al. [10]. The latter evolved into EMPA (**E**xtended **M**arkovian **P**rocess **A**lgebra) where apart from actions that can be delayed by exponential distributions — like in TIPP and PEPA — immediate actions, non-determinism, priorities and weights are incorporated. For an overview of EMPA we refer to [11]. A stochastic variant of the π-calculus that is strongly influenced by PEPA has been developed by Priami [86]. These approaches have in common that the underlying semantical model is closely related to continuous-time Markov chains (extended with action-labels). An interesting model, for which to our knowledge no process algebra exists, is the probabilistic I/O-automata model of Wu et al. [106], where sojourn times are exponential, and branching is governed by discrete probability distributions.

As opposed to classical queueing systems and stochastic variants of Petri nets [3], stochastic process algebras like TIPP, PEPA and EMPA allow the specification and generation of complex Markov chains in a *compositional* way. The advantages of stochastic process algebras have been illustrated by several results. It has been proven by Hillston [63] and Buchholz [20] that strong Markovian bisimulation — an adaptation of Larsen and Skou's probabilistic bisimulation [74] — coincides with (ordinary) lumpability, an elementary notion for the aggregation of Markov chains. This result facilitates the adaptation of efficient algorithms for computing strong bisimulation to Markov chains [47]; for the first time a constructive and efficient way of computing lumpability has become available! In addition, the practicability of the algebraic approach to performance analysis has been illustrated by several examples, see e.g. [12,34,51,60], and important advances have been made in exploiting the structure of the compositionality for analysis purposes, for an overview see [64]. Several algorithms have been implemented in tools, like the TIPPtool [48], PEPA Workbench [33] and TwoTowers (for EMPA) [13].

For LOTOS preliminary proposals for stochastic extensions were presented by

Rico and von Bochmann [90] using semi-Markov chains, and by Valderrutten et al. [102] who derived queueing networks from extended LOTOS specifications. A similar, informal approach, has been proposed by Schot [93]. Ajmone Marsan et al. [1] defined a stochastic extension of LOTOS that allows more general distributions, but only allows the specification of stochastic delays at "top level", thus reducing compositionality significantly.

Although TIPP originally was intended to deal with arbitrary distributions [39], its theory has been developed as a stochastic process algebra supporting exponential distributions, like PEPA and EMPA. These calculi are also referred to as *Markovian* process algebras. The main problem with the incorporation of arbitrary distributions is the absence of the memoryless property. Brinksma et al. [18] proposed to use a non-interleaving semantics (using event structures) to deal with more general distributions. The derivation of discrete-event simulation models like GSMPs (Generalised Semi-Markov Processes [95]) from these extended event structures was presented in Katoen et al. [70]. Other non-interleaving approaches are the variant of the stochastic π-calculus of Priami [87] based on decorated transition systems, and the (informal) approach based on stochastic task graphs — structures for which efficient numerical and approximative analysis methods do exist — by Herzog [59]. Harrison and Strulo [45,99] used an interleaving semantics that, however, results in highly infinite semantical structures (since they map directly onto a kind of Borel space). Recently, Kanani et al. [32,68] developed a prototype tool for this approach and carried out several case studies. Using a stochastic variant of timed automata [4], D'Argenio et al. [25–27] defined the non-Markovian calculus $\spadesuit$ (which stands for SPADES, **S**tochastic **P**rocess **A**lgebra for **D**iscrete-**E**vent **S**imulation). They showed that GSMPs are a subset of stochastic automata [27] and reported on a prototype tool [26]. A non-Markovian variant of EMPA, referred to as GSMPA (**G**eneralised **S**emi-**M**arkovian **P**rocess **A**lgebra), is based on a combined form of step semantics and ST-semantics [16].

An overview of the activities in the field of stochastic process algebras can be obtained by consulting the proceedings of PAPM '93 – '98. [4]

[4] See `www.dcs.ed.ac.uk/PAPM/` and `fmt.cs.utwente.nl/~papm/`.

3 Preliminaries

3.1 Exponential distributions

Definition 1 *(Exponential distribution.) A probability distribution function F, defined by $F(t) = 1-e^{-\lambda t}$ for $t \geqslant 0$ and $F(t) = 0$ for $t < 0$ is an exponential distribution with rate $\lambda \in \mathbb{R}^+$. A random variable Φ is exponentially distributed if $\mathsf{Prob}(\Phi \leqslant t)$ is an exponential distribution.*

Evidently, a rate uniquely characterises an exponential distribution. The rate is the reciprocal of the mean value of an exponentially distributed random variable.

Definition 2 *(Memory-less property.) Let Φ be an exponentially distributed random variable and t, t' be positive reals. Φ possesses the* memory-less property *iff $\mathsf{Prob}(\Phi \leqslant t+t' \mid \Phi > t) = \mathsf{Prob}(\Phi \leqslant t')$.*

In fact, the exponential distribution is the only continuous probability distribution function that possesses this property. Other relevant properties of exponential distributions for this paper are closure properties of exponentially distributed random variables with respect to maximum and minimum. From basic probability theory [73] it is known that the distribution of the maximum of two independent random variables has a distribution that equals the product of their distributions. However, exponential distributions are not closed under product; the product is a phase-type distribution [22,81].

For the minimum there are, however, no such problems. Let Φ_i be pairwise independent, exponentially distributed random variables with rate λ_i for $0 < i \leqslant n$ and $n \geqslant 1$.

Lemma 3 *The random variable $min_{0<i\leqslant n}\, \Phi_i$ is exponentially distributed with rate $\sum\limits_{i=1}^{n} \lambda_i$.*

In addition, the probability that Φ_j (for some j) finishes first is given by a constant:

Lemma 4 $\mathsf{Prob}(\Phi_j < min_{0<i\leqslant n}\, \Phi_i) = \dfrac{\lambda_j}{\sum_{i=1}^{n} \lambda_i}$.

For convenience, we will use the terms maximum and minimum of distributions to denote maximum and minimum of their respective random variables.

A (discrete space) stochastic process is a collection of random variables $\{\, \Phi(t) \mid t \in T \,\}$ where $\Phi(t)$ assigns probabilities to elements of a discrete set S of states, the state space. If the set T (usually called the *time range*) is a continuous domain, the stochastic process is referred to as continuous-time. A continuous-time Markov chain (CTMC) is a continuous-time stochastic process that satisfies the Markov property: for each sequence of time instances $t_{n+1} > t_n > t_{n-1} > \ldots > t_0$ (of arbitrary length n), we have

$$\begin{aligned} &\mathsf{Prob}\{\, \Phi(t_{n+1}) = s_{n+1} \mid \Phi(t_n) = s_n, \Phi(t_{n-1}) = s_{n-1}, \ldots, \Phi(t_0) = s_0 \,\} \\ = &\mathsf{Prob}\{\, \Phi(t_{n+1}) = s_{n+1} \mid \Phi(t_n) = s_n \,\}. \end{aligned}$$

Thus, the fact that the chain was in state s_{n-1} at time t_{n-1}, in state s_{n-2} at time t_{n-2}, and so on, up to the fact that it was in state s_0 at time t_0 is irrelevant: the probability distribution on states at time t_n, given by $\Phi(t_n)$, contains all relevant history information to determine the distribution on S at time t_{n+1}.

Throughout this paper we consider *time-homogeneous* CTMCs. These chains are invariant under time shifts, i.e. for $\Delta > 0$ we have

$$\begin{aligned} &\mathsf{Prob}\{\, \Phi(t_{n+1}-\Delta) = s_{n+1} \mid \Phi(t_n-\Delta) = s_n \,\} \\ = &\mathsf{Prob}\{\, \Phi(t_{n+1}) = s_{n+1} \mid \Phi(t_n) = s_n \,\}. \end{aligned}$$

For each state of a CTMC there is some rate λ representing the distribution of the sojourn time for this state, which, in fact, turns out to be an exponential distribution. Furthermore, a CTMC is completely characterised by its generator matrix $\mathbf{Q}$ and its initial distribution. The entries of the generator matrix $\mathbf{Q}$ specify the transition rates: $\mathbf{Q}(i,j)$ denotes the rate of moving from state s_i to state s_j, where $i{\neq}j$. The initial distribution specifies the probability of starting in a certain state. More precisely,

Definition 5 *(Generator matrix) A square matrix* $\mathbf{Q}$ *is the (infinitesimal) generator matrix of a CTMC iff, for all* i, $\mathbf{Q}(i,j) \geqslant 0$ $(j \neq i)$, *and* $\mathbf{Q}(i,i) = -\sum_{j \neq i} \mathbf{Q}(i,j)$.

The states of a CTMC can be classified into recurrent, transient and absorbing states. A state s_i is said to be transient if there is a positive probability of never returning to that state after leaving it. A recurrent state is revisited (in a finite amount of time) with probability 1. An absorbing state is a state without any outgoing transition, i.e. $\mathbf{Q}(i,j) = 0$, for all j.

Lumpability is an important notion on CTMCs that allows their aggregation without affecting performance properties [71,19].

Definition 6 *(Ordinary lumpability.) For $S = \{ S_1, \ldots, S_N \}$ a partitioning of the state space of a CTMC, the CTMC is* ordinary lumpable *with respect to S if and only if for any partition S_I and states $s_i, s_j \in S_I$:*

$$\forall 0 < K \leqslant N. \sum_{k \in S_K} \mathbf{Q}(i,k) = \sum_{k \in S_K} \mathbf{Q}(j,k).$$

That is, for any two states in a given partition the cumulative rate of moving to any other partition should be equal. The performance measures of a CTMC and its lumped counterpart are strongly related. The (macro-)probability of the lumped CTMC being in state S_I equals $\sum_{i \in S_I} \pi(i)$, where $\pi(i)$ denotes the probability of being in state s_i. This correspondence holds for transient and stationary probabilities.

4 An algebra for CTMCs

4.1 Syntax

Let X be drawn from a set of process variables, and I drawn from a set of finite sets of indices. Furthermore let $\lambda_i \in \mathbb{R}^+$ for $i \in I$. The syntax of the algebra MC is

$$P ::= \sum_{i \in I} (\lambda_i) \,.\, P \;\Big|\; X \;\Big|\; recX.P$$

The term $recX.P$ defines a recursive process X by P, that possibly contains occurrences of X. A variable X is called bound in an expression Q, if each occurrence of X lies inside a (binding) subexpression $recX.P$ of Q, and free otherwise. For $I = \varnothing$ let $\sum_{i \in I} (\lambda_i) \,.\, P_i = \mathbf{0}$, the process that cannot perform any action. For I a singleton set, the term $(\lambda) \,.\, P$ denotes a process that evolves into P within t time units ($t \geqslant 0$) with probability $1 - e^{-\lambda t}$. That is, it behaves like P after a certain delay that is determined by a (continuous) random variable, Φ say, such that $\mathsf{Prob}(\Phi \leqslant t)$ equals $1 - e^{-\lambda t}$ for positive t, and equals zero otherwise. The prefix $(\lambda) \,.\, P$ can be considered as the probabilistic version of the deterministic timed prefix $(t) \,.\, P$ that typically occurs in timed process algebras, like in TCCS [80] or in Timed CSP [92]. In general, the term $\sum_{i \in I} (\lambda_i) \,.\, P_i$ offers a probabilistic choice among the processes P_i. The precise meaning of this construct is defined below. Notation: if I consists of two elements we use binary choice, denoted $+$.

4.2 Semantics

The structured operational semantics of the algebra MC is presented in Table 1. The inference rules define a mapping of this algebra onto CTMCs (as we will see). The rule for recursion is standard; we just recall that $P\{\, Q/X \,\}$ denotes term P in which all free occurrences of process variable X in P are replaced by Q.

The rule for choice requires some explanation. Consider $\sum_{i \in I} (\lambda_i) \,.\, P_i$. At execution, the fastest process, that is, the process that is enabled first, is selected. This is known as the race condition. The time until selecting the fastest process among a set of time-prefixed processes is distributed according to the *minimum* of the exponential distributions involved. That is, the delay until the resolution of the choice is exponentially distributed with rate $\sum_{i \in I} \lambda_i$ (cf. Lemma 3). The probability of choosing a particular alternative, P_j say ($j \in I$), equals $\frac{\lambda_j}{\sum_{i \in I} \lambda_i}$ (cf. Lemma 4), assuming that summands with distinct indices are distinct.

This explanation also justifies the extension of transitions with an auxiliary label indicated as a subscript of the transition relation that is used to distinguish between different deduction trees of a term. In absence of such mechanism, we would, for instance, for $(\lambda_1) \,.\, P + (\lambda_2) \,.\, P$, obtain two distinct transitions, except if $\lambda_1 = \lambda_2$. In that specific case we would obtain two different deduction trees for the same transition labelled λ_1 (or λ_2); this, however, does suggest that P can be reached with rate λ_1 (or λ_2), whereas this should be — according to the above explanation — rate $\lambda_1 + \lambda_2$. A similar mechanism is standard in probabilistic process calculi like PCCS [35].

Table 1
Operational semantics for MC

$$\sum_{i \in I} (\lambda_i) \,.\, P_i \xmapsto{\lambda_j}_j P_j \quad (j \in I) \qquad\qquad \frac{P\{\, recX.P/X \,\} \xmapsto{\lambda}_i P'}{recX.P \xmapsto{\lambda}_i P'}$$

The operational semantics of Table 1 maps a term onto a transition system where transitions are labelled by rates. It is not difficult to check that by omitting self-loops and replacing the set of transitions from s to s' by a single transition with the sum of the rates of the transitions from s to s', a CTMC is obtained.

Example 7 *In order to illustrate the language and its semantics we develop a simple example. For convenience, we shall use recursive defining equations that are not part of the language* MC. *The encoding of recursive defining equations into instances of recursion is routine, and left to the reader.*

Consider the following term X_0 of MC, *defined by:*

$$
\begin{aligned}
X_0 &:= (\lambda) \,.\, X_1 \\
X_1 &:= (\mu) \,.\, X_0 + (\lambda) \,.\, X_2 \\
X_2 &:= (\mu) \,.\, X_1 + (\mu) \,.\, ((\mu) \,.\, X_0 + (\lambda) \,.\, X_2)
\end{aligned}
$$

Applying the semantics leads to the transition system depicted in Fig. 4 (transition subscripts are omitted). The initial state is indicated by a bold circle.

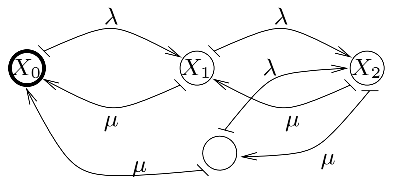

Fig. 4. Semantics of an example MC expression.

4.3 Lumping equivalence

Lumping equivalence is defined in the same style as Larsen & Skou's probabilistic bisimulation [74] and Hillston's strong equivalence [63]. Let $\{\!|\ \dots\ |\!\}$ denote multi-set brackets.

Definition 8 *(Lumping equivalence.) An equivalence relation $\mathcal{S}$ on* MC *is a* lumping equivalence *if and only if for any pair $(P, Q) \in \mathsf{MC} \times \mathsf{MC}$ we have that $(P, Q) \in \mathcal{S}$ implies for all equivalence classes $C \in \mathsf{MC}/_{\mathcal{S}}$:*

$$\gamma_m(P, C) = \gamma_m(Q, C) \text{ with } \gamma_m(R, C) = \sum_i \{\!|\ \lambda \mid R \overset{\lambda}{\longmapsto}_i R', R' \in C\ |\!\}.$$

Processes P and Q are lumping equivalent*, denoted $P \sim Q$, if $(P, Q) \in \mathcal{S}$ with $\mathcal{S}$ a lumping equivalence.*

Here, we use $\mathsf{MC}/_{\mathcal{S}}$ to denote the set of equivalence classes induced by $\mathcal{S}$ over MC. Stated in words, P and Q are lumping equivalent if the total rate of moving to equivalence class C under $\sim$ is identical for all such classes.

Example 9 *X_0 (cf. Example 7) is lumping equivalent to X_3, given by*

$$X_3 := (\lambda) \,.\, X_4$$
$$X_4 := (\mu) \,.\, X_3 + (\lambda) \,.\, (2\mu) \,.\, X_4$$

To illustrate this, let $\mathcal{S}$ be the equivalence relation containing (exactly) those pairs of states in Fig. 5 that are shaded with identical patterns. It is easy to see that $\mathcal{S}$ is a lumping equivalence containing (X_0, X_3). Thus, $X_0 \sim X_3$.

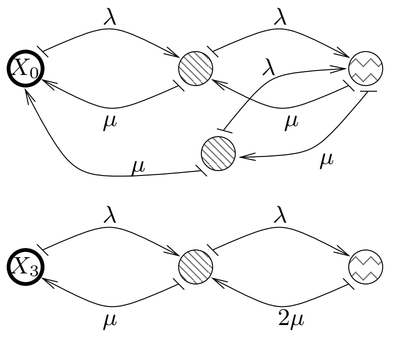

Fig. 5. Lumping equivalence classes.

Theorem 10 *$P \sim Q$ if and only if their underlying CTMCs can be partitioned into isomorphic ordinary lumpable partitionings.*

Theorem 11 *$\sim$ is a congruence with respect to the operators of* MC.

Due to the fact that $\sim$ is a congruence, lumping can be performed in a compositional way, i.e. component-wise. Thus, the compositional structure of an algebraic system specification can be exploited in order to generate and reduce the underlying CTMC in a modular way, component by component.

4.4 Equational theory

In this section we present a sound and complete axiomatisation of $\sim$ for MC. Such an axiomatisation facilitates the lumping of CTMCs at a syntactical level, i.e., without constructing the underlying CTMC at all. The axioms for sequential finite terms are listed as (B1) through (B4) in Table 2. Note that summations are always finite in our case, since we consider finite index sets. The axioms (B1) through (B3) are well known from classical process calculi. Axiom (B4) is a distinguishing law for our calculus and can be regarded as a replacement in the Markovian setting of the traditional idempotence axiom for choice ($P + P = P$). Axiom (B4) reflects that the resolution of choice is modelled by the minimum of (statistically independent) exponential distributions.

Axioms (R1) through (R3) constitute the laws for recursion. These are identical to those for classical process calculi and are supposed to be self-explanatory. We just recall that a process variable X is *guarded* in P if each occurrence of X is within some sub-term of P of the form $(\lambda) . Q$. A term P is called guarded if, for every sub-term of the form $recX.Q$, the process variable X is guarded in Q.

Table 2
Axioms for finite sequential processes

(B1)	$P + \mathbf{0}$	$=$	P	
(B2)	$P + Q$	$=$	$Q + P$	
(B3)	$(P + Q) + R$	$=$	$P + (Q + R)$	
(B4)	$(\lambda) . P + (\mu) . P$	$=$	$(\lambda + \mu) . P$	
(R1)	$recX.P$	$=$	$recY.(P\{ Y/X \})$	Y is not free in $recX.P$
(R2)	$recX.P$	$=$	$P\{ recX.P/X \}$	
(R3)	$Q = P\{ Q/X \}$	$\Rightarrow$	$Q = recX.P$	X is guarded in P

The next result shows that the axioms (B1) through (B4) together with (R1) through (R3) form a sound and complete axiomatisation of lumping equivalence for guarded terms in MC. Let $\mathcal{A}$ denote the axiom system that consists of the axioms in Table 2 and let MC_G denote the set of guarded terms in MC.

Theorem 12 *For any $P, Q \in \mathsf{MC}_G$, $\mathcal{A} \vdash (P = Q)$ if and only if $P \sim Q$.*

Here, $\mathcal{A} \vdash (P = Q)$ means that the equality $P = Q$ can be deduced from the axioms in $\mathcal{A}$.

4.5 Parallel composition

We add a simple parallel composition operator to our calculus, denoted by $||$. Intuitively, the term $P \,||\, Q$ can evolve while either P evolves or Q evolves independently from each other. Parallel composed processes can delay completely independently:

$$\boxed{\dfrac{P \overset{\lambda}{\longmapsto}_i P'}{\begin{array}{c} P \,||\, Q \overset{\lambda}{\longmapsto}_{(i,*)} P' \,||\, Q \\ Q \,||\, P \overset{\lambda}{\longmapsto}_{(*,i)} Q \,||\, P' \end{array}}}$$

(Notice that we create new auxiliary labels of the form $(i, *)$ and $(*, i)$ in order to obtain a multi-transition relation.) This is different from a deterministic time setting where parallel processes typically are forced to synchronise on the advance of time, as in TCCS [80]. The justification for independent delaying relies on the memory-less property of exponential distributions. To understand the meaning of this property in our context consider the process $(\lambda) \, . \, P \,||\, (\mu) \, . \, Q$ and suppose that the delay of the left process finishes first (with rate λ). Due to the memory-less property, the remaining duration until an initial delay of Q may occur is determined by an exponential distribution with rate μ, exactly the delay prior to the enabling of these delays before the delay of the first process has finished. Stated differently, the delay of the left process does not have any impact on the remaining delay in the other process — the advance of time governed by memory-less distributions is independent. One of the consequences of this independent delaying is that an expansion law is obtained rather straightforwardly. This is not the case for a deterministic time setting [37]. This expansion law is listed in Table 3. Notice that a CTMC can be obtained in the same way as before: by omitting self-loops and collapsing multi-transitions while adding up rates.

Table 3
Expansion law

for $P = \sum_i (\lambda_i) \, . \, P_i$ and $Q = \sum_j (\mu_j) \, . \, Q_j$ we have		
(P1) $P \,||\, Q$	$=$	$\sum_i (\lambda_i) \, . \, (P_i \,||\, Q) + \sum_j (\mu_j) \, . \, (P \,||\, Q_j)$

Theorem 13 $\sim$ *is a congruence with respect to parallel composition.*

Let $\mathcal{A}^*$ denote the axiom system $\mathcal{A}$ (cf. Table 2) extended with the expansion law (P1) and let MC_R denote the set of regular terms in MC where parallel composition is added. A term P is called *regular* if it is guarded and if any sub-term of P of the form $recX.Q$ does not contain an occurrence of $||$ in Q. The restriction to regular terms is as in classical process calculi in order to obtain completeness.

Theorem 14 *For any* $P, Q \in \mathsf{MC}_R$, $\mathcal{A}^* \vdash (P = Q)$ *if and only if* $P \sim Q$.

5 Pure Markovian process algebra

The algebra MC of the previous section provides a restricted way of specifying CTMCs in a compositional way. For instance, synchronisation between different processes cannot be described at all. The idea that we pursue in this section is to integrate the notion of actions in the classical process algebraic sense with random delays governed by exponential distributions. This combination of processes and CTMCs has been brought up independently in 1993 by Götz, Herzog and Rettelbach [39] and by Hillston [62]. Notions of equivalence and axiomatisations are studied in [54] for TIPP, [63] for PEPA and [11] for EMPA.

5.1 Syntax

Let X be an element of a set of process variables, I drawn from a set of finite sets of indices and $\lambda_i \in \mathbb{R}^+$ for $i \in I$. Let **Obs** be a denumerable set of actions and let $\mathbf{A} = \mathbf{Obs} \cup \{ \tau \}$ be the set of actions ranged over by $a, b, c, \ldots$ (possibly subscripted with some index from I), where τ is a distinguishing action that models some internal activity. Furthermore, let $A \subseteq \mathbf{Obs}$ and $f : \mathbf{A} \to \mathbf{A}$ such that $f(\tau) = \tau$. The syntax of the algebra MAC is now given by

$$P ::= \sum_{i \in I} (a_i, \lambda_i) \,.\, P \;\Big|\; P \,||_A\, P \;\Big|\; P[f] \;\Big|\; X \;\Big|\; recX.P$$

The prefix $(a, \lambda) \,.\, P$ denotes that action a is offered *after* a delay determined by an exponential distribution with rate λ. Similar as before, the term $\sum_{i \in I} (a_i, \lambda_i) \,.\, P$ selects the fastest enabled alternative. For instance, the process $(a, \lambda) \,.\, \mathbf{0} + (b, \mu) \,.\, \mathbf{0}$ is able to offer a with probability $\frac{\lambda}{\lambda+\mu}$ and action b with probability $\frac{\mu}{\lambda+\mu}$ provided actions a and b are not blocked by the environment. Process $P \,||_A\, Q$ denotes the parallel composition of P and Q where synchronisation is required on all actions in A, while actions not in A can be performed autonomously by P and Q. Relabelling is defined as usual. Abstraction is defined using relabelling: $P \setminus A = P[f]$ where $f(a) = \tau$ for all $a \in A$ and $f(a) = a$ for all $a \notin A$. (Abstraction should not be confused with CCS-restriction.)

5.2 Semantics

The basic difference between the stochastic process algebras TIPP [39], PEPA [63] and EMPA [11] is the calculation of the resulting rate in case of synchronisation. TIPP proposes the product of rates, EMPA forbids this type of synchronisation and requires one component to determine the rate only while the other components need to be passive (i.e. willing to accept any rate), and finally PEPA computes the maximum of mean delays while incorporating the individual synchronisation capacities of processes. For the purpose of the following discussion we do not want to stick to a particular approach. We rather assume that synchronisation leads to some exponential distribution (in order to have CTMCs as underlying model). Formally, we determine the resulting rate of a synchronisation of two actions with rates λ and μ say, by $\lambda \odot \mu$ where $\odot : \mathbb{R}^+ \times \mathbb{R}^+ \rightarrow \mathbb{R}^+$.[5]

Table 4
Operational semantics for MAC

$$\sum_{i \in I} (a_i, \lambda_i) \, . \, P_i \xrightarrow{a_j, \lambda_j}_j P_j \quad (j \in I) \qquad \frac{P\{\, recX.P/X \,\} \xrightarrow{a,\lambda}_i P'}{recX.P \xrightarrow{a,\lambda}_i P'}$$

$$\frac{P \xrightarrow{a,\lambda}_i P'}{\begin{array}{l} P \,||_A\, Q \xrightarrow{a,\lambda}_{(i,*)} P' \,||_A\, Q \\ Q \,||_A\, P \xrightarrow{a,\lambda}_{(*,i)} Q \,||_A\, P' \end{array}} \quad (a \notin A) \qquad \frac{P \xrightarrow{a,\lambda}_i P', Q \xrightarrow{a,\mu}_j Q'}{P \,||_A\, Q \xrightarrow{a,\lambda \odot \mu}_{(i,j)} P' \,||_A\, Q'} \quad (a \in A)$$

$$\frac{P \xrightarrow{a,\lambda}_i P'}{P[f] \xrightarrow{f(a),\lambda}_i P'[f]}$$

The operational semantics of Table 4 maps a term P onto a transition system where transitions are labelled with action/rate-pairs. As before we equip the transition relation with an auxiliary label in order to distinguish between different deduction trees. Similar as in the previous section, a CTMC is obtained (after the removal of actions in the labels), if cycles are eliminated and subsequently multi-transitions are collapsed while adding up rates.

5.3 Strong Markovian bisimulation

Definition 15 *(Strong Markovian bisimulation.) An equivalence relation $\mathcal{S}$ on* MAC *is a* strong Markovian bisimulation *if and only if for any pair $(P, Q) \in \mathsf{MAC} \times \mathsf{MAC}$ we have that $(P, Q) \in \mathcal{S}$ implies for all actions a and all equivalence classes $C \in \mathsf{MAC}/_{\mathcal{S}}$:*

$$\gamma'_m(P, a, C) = \gamma'_m(Q, a, C) \text{ with } \gamma'_m(R, a, C) = \sum_i \{\!|\, \lambda \mid R \xrightarrow{a,\lambda}_i R', R' \in C \,|\!\}.$$

Processes P and Q are strongly Markovian bisimilar*, denoted $P \sim_m Q$, if $(P, Q) \in \mathcal{S}$ with $\mathcal{S}$ a strong Markovian bisimulation.*

[5] In order to take the PEPA-approach into account $\odot$ should be a function $\mathbb{R}^+ \times \mathbb{R}^+ \times \mathsf{MAC} \times \mathsf{MAC} \rightarrow \mathbb{R}^+$ such that $\odot(\lambda, \mu, P, Q)$ takes the synchronisation capacities of processes P and Q into account. For the sake of simplicity we omit the dependencies on the processes involved.

Intuitively, two processes are strongly Markovian bisimilar if they are strongly bisimilar in the classical process algebraic sense, and if the cumulated rates of moving by an a-transition (for any a) to each equivalence class are equal. The relation with lumping equivalence (cf. Definition 8) is as follows. For $P \in \mathsf{MAC}$ let $\overline{P}$ denote P where all action labels are removed from P. (This function can be easily defined by structural induction and is omitted here.) Then,

Lemma 16 *For all $P, Q \in \mathsf{MAC}$, $P \sim_m Q \Rightarrow \overline{P} \sim \overline{Q}$.*

Notice that the reverse implication is not valid, since transitions can be labelled with different action labels. Similarly, two processes are strongly bisimilar (in the classical sense) after removing rate labels, if they are Markovian bisimilar.

If the combination of rates in case of synchronisation fulfils certain algebraic properties, then $\sim_m$ is a congruence:

Theorem 17 *For a commutative ring $\langle \mathbb{R}^+, +, \odot \rangle$, $\sim_m$ is a congruence with respect to all operators in MAC.*

Distributivity of $\odot$ over $+$ is a prerequisite for congruence, while commutativity and associativity of $\odot$ imply the respective properties of $||$. In the sequel we will assume that $\odot$ satisfies the aforementioned algebraic properties.

Like for strong bisimulation, $\sim_m$ treats internal actions like any other action. One might wonder whether a notion like weak bisimulation — that abstracts from internal transitions — can be adapted to the calculus MAC. The basic idea of such a relation would facilitate the replacement of several consecutive τ-actions by a single τ-action with a duration that equals the end-to-end duration of the sequence of τ's it replaces. Such an equivalence notion is, however, not possible. Although action names become invisible to the environment, the timing behaviour remains visible. For example, a term like $(a, \lambda) \,.\, (\tau, \mu) \,.\, \mathbf{0}$ is not equivalent to $(a, \nu) \,.\, \mathbf{0}$ for any rate ν, since the sequence (i.e. convolution) of two exponentially distributed phases is not exponential but constitutes a phase-type distribution [22,81]. A possible solution is to relax the requirement of obtaining equal distributions for equivalent processes, e.g. by only requiring equal mean durations like in Hillston's weak isomorphism [63]. This notion is a congruence for prefixing, relabelling and parallel composition, not for choice.

5.4 Equational theory

Table 5 lists some axioms for MAC (for $\sim_m$). Axiom (B4′) is the equivalent of the distinguishing axiom (B4) in an action-based setting. The axioms for relabelling are standard for classical process calculi, whereas the expansion law is a straightforward adaptation of the traditional expansion law. Recall

Table 5
Axioms for MAC

(B4′)	$(a,\lambda) \,.\, P + (a,\mu) \,.\, P$	$=$	$(a,\lambda{+}\mu) \,.\, P$
(RL1)	$\mathbf{0}[f]$	$=$	$\mathbf{0}$
(RL2)	$((a,\lambda) \,.\, P)[f]$	$=$	$(f(a),\lambda) \,.\, P[f]$
(RL3)	$(P+Q)[f]$	$=$	$P[f]+Q[f]$

for $P = \sum_i (a_i,\lambda_i) \,.\, P_i$ and $Q = \sum_j (b_j,\mu_j) \,.\, Q_j$ we have

(P1′)
$$\begin{aligned} P \,||_A\, Q = \quad & \sum_i \{ (a_i,\lambda_i) \,.\, (P_i \,||_A\, Q) \mid a_i \notin A \} \\ & + \sum_j \{ (b_j,\mu_j) \,.\, (P \,||_A\, Q_j) \mid b_j \notin A \} \\ & + \sum_{i,j} \{ (a_i,\lambda_i \odot \mu_j) \,.\, (P_i \,||_A\, Q_j) \mid a_i, b_j \in A, a_i = b_j \} \end{aligned}$$

that the validity of this rule is due to the memory-less property of exponential distributions. Let $\hat{\mathcal{A}}$ be the axiom system consisting of axioms (B1) through (B3) (cf. Table 2), (B4′), (R1) through (R3) (cf. Table 2), (RL1) through (RL3) and (P1′). This axiom system is sound and complete for regular terms in MAC. Here, P is called *regular* if it is guarded (in the sense we have seen before) and if any sub-term of P of the form $recX.Q$ contains neither an occurrence of $||$ nor an occurrence of $[f]$ in Q. Let MAC_R denote the set of regular terms in MAC.

Theorem 18 *For any $P, Q \in \mathsf{MAC}_R$, $\hat{\mathcal{A}} \vdash (P = Q)$ if and only if $P \sim_m Q$.*

5.5 Passive actions

To enable the specification of passive activities, like in client-server communications, that simply wait for an interaction partner, MAC could be extended with the notion of *passive* actions. A passive action has an unspecified rate (denoted $\mathbf{1}$). The idea is that a passive action, a say, intends to synchronise with an active action — an action with a specified rate λ say — such that the resulting synchronised action incurs a delay $\mathbf{1} \odot \lambda = \lambda$. That is, $\mathbf{1}$ is a (left and right) identity of $\odot$.[6]

Example 19 *A compositional specification of a simple producer/consumer*

[6] Here, we assume a commutative ring $\langle \mathsf{D}, +, \odot \rangle$ where $\mathbf{1}$ is the identity of $\odot$ and D is the smallest set containing $\mathbb{R}^+ \cup \{ \mathbf{1} \}$ that is closed under $+$ and $\odot$.

example might look as follows:

$$(recP.(p, \lambda) \,.\, (h, \mu) \,.\, P) \,||_h\, (recC.(h, \mathbf{1}) \,.\, (c, \nu) \,.\, C).$$

A producer P generates (p) an item with a certain rate λ and then hands, (h, μ), the item to the consumer C that passively waits for the item. After passing the item to the comsumer, the consumer consumes (c) the item with rate ν while a new item can be produced independently with rate λ. Fig. 6 depicts the transition system obtained by applying the operational semantics.

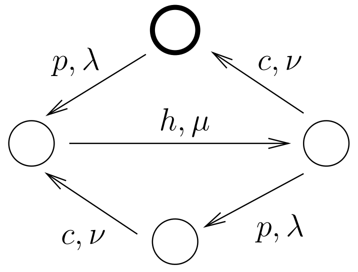

Fig. 6. Semantics of the simple producer/consumer system

Although from a specification point of view, passive actions are convenient, their treatment appears to introduce non-determinism, for instance, when multiple passive actions are enabled in a certain state. The existing approaches (like in TIPP, PEPA and EMPA) take a similar approach, though technically realised in different ways: choices between multiple passive actions are treated as equi-probable. That is, whenever multiple passive transitions may independently synchronise with an active transition, all synchronisations are possible, and the resulting rates are identical.

The technical differences between the three mentioned calculi are illustrated by considering the following term:

$$((a, \mathbf{1}) \,.\, \mathbf{0} + (a, \mathbf{1}) \,.\, \mathbf{0} + (a, \mathbf{1}) \,.\, (b, \lambda) \,.\, \mathbf{0}) \,||_a\, (a, \mu) \,.\, \mathbf{0}$$

where the leftmost process has three passive a-actions enabled initially, one that is followed by a b-action, and two that are not. In TIPP this term can be rewritten using an axiom like (B4′) into:

$$((a, \mathbf{1}{+}\mathbf{1}) \,.\, \mathbf{0} + (a, \mathbf{1}) \,.\, (b, \lambda) \,.\, \mathbf{0}) \,||_a\, (a, \mu) \,.\, \mathbf{0} \tag{1}$$

which after expansion reduces to:

$$(a, 2\mu) \,.\, \mathbf{0} + (a, \mu) \,.\, (b, \lambda) \,.\, \mathbf{0}.$$

As a consequence of Lemma 4, with probability $\frac{2}{3}$ no b action will be offered, and with probability $\frac{1}{3}$ a b-action will be offered. In PEPA we obtain the same branching probabilities, but with different delays: expansion after rewriting the original term into (1) results into:

$$(a, \frac{2}{3}\mu) \,.\, \mathbf{0} + (a, \frac{1}{3}\mu) \,.\, (b, \lambda) \,.\, \mathbf{0}.$$

Finally, in EMPA the original term can be rewritten into:

$$((a, \mathbf{1}) \,.\, \mathbf{0} + (a, \mathbf{1}) \,.\, (b, \lambda) \,.\, \mathbf{0}) \,||_a\, (a, \mu) \,.\, \mathbf{0}$$

by using the idempotency law for choice. Like for TIPP and PEPA by applying expansion, the choice between the remaining two passive actions is treated as equi-probable. This results in:

$$(a, \frac{1}{2}\mu) \,.\, \mathbf{0} + (a, \frac{1}{2}\mu) \,.\, (b, \lambda) \,.\, \mathbf{0}$$

Thus, in EMPA the probability of the occurrence of a b-action (after the occurrence of a) is $\frac{1}{2}$.

6 Immediate actions

In the previous calculus MAC all actions have a duration. In order to support temporal abstraction it would, however, be fruitful to abstract from durations and allow for the incorporation of actions without duration, so-called *immediate* actions. A similar concept appears in Generalised Stochastic Petri Nets (GSPNs [3]) where immediate transitions are an essential ingredient of the model. In this section we investigate such extension for MAC. The extension of stochastic process algebras with immediate actions and the notion of Markovian observational congruence has been brought up by Hermanns, Rettelbach and Weiss [56]. Immediate actions also appear in EMPA [11].

6.1 Syntax

We extend the process algebra MAC with immediate actions, actions that do not exhibit any delay once they are enabled. Let $\underline{\mathbf{A}}$ denote the set of immediate actions (ranged over by $\underline{a}, \underline{b}, \ldots$) such that $\mathbf{A} \cap \underline{\mathbf{A}} = \varnothing$. Let $\widehat{\mathbf{A}} = \mathbf{A} \cup \underline{\mathbf{A}}$. The basic idea is to extend the syntax with an immediate prefix, denoted by $\underline{a} \,.\, P$. The term $\underline{a} \,.\, P$ can perform $\underline{a}$ without any delay while evolving into P. The other operators in MAC remain unchanged, with the only difference that the synchronisation set $A \subseteq \widehat{\mathbf{A}}$ and that $f : \widehat{\mathbf{A}} \rightarrow \widehat{\mathbf{A}}$ with $f(\tau) = \tau$, $f(\underline{\tau}) = \underline{\tau}$ and $f(a) \in \mathbf{A}$ and $f(\underline{a}) \in \underline{A}$. The resulting calculus is denoted by IMAC.

6.2 Semantics

The operational semantics for IMAC is defined using two transition relations: $\xrightarrow{a,\lambda}$, which is defined using the rules in Table 4, and $\overset{a}{\hookrightarrow}$, for the execution of immediate actions, whose inference rules are identical to those of classical process calculi; for completeness, they are listed in Table 6. Note that IMAC

Table 6
Additional inference rules for IMAC

$$\sum_{i \in I} \underline{a_i} \,.\, P_i \overset{a_j}{\hookrightarrow} P_j \quad (j \in I) \qquad \frac{P\{\mathit{recX}.P/X\} \overset{a}{\hookrightarrow} P'}{\mathit{recX}.P \overset{a}{\hookrightarrow} P'} \qquad \frac{P \overset{a}{\hookrightarrow} P'}{P[f] \overset{f(a)}{\hookrightarrow} P'[f]}$$

$$\frac{P \overset{a}{\hookrightarrow} P'}{\begin{array}{c} P \,||_A\, Q \overset{a}{\hookrightarrow} P' \,||_A\, Q \\ Q \,||_A\, P \overset{a}{\hookrightarrow} Q \,||_A\, P' \end{array}} \quad (a \notin A) \qquad \frac{P \overset{a}{\hookrightarrow} P', Q \overset{a}{\hookrightarrow} Q'}{P \,||_A\, Q \overset{a}{\hookrightarrow} P' \,||_A\, Q'} \quad (a \in A)$$

incorporates both a probabilistic choice, if both arguments are time-prefixed terms, and a non-deterministic choice in the usual process-algebraic sense (e.g. in $\underline{a} \,.\, P + \underline{a} \,.\, Q$). In case of a competition between a time-prefixed and an action-prefixed process, the latter process will be selected if the offered action is not blocked by the environment (e.g. in case of an internal action). This is justified by the fact that the probability that an exponentially distributed duration finishes instantaneously, is zero. However, to achieve compositionality, the precedence of timed-prefixed over (non-blocked) action-prefixed processes is not reflected in the rules defining the transition relations $\xrightarrow{a,\lambda}$ and $\overset{a}{\hookrightarrow}$. Instead, we introduce below a notion of equality, referred to as weak Markovian bisimulation, that (among others) takes care of the interplay of time- and action-prefixes.

Due to the incorporation of immediate transitions, the generation of a CTMC is more involved than for MAC. Moreover, it is not guaranteed that a CTMC results anyway. In particular, in the presence of nondeterminism a CTMC can not be obtained in general. In absence of nondeterminism the procedure works briefly as follows. Let us distinguish between states with outgoing immediate transitions and states without. The former states will have a zero sojourn time. Consequently, such states are called vanishing states. In contrast, states without immediate outgoing transitions are referred to as tangible states. The model that results is a semi-Markov process. In order to solve such models it is preferred to eliminate vanishing states a priori. A frequently used technique to eliminate vanishing states is to introduce transitions that occur due to the traversal of some vanishing states between two tangible states until all vanishing states are bypassed [2]. The main disadvantage of this technique, however, is that it is not compositional if nondeterminism is involved. In the sequel we propose to eliminate (internal) immediate transitions by means of a congruence relation (cf. Section 8.1).

6.3 Markovian observational congruence

A notion of strong bisimulation can be defined for IMAC in the same sense as for MAC where in addition to the requirements listed in Definition 15 one requires that immediate actions are bisimilar in the classical sense.

More interesting, though, is to adopt a notion of weak bisimulation. As argued before, the probability distribution of the duration of a sequence of exponential distributions is not exponential, which makes it impossible to define weak bisimulation on timed actions of the form (a, λ). Instead, the delays of these actions will be treated in the same way as in Markovian bisimulation ($\sim_m$), but they may be preceded and followed by internal immediate actions.

The basic idea is to adopt weak bisimulation on immediate actions. This facilitates the elimination of internal immediate actions, an essential ingredient for obtaining a CTMC. The combination of Markovian bisimulation for delayed actions and weak bisimulation on immediate actions is referred to as *weak Markovian bisimulation* (notation $\approx_m$).

The definition of $\approx_m$ is obtained in the following way. Define $C^\tau = \{ P \mid P \overset{\tau^*}{\hookrightarrow} P', P' \in C \}$, for some set C of processes. In words, C^τ is the set of processes that can reach some process in C via a (possibly empty) sequence of immediate transitions. Furthermore let

$$\gamma_i(R, \underline{a}, C) = \begin{cases} 1 & \text{if } \{ R' \in C \mid R \overset{a}{\hookrightarrow} R' \} \neq \varnothing \\ 1 & \text{if } \underline{a} = \underline{\tau} \text{ and } R \in C \\ 0 & \text{otherwise} \end{cases}$$

As usual in weak bisimulation, a non-internal step must be simulated by a matching step, possibly preceded and/or followed by arbitrarily many internal steps. An internal step may be simulated in a similar way, but can also be mimicked by staying (i.e., taking no transition at all) provided the equivalence classes match. Technically, this is achieved by requiring that for each equivalence class C and processes P and Q, $\gamma_i(P, \underline{a}, C) = \gamma_i(Q', \underline{a}, C^\tau)$, for some Q',

$Q \overset{\tau^*}{\hookrightarrow} Q'$. Apart from using nonstandard notation, this requirement coincides with the usual formulation. We demand that timed transitions have to be bisimulated in the same sense, but with a slight exception. Since the probability that a continuously distributed duration finishes immediately (i.e. at time zero) is zero, whereas the probability for an internal immediate transition to take place immediately is one, there is no need to require equality of cumulated rates for states that have an outgoing immediate τ-transition. Stochastically speaking, these states have a zero sojourn time since they are immediately left using an internal (immediate) move. In process algebraic terms this stochastic observation boils down to the well-known *maximal progress* property: A process that has something internal to do will do it, without letting time pass. Let $P \overset{\tau}{\not\hookrightarrow}$ if and only if $\{ P' \mid P \overset{\tau}{\hookrightarrow} P' \} = \varnothing$.

Definition 20 *(Weak Markovian bisimulation.) An equivalence relation* $\mathcal{S}$ *on* IMAC *is a* weak Markovian bisimulation *if and only if for any pair* $(P, Q) \in \mathsf{IMAC} \times \mathsf{IMAC}$ *we have that* $(P, Q) \in \mathcal{S}$ *implies for all equivalence classes* $C \in \mathsf{IMAC}/_{\mathcal{S}}$*:*

(1) $\gamma_i(P, \underline{a}, C) = \gamma_i(Q', \underline{a}, C^\tau)$ *for some* Q', $Q \overset{\tau^*}{\hookrightarrow} Q'$
(2) $\gamma_i(Q, \underline{a}, C) = \gamma_i(P', \underline{a}, C^\tau)$ *for some* P', $P \overset{\tau^*}{\hookrightarrow} P'$
(3) $P \overset{\tau}{\not\hookrightarrow}$ *implies* $\gamma'_m(P, a, C) = \gamma'_m(Q', a, C^\tau)$ *for some* Q', $Q \overset{\tau^*}{\hookrightarrow} Q'$, $Q' \overset{\tau}{\not\hookrightarrow}$
(4) $Q \overset{\tau}{\not\hookrightarrow}$ *implies* $\gamma'_m(Q, a, C) = \gamma'_m(P', a, C^\tau)$ *for some* P', $P \overset{\tau^*}{\hookrightarrow} P'$, $P' \overset{\tau}{\not\hookrightarrow}$.

Processes P *and* Q *are* weak Markovian bisimilar*, denoted* $P \approx_m Q$*, if* $(P, Q) \in \mathcal{S}$ *with* $\mathcal{S}$ *a weak Markovian bisimulation.*

Example 21 *Fig. 7 illustrates the distinguishing power of* $\approx_m$ *by means of a few examples. The first two transition systems are equated by weak Markovian bisimulation since the* $\xrightarrow{b,\nu}$ *branch of the initial state is irrelevant, due to maximal progress. In contrast, the third transition system falls into a different equivalence class of* $\approx_m$*, because the rates cumulated by* γ'_m *of moving with timed action* a *from the initial state differ. Note that the latter two transition systems would be identified under the usual notion of weak bisimulation (ignoring the rate label) while they would be distinguished by branching bisimulation [36].*

Weak Markovian bisimulation is a congruence for all operators of the language, except choice. This deficiency is inherited from classical weak bisimulation, we hence define the coarsest congruence contained in $\approx_m$, Markovian observational congruence.

In order to preserve congruence, the initial τ-steps have to be matched in a more rigid way than in the definition of $\approx_m$ where, according to the definition

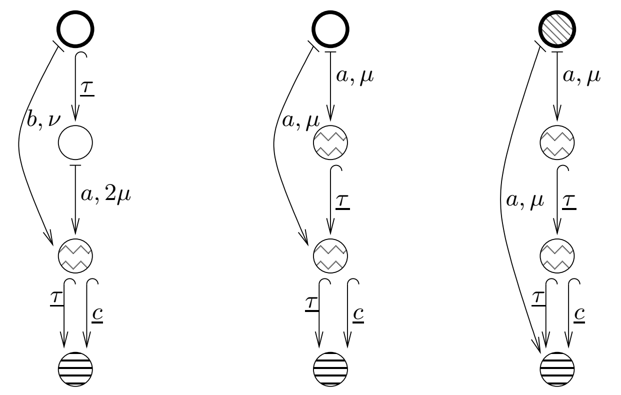

Fig. 7. Some distinguishing examples for weak Markovian bisimulation.

of function γ_i, τ-steps do not have to be mimicked by a matching process. This is actually the reason why the congruence property for choice fails. To fix this, we demand that from the initial state each transition $\overset{a}{\hookrightarrow}$ has to be matched by a sequence $\overset{\tau^*}{\hookrightarrow}\overset{a}{\hookrightarrow}\overset{\tau^*}{\hookrightarrow}$. In this way, also internal transitions are mimicked by at least one internal transition. This gives rise to the following definition:

Definition 22 *(Markovian observational congruence.) Two processes P and Q are* Markovian observational congruent*, denoted $P \approx^c_m Q$, if and only if for all equivalence classes $C \in \mathsf{IMAC}/_{\approx_m}$:*

(1) $P \overset{a}{\hookrightarrow} P'$, $P' \in C$ *implies* $Q \overset{\tau^*}{\hookrightarrow}\overset{a}{\hookrightarrow}\overset{\tau^*}{\hookrightarrow} Q'$, *for some* $Q' \in C$
(2) $Q \overset{a}{\hookrightarrow} Q'$, $Q' \in C$ *implies* $P \overset{\tau^*}{\hookrightarrow}\overset{a}{\hookrightarrow}\overset{\tau^*}{\hookrightarrow} P'$, *for some* $P' \in C$
(3) $P \overset{\tau}{\not\hookrightarrow}$ *implies* $\gamma'_m(P, a, C) = \gamma'_m(Q, a, C^\tau)$ *for some* Q', $Q \overset{\tau^*}{\hookrightarrow} Q'$, $Q' \overset{\tau}{\not\hookrightarrow}$
(4) $Q \overset{\tau}{\not\hookrightarrow}$ *implies* $\gamma'_m(Q, a, C) = \gamma'_m(P, a, C^\tau)$ *for some* P', $P \overset{\tau^*}{\hookrightarrow} P'$, $P' \overset{\tau}{\not\hookrightarrow}$.

Theorem 23 *$\approx^c_m$ is a congruence with respect to all operators in* IMAC.

6.4 Equational theory

We now introduce a complete axiom system for $\approx^c_m$ on IMAC. An occurrence of process variable X is *strongly guarded* in term P if P has a sub-term $\underline{a} \, . \, Q$ ($\underline{a} \neq \underline{\tau}$) or $(a, \lambda) \, . \, Q$ ($a \neq \tau$) such that this occurrence of X is in Q. Process variable X is strongly guarded in P if each occurrence of it is strongly guarded. Term P is strongly guarded if all its process variables are. Finally, P is called strongly regular if it is strongly guarded and if any sub-term of P of the form $recX.Q$ does neither contain an occurrence of $||$ nor an occurrence of $[f]$ in Q. Let $\underline{\mathcal{A}}$ be the axiom system consisting of axioms (B1) through (B3) (cf. Table 2), (B4′) (cf. Table 5), (R1) through (R3) (cf. Table 2 where guarded means strongly guarded), (RL1) through (RL3) (cf. Table 5), and the axioms listed in Table 7. This axiom system is sound and complete for strongly regular

Table 7
Axioms for IMAC for Markovian observational congruence

(I1)	$\underline{a}\,.\,P + \underline{a}\,.\,P$	$=$	$\underline{a}\,.\,P$
(RL4)	$(\underline{a}\,.\,P)[f]$	$=$	$\underline{f(a)}\,.\,P[f]$
(τ1)	$P + \underline{\tau}\,.\,P$	$=$	$\underline{\tau}\,.\,P$
(τ2)	$\underline{a}\,.\,\underline{\tau}\,.\,P$	$=$	$\underline{a}\,.\,P$
(τ3)	$\underline{a}\,.\,(P + \underline{\tau}\,.\,Q) + \underline{a}\,.\,Q$	$=$	$\underline{a}\,.\,(P + \underline{\tau}\,.\,Q)$
(τ4)	$(a,\lambda)\,.\,\underline{\tau}\,.\,P$	$=$	$(a,\lambda)\,.\,P$
(τ5)	$(a,\lambda)\,.\,P + \underline{\tau}\,.\,Q$	$=$	$\underline{\tau}\,.\,Q$
(P1$''$)	for $P = \sum_i (a_i,\lambda_i)\,.\,P_i + \sum_k \underline{a_k}\,.\,P_k$ and $Q = \sum_j (b_j,\mu_j)\,.\,Q_j + \sum_l \underline{b_l}\,.\,Q_l$: $\begin{aligned} P \,\|_A\, Q = {} & \sum_i \{ (a_i,\lambda_i)\,.\,(P_i \,\|_A\, Q) \mid a_i \in \mathbf{A}, a_i \notin A \} \\ & + \sum_j \{ (b_j,\mu_j)\,.\,(P \,\|_A\, Q_j) \mid b_j \in \mathbf{A}, b_j \notin A \} \\ & + \sum_{i,j} \{ (a_i,\lambda_i \odot \mu_j)\,.\,(P_i \,\|_A\, Q_j) \mid a_i, b_j \in A \setminus \underline{\mathbf{A}}, a_i = b_j \} \\ & + \sum_k \{ \underline{a_k}\,.\,(P_k \,\|_A\, Q) \mid \underline{a_k} \in \underline{\mathbf{A}}, \underline{a_k} \notin A \} \\ & + \sum_l \{ \underline{b_l}\,.\,(P \,\|_A\, Q_l) \mid \underline{b_l} \in \underline{\mathbf{A}}, \underline{b_l} \notin A \} \\ & + \sum_{k,l} \{ \underline{a_k}\,.\,(P_k \,\|_A\, Q_l) \mid \underline{a_k}, \underline{b_l} \in A \cap \underline{\mathbf{A}}, \underline{a_k} = \underline{b_l} \} \end{aligned}$		

terms in IMAC. Let IMAC_{SR} denote the set of strongly regular terms of IMAC.

Theorem 24 *For any $P, Q \in \mathsf{IMAC}_{SR}$, $\underline{\mathcal{A}} \vdash (P = Q)$ if and only if $P \approx^c_m Q$.*

The axiom (I1) is a specialisation of the traditional idempotence axiom for choice ($P + P = P$), that is invalidated by (B4$'$). Axioms (τ1)-(τ3) are the usual axioms for classical weak bisimulation, and (τ4) is a direct adaption of (τ2). Note that no delayed variant of (τ3) is included in Table 7 (such as $(\lambda)\,.\,(P + \tau\,.\,Q) + (\lambda)\,.\,Q = (\lambda)\,.\,(P + \tau\,.\,Q)$). This is a consequence of the fact that Markovian observational congruence treats Markovian transitions in the same way as non-internal transitions are treated in branching bisimulation (congruence). Recall that axiom (τ3) is the distinguishing axiom that is valid for weak, but invalid for branching bisimulation. Axiom (τ5) is the axiomatic counterpart of maximal progress. No time will be spent in the presence of an internal, immediate alternative.

7 Separating time and action transitions

The Markovian process algebras like TIPP, PEPA and EMPA are all based on the principle of considering actions and time consumption as a single entity. We have argued before that a major issue in this setting is to compute the resulting rate in case of synchronisation (modelled by the operator $\odot$). Evidently,

$$(a, \lambda) \,.\, P \,||_a\, ((a, \mu) \,.\, Q + (a, \nu) \,.\, R)$$

equals

$$(a, ?) \,.\, (P \,||_a\, Q) + (a, ??) \,.\, (P \,||_a\, R)$$

According to the scheme that we pursued so far, we would obtain $? = \lambda \odot \mu$ and $?? = \lambda \odot \nu$. Intuitively though, a synchronisation can take place if both partners are willing to participate. Clearly, $(a, \lambda) \,.\, P$ is ready to participate after a delay determined by λ and $(a, \mu) \,.\, Q$ is ready to offer a after a delay determined by μ. Given our intuition, the actual synchronisation could take place after *both* delays have passed, i.e., after the maximum of the exponential distributions (of rate λ and μ). Since the class of exponential distributions is, however, not closed under maximum (product, cf. Section 3), this solution is not feasible. Therefore, some function $\odot$ on rates is used instead, with the effect that the resulting distribution for a synchronisation action does not correspond to what one intuitively expects.[7] For instance, for the above example the probability that (a, μ) synchronises with (a, λ) (as opposed to (a, ν)) is given by

$$\frac{\lambda \odot \mu}{(\lambda \odot \mu) + (\lambda \odot \nu)}$$

This does not correspond to the probability that the maximum of an exponential distribution with rate μ and one with rate λ is smaller than the maximum of an exponential distribution with rate μ and one with rate ν.

In this section we propose a different view and treat the occurrence of actions and the consumption of time as strictly separate. This embodies that phases, during which one or more actions occur (together with their corresponding state changes), but where no time elapses, alternate with phases where time

[7] In the process calculus IMAC the maximum of two exponential distributions may be mimicked by incorporating additional internal actions. For instance $(\tau, \lambda) \,.\, \underline{a} \,.\, P \,||_a\, (\tau, \mu) \,.\, \underline{a} \,.\, Q$ denotes a process that can perform a after the maximum of exponential distributions with rate λ and μ.

passes, but during which no actions happen. The resulting calculus is a proper extension of classical process algebras and the algebra for describing CTMCs, MC (cf. Section 4). We will show that this separation — which is also applied in several timed process algebras like TCCS [80], Temporal CCS [104] and Timed CSP [92]— results in a synchronisation scheme that corresponds to the aforementioned intuition. In the context of Markovian process algebras, this separation has been brought up by Hermanns and Rettelbach [55] and has been worked out in [47].

7.1 Syntax

The syntax of the algebra IMC is given by

$$P ::= \sum_{i \in I} a_i \,.\, P \;\Big|\; \sum_{i \in I} (\lambda_i) \,.\, P \;\Big|\; P \,||_A\, P \;\Big|\; P[f] \;\Big|\; X \;\Big|\; recX.P$$

7.2 Semantics

Due to the explicit separation between the (probabilistic) advance of time and the occurrence of actions, the semantics of IMC is defined using two transition relations. The transition relation $\xrightarrow{a}$ corresponds to action occurrences and is defined using the inference rules of Table 6. The transition relation $\overset{\lambda}{\longmapsto}$ represents the passage of time; this relation is the smallest relation defined using the rules of Table 8. (Notice that these rules are simply the rules for MC extended with a straightforward rule for relabelling.)

Table 8
Delay transitions for IMC

$$\sum_{i \in I} (\lambda_i) \,.\, P_i \overset{\lambda_j}{\longmapsto}_j \; P_j \quad (j \in I) \qquad \frac{P\{\, recX.P/X \,\} \overset{\lambda}{\longmapsto}_i \; P'}{recX.P \overset{\lambda}{\longmapsto}_i \; P'}$$

$$\frac{P \overset{\lambda}{\longmapsto}_i \; P'}{\begin{array}{c} P \,||_A\, Q \overset{\lambda}{\longmapsto}_{(i,*)} \; P' \,||_A\, Q \\ Q \,||_A\, P \overset{\lambda}{\longmapsto}_{(*,i)} \; Q \,||_A\, P' \end{array}} \qquad \frac{P \overset{\lambda}{\longmapsto}_i \; P'}{P[f] \overset{\lambda}{\longmapsto}_i \; P'[f]}$$

A notion of strong bisimulation for IMC can be defined by superposing the classical definition of strong bisimulation with the definition of lumping equivalence on MC (Definition 8). To incorporate maximal progress, we define

Definition 25 *(Strong lumping bisimulation.) An equivalence relation $\mathcal{S}$ on* IMC *is a* strong lumping bisimulation *if and only if for any pair $(P, Q) \in \mathsf{IMC}\times\mathsf{IMC}$ we have that $(P, Q) \in \mathcal{S}$ implies for all actions a and all equivalence classes $C \in \mathsf{MAC}/_{\mathcal{S}}$:*

(1) $P \xrightarrow{a} P'$, $P' \in C$ implies $Q \xrightarrow{a} Q'$, for some $Q' \in C$
(2) $Q \xrightarrow{a} Q'$, $Q' \in C$ implies $P \xrightarrow{a} P'$, for some $P' \in C$
(3) $P \stackrel{\tau}{\not\longrightarrow}$ implies $\gamma_m(P, C) = \gamma_m(Q, C)$
(4) $Q \stackrel{\tau}{\not\longrightarrow}$ implies $\gamma_m(Q, C) = \gamma_m(P, C)$

Processes P and Q are strongly lumping bisimilar*, denoted $P \sim_l Q$, if $(P, Q) \in \mathcal{S}$ with $\mathcal{S}$ a strong lumping bisimulation.*

The definitions of weak lumping bisimulation ($\approx_l$) and lumping observational congruence ($\approx_l^c$) are directly obtained from weak Markovian bisimulation (Definition 20), respectively Markovian observational congruence (Definition 22), by replacing $\gamma'_m(\_, a, C)$ by $\gamma_m(\_, C)$.

Theorem 26 *$\sim_l$ and $\approx_l^c$ are congruences with respect to all operators in* IMC*.*

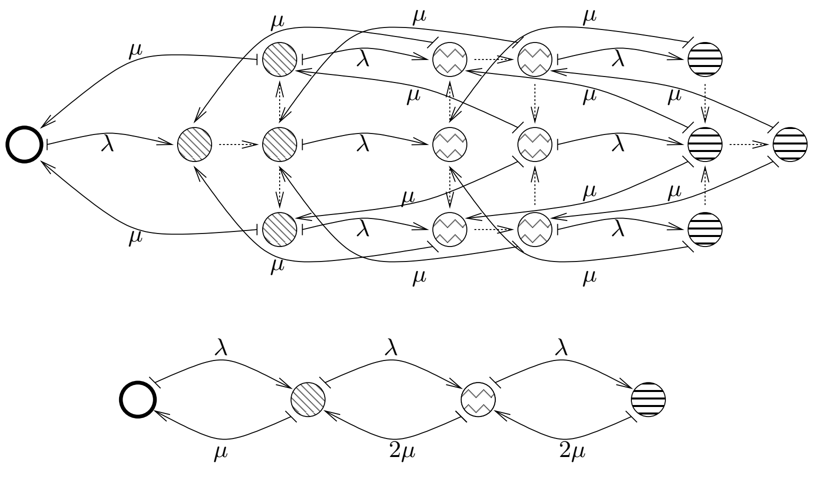

Fig. 8. Semantic model of the M/M/2/1-queueing system.

Example 27 *Consider a queueing system in which jobs arrive and wait until they are executed by two servers. An infinite population of jobs is assumed. Jobs arrive with an exponentially distributed inter-arrival time (with rate λ) while the processing delay of jobs by each server takes exponential time (with rate μ). This system is known as a M/M/2/1-queueing system, where M* *stands for exponential distribution of the arrival and service process, respectively, 2 indicates the number of servers, and* 1 *denotes the buffer capacity. We consider a modular description of an $M/M/2/1$-queueing system. The use of immediate actions is crucial for the modularity of the specification [56].*

The queueing system is constructed from a few components: an arrival process Arr, a buffer Buff and a server Multi consisting of two processors Proc. The arrival process is modelled as a Poisson stream with rate λ:

$$Arr := (\lambda)\,.\,a\,.\,Arr \qquad Buff := a\,.\,d\,.\,Buff$$

A simple one-place buffer is used to disconnect the arrival stream from the servers. If the buffer is non-empty, it can deliver (d) a job to either of the servers. If it is full, incoming jobs are ignored. The two servers are working independently. If possible, a server takes a job out of the queue (d) without spending time. Afterwards it processes the job with rate μ.

$$Multi := Proc\,||_{\varnothing}\,Proc \qquad Proc := d\,.\,(\mu)\,.\,Proc$$

The whole system consists of the arrival stream, queue and multi-server appropriately synchronised. From the environment, the actions a and d are not visible. This is realized by the relabelling function h defined as the identity function on $\mathbf{A}$, *except for $h(a) = \tau$ and $h(d) = \tau$.*

$$Sys := \Big(Arr\,||_a\,(Buff\,||_d\,Multi)\Big)[h]$$

The semantic model obtained via the operational semantic rules is depicted in Fig. 8 (top). For the sake of readability, dotted transitions are used to indicate τ-transitions. One may argue that this model is far away from the model usually expected, depicted below. However, both belong to the same equivalence class of lumping observational congruence. Again, equally shaded states form an equivalence class.

7.4 Equational theory

To complete the picture, we address sound and complete axiomatisations for $\sim_l$ and $\approx_l^c$. Let $\overline{\mathcal{A}}$ be the axiom system consisting of axioms (B1) through (B4) (cf. Table 2), (R1) through (R3) (cf. Table 2), (RL1) and (RL3) (cf. Table 5), (I1), (RL4) (cf. Table 7) and the axioms (RL5) and (P1$'''$) listed in Table 9. This axiom system is sound and complete for regular terms in IMC and lumping bisimulation.

Theorem 28 *For any $P, Q \in \mathsf{IMC}_R$, $\overline{\mathcal{A}} \vdash (P = Q)$ if and only if $P \sim_l Q$.*

Now let $\overline{\mathcal{A}}'$ be the axiom system consisting of the axioms in $\overline{\mathcal{A}}$ (where guarded means strongly guarded), and of ($\tau 1$) through ($\tau 3$) (cf. Table 7), and of ($\tau 4'$), $(\lambda) \,.\, \tau \,.\, P = (\lambda) \,.\, P$. $\overline{\mathcal{A}}'$ is sound and complete for strongly regular terms in IMC and lumping observational congruence.

Theorem 29 *For any $P, Q \in \mathsf{IMC}_{SR}$, $\overline{\mathcal{A}}' \vdash (P = Q)$ if and only if $P \approx_l^c Q$.*

Table 9
Axioms for IMC for lumping bisimulation

(RL5)	$((\lambda) \,.\, P)[f] \;=\; (\lambda) \,.\, P[f]$
($\tau 5'$)	$(\lambda) \,.\, P + \tau \,.\, Q \;=\; \tau \,.\, Q$
	for $P = \sum_i (\lambda_i) \,.\, P_i + \sum_k a_k \,.\, P_k$ and $Q = \sum_j (\mu_j) \,.\, Q_j + \sum_l b_l \,.\, Q_l$:
(P1''')	$P \,\|_A\, Q = \sum_i (\lambda_i) \,.\, (P_i \,\|_A\, Q) + \sum_j (\mu_j) \,.\, (P \,\|_A\, Q_j) + \sum_{k,l} \{\, a_k \,.\, (P_k \,\|_A\, Q_l) \mid a_k, b_l \in A, a_k = b_l \,\} + \sum_k \{\, a_k \,.\, (P_k \,\|_A\, Q) \mid a_k \notin A \,\} + \sum_l \{\, b_l \,.\, (P \,\|_A\, Q_l) \mid b_l \notin A \,\}$

Example 30 *The two systems depicted in Fig. 8 are lumping observational congruent. To establish this property formally, we shall exemplify a small fragment of the necessary syntactic transformations, applying the complete axiomatisation of lumping observational congruence. For the sake of simplicity, axioms (R1)-(R3) are applied implicitly. First, the starting state is expanded*

using (P1‴), afterwards symmetries and maximal progress can be exploited.

$$
\begin{aligned}
Sys &= (Arr \,||_a\, (Buff \,||_d\, (Proc \,||_\varnothing\, Proc))))[h] \\
&\overset{(P1''')}{=} ((\lambda) \,.\, (a \,.\, Arr \,||_a\, (Buff \,||_d\, (Proc \,||_\varnothing\, Proc))))[h] \\
&\overset{(P1''')}{=} ((\lambda) \,.\, a \,.\, (Arr \,||_a\, (d \,.\, Buff \,||_d\, (Proc \,||_\varnothing\, Proc))))[h] \\
&\overset{(RL5),(RL4)}{=} (\lambda) \,.\, \tau \,.\, (Arr \,||_a\, (d \,.\, Buff \,||_d\, (Proc \,||_\varnothing\, Proc))[h]) \\
&\overset{(\tau 1)}{=} (\lambda) \,.\, (Arr \,||_a\, (d \,.\, Buff \,||_d\, (Proc \,||_\varnothing\, Proc))[h]) \\
&\overset{(RL4),(P1''')}{=} (\lambda) \,.\, (\quad (\lambda) \,.\, (a \,.\, Arr \,||_a\, (d \,.\, Buff \,||_d\, ((Proc \,||_\varnothing\, Proc)))[h] \\
&\qquad\quad + \quad \tau \,.\, (Arr \,||_a\, (Buff \,||_d\, (Proc \,||_\varnothing\, (\mu) \,.\, Proc)))[h] \\
&\qquad\quad + \quad \tau \,.\, (Arr \,||_a\, (Buff \,||_d\, ((\mu) \,.\, Proc \,||_\varnothing\, Proc)))[h] \\
&\qquad)\\
&\overset{(\tau 5)}{=} (\lambda) \,.\, (\quad \tau \,.\, (Arr \,||_a\, (Buff \,||_d\, (Proc \,||_\varnothing\, (\mu) \,.\, Proc)))[h] \\
&\qquad\quad + \tau \,.\, (Arr \,||_a\, (Buff \,||_d\, ((\mu) \,.\, Proc \,||_\varnothing\, Proc)))[h] \\
&\qquad)\\
&\overset{(P1'''),(B2)}{=} (\lambda) \,.\, (\quad \tau \,.\, (Arr \,||_a\, (Buff \,||_d\, (Proc \,||_\varnothing\, (\mu) \,.\, Proc)))[h] \\
&\qquad\quad + \tau \,.\, (Arr \,||_a\, (Buff \,||_d\, (Proc \,||_\varnothing\, (\mu) \,.\, Proc)))[h] \\
&\qquad)\\
&\overset{(I1)}{=} (\lambda) \,.\, (\tau \,.\, (Arr \,||_a\, (Buff \,||_d\, (Proc \,||_\varnothing\, (\mu) \,.\, Proc)))[h]) \\
&\overset{(\tau 1)}{=} (\lambda) \,.\, (\underbrace{(Arr \,||_a\, (Buff \,||_d\, (Proc \,||_\varnothing\, (\mu) \,.\, Proc)))[h]}_{Sys'})
\end{aligned}
$$

After reducing the expanded starting term, a sub-term Sys' remains. Following the above way, the sub-term Sys' can be explored further, leading to

$$Sys' := (\mu) \,.\, Sys + (\lambda) \,.\, Sys''$$

where $Sys'' = (Arr \,||_a\, (Buff \,||_d\, ((\mu) \,.\, Proc \,||_\varnothing\, (\mu) \,.\, Proc)))[h]$. *From that we get*

$$Sys'' := (2\mu) \,.\, Sys' + (\lambda) \,.\, Sys'''$$

where $Sys''' = (Arr \,||_a\, (d \,.\, Buff \,||_d\, ((\mu) \,.\, Proc \,||_\varnothing\, (\mu) \,.\, Proc)))[h]$. *Finally, we get*

$$Sys''' := (2\mu) \,.\, Sys'' + (\lambda) \,.\, Sys'''$$

The transition system that is sketched in Fig. 8 (bottom) describes the be-

haviour of Sys.

8 From specification to quality assurance

8.1 Obtaining a CTMC

By transforming the semantic model into a CTMC and then analysing it by means of numerical solution algorithms for Markov chains, we can obtain performance and reliability measures for a given specification. If the model contains timed transitions only, it directly corresponds to a CTMC. An ordinary differential equation system needs to be solved to obtain the state probabilities of the CTMC at a particular time instant t (transient analysis). Alternatively, solving a linear equation system leads to the state probabilities in the equilibrium (stationary analysis). These limiting probabilities (where $t \to \infty$) are known to exist for arbitrary finite (homogeneous, continuous time) Markov chains. Efficient numerical algorithms are known for either of these tasks [97].

If immediate actions are involved (as it is the case in IMAC and IMC), the situation is different. As discussed in Section 6, immediate actions happen as soon as they become enabled. In order to ensure that this enabling cannot be delayed by further composition, abstraction of immediate actions is mandatory. In the stochastic process, these immediate actions correspond to immediate transitions. The presence of immediate transitions leads to two kinds of states in this process: (vanishing) states with outgoing immediate transitions and (tangible) states without such transitions. If several immediate transitions emanate from a single state, the decision among these alternatives is non-deterministic, and depends on which actions are offered by the environment. If we consider the system as a closed system (which is made explicit by hiding all immediate actions) the decision among several (now internalised) immediate transitions still has to be taken. Note that a non-deterministic decision is conceptually different from an equi-probable decision!

Therefore, a system with non-determinism might be regarded as stochastically under-specified, but actually it does not necessarily have to be so, as in the example in Fig. 8. There non-determinism has been eliminated by application of lumping observational congruence, since non-deterministic alternatives only lead (via some internal, immediate steps) into equivalent states. This exemplifies the way we generally pursue to cope with non-determinism. If non-determinism still remains after factorising with respect to lumping (or Markovian) observational congruence, the specification is stochastically under-specified. In this case one may resort to some notion of scheduler [103] to determine the stochastic process to be analysed. Alternatively, one derives performance bounds.

8.2 Compositional model generation and reduction

From a practitioner's point of view relations such as lumping observational congruence are beneficial both for eliminating immediate transitions, and for alleviating the state space explosion problem by means of lumping. Both effects can be achieved by means of the same strategy, known as compositional generation and reduction. Exploiting the congruence property, the semantics of a complex specification is applied in a component-wise fashion, and interweaved with reduction steps, where sub-terms are replaced by minimised, yet congruent, representatives. Minimising an arbitrary sub-term of a specification in this way does not alter the behaviour of the whole specification. In principle, it is possible to produce the minimised representatives by term rewriting on the level of the syntax [49]. This symbolic approach has been automated for some non-stochastic process calculi [40].

A different approach works on the level of the transition system, factorising the whole state space into equivalence classes of states. A minimal representation is obtained afterwards, representing each class by a single state. This strategy requires a finite state space. The necessary algorithms to compute (strong, weak) Markovian (or lumping) bisimulation have essentially the same complexity as their classical counterparts [47,57]. This embodies that in the worst case strong Markovian bisimulation can be checked in time that is linear in the number of transitions and logarithmically in the number of states. Weak Markovian bisimulation has a worst case time complexity that is cubic in the number of states. By applying compositional minimisation, compositional Markov chain specifications with very large state spaces become tractable, as outlined in [51].

8.3 Exploiting the structure of the specification for analysis

Investigating todays (and future) resource-sharing systems we are usually faced with the problem that the real system is extremely complex having a variety of different hardware and software components. As a consequence, a straightforward modelling technique using stochastic process algebras usually leads directly to models of unmanageable size, known as the *state space explosion problem*. Simplifying the model is a potential way out, but frequently leads to oversimplification implying inaccurate evaluation of the system's properties. Hence, systematic search for structure and symmetry is neccessary, and much of the research in the area of stochastic process algebras has been devoted to this topic indeed. Fig. 9 illustrates several ways to cope with the state space explosion problem, by exploiting the structure of the stochastic process algebra specification. We take this figure as a roadmap to discuss different possibilities that have been studied. A more thorough overview is given in [64].

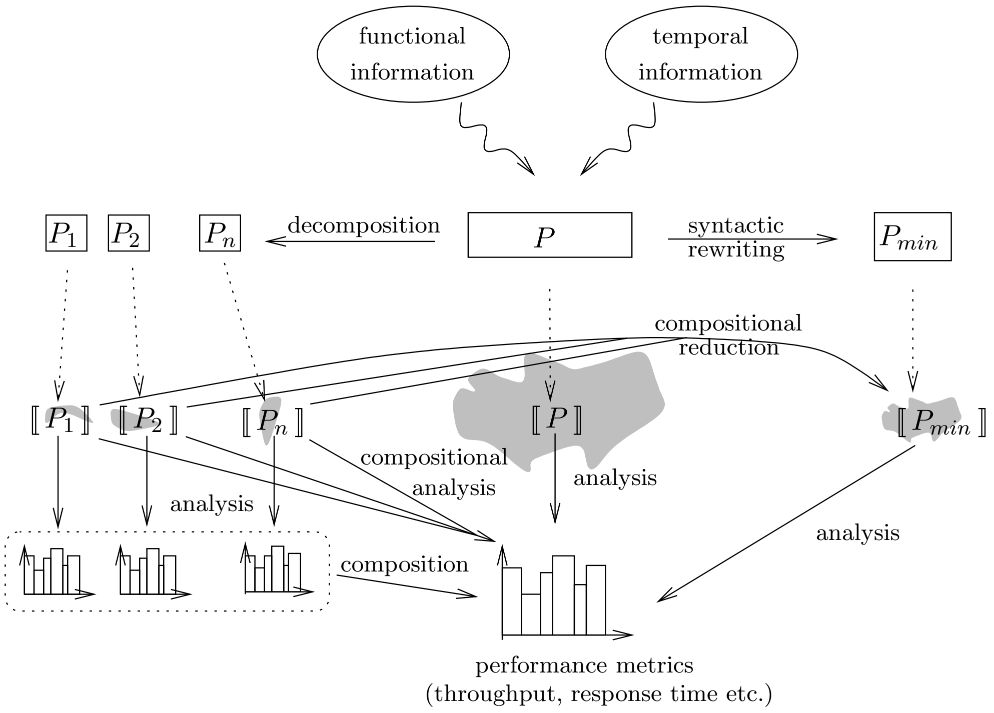

Fig. 9. From stochastic process algebra specification to performance metrics.

A specification P (a stochastic process algebra term) is usually obtained from a detailed study of functional and temporal information of the resource-sharing system to be investigated. This process might involve assumptions about (or measurement of) the timing behaviour of components of the real system, and/or formalising properties of the system from an informal description.

For such a specification P, the direct way to obtain performance metrics for P is depicted in the center of Fig. 9. By applying the formal semantics, a semantic model $[\![P]\!]$ is obtained, that is subsequently analysed as described in Section 8.1. However, the state space of $[\![P]\!]$ will frequently be far too large to be manageable (as indicated by the size of the grey shade). In several cases however, there is a minimised semantic model that is equivalent with respect to an appropriate notion of equivalence (such as $\approx_m^c$ or $\approx_l^c$), and is still manageable [51]. It is depicted in the middle of the right column of the figure. Since the equivalence notion does not affect functional and temporal properties of a specification, performance metrics can equally be obtained from the minimised semantic model. However, it is not clear how to reach that minimised model. Minimising $[\![P]\!]$ suffers from the fact that $[\![P]\!]$ is intractably large. But, as mentioned in Section 8.2 it is possible to obtain a term P_{min} by term rewriting on the level of the syntax, such that $[\![P_{min}]\!]$ has a minimal state space. We have exemplified this strategy in Section 7 (cf. Example 30). Another way to reach this minimal state space is based on compositional reduction, as discussed in Section 8.2. The complete specification P is decomposed into subterms $P_1 \dots P_n$. Each of these subterms have their respective semantic model $[\![P_1]\!] \dots [\![P_n]\!]$ that can be minimised in isolation and then be composed in a stepwise manner, possibly involving further minimisation steps. The congruence property is crucial for compositional reduction.

The remaining paths depicted on the left in Fig. 9 have not been covered in this paper so far. They are all based on decomposition of the specification P into subterms with their corresponding semantic models $[\![P_1]\!] \dots [\![P_n]\!]$. These models can either be analysed in isolation, or their analysis can be mixed. The latter approach, compositional analysis, has been investigated by Mertsiotakis and others [77,65,78,14].

Typically, compositional analysis methods are tailored to models of a particular shape (i.e. of a certain syntactic subclass), they usually provide approximate results for the stationary state probabilities. For instance, if the model contains rates of very different dimensions, like in models involving reliability aspects, this leads to so-called *stiff* Markov chains, for which the standard solution effort is drastically increased. Such models are the attack point for *time-scale decomposition* [65], where the state space is partitioned into fast and slow components, based on a distinction between fast and slow actions, according to a threshold value for the rate. The generation of the whole state space at once is avoided; only a single partition is held in memory at a time. Under certain conditions (near complete decomposability) the accuracy of the results is excellent.

The most elegant way traverses via the lower left corner of Fig. 9. However it is only applicable to highly restricted syntactic subclasses. Here, the performance metrics (usually steady-state probabilities) are calculated for each individual submodel $[\![P_1]\!] \dots [\![P_n]\!]$. Afterwards, the overall performance metrics are composed from the individual results, for instance by multiplying individual state probabilities. Hillston and co-authors have studied such *product form solutions* in the setting of stochastic process algebras [44,66,67,96].

9 Non-Markovian process algebra

The calculi we have discussed so far have in common that delays are governed by exponential distributions. In this section we will briefly discuss how — on the basis of separating time- and action-transitions — arbitrary, non-Markovian probability distributions can be supported. We have shown that due to the memory-less property, exponential distributions can be smoothly incorporated into an interleaved setting. As a consequence, Milner's expansion law can be easily obtained for the exponential case. If we allow actions to be delayed by arbitrary distributions, though, the following equation is *invalid* (for arbitrary distributions F and G):

$$(F)\,.\,a\,.\,P\,||_{\varnothing}\,(G)\,.\,b\,.\,Q = (F)\,.\,(a\,.\,P\,||_{\varnothing}\,(G)\,.\,b\,.\,Q) + (G)\,.\,(a\,.\,P\,||_{\varnothing}\,b\,.\,Q) \quad (2)$$

since, for instance, after the delay imposed by F, the residual lifetime of G has to be computed in order to correctly determine the remaining delay before b becomes enabled.

The idea that we pursue in this section is to make a distinction between three activities: (i) starting a delay, (ii) finishing a delay, and (iii) the occurrence of immediate actions. This syntactic separation and a semantical model (based on a variant of timed automata [4]) that supports this view has been brought up by D'Argenio, Katoen and Brinksma [25]. A similar distinction, though only at a semantical level, has been made in GSMPA [16].

9.1 Syntax

Since the emphasis of this paper is on Markovian process algebras, we do not give a complete description of this approach — for this we refer to [25–27] — but sketch the basic ideas. The three distinguished activities mentioned above are represented in the syntax as follows. Let $\mathcal{C}$ be a set of clocks with $(x, F) \in \mathcal{C}$ for x a clock name and F an arbitrary distribution. We abbreviate (x, F) by x_F. Let $C \subseteq \mathcal{C}$ be a finite set of clocks. A clock is initialised with a sample of its associated distribution. The term $\{\!|\, C \,|\!\}\, P$ behaves like P after all clocks in C have been initialised according to their distribution. Once initialised a clock counts down until it expires (i.e., when it reaches the value 0). All clocks run at the same speed. On the expiration of clocks, the occurrence of actions is triggered. The term $C \mapsto P$ behaves after the expiration of all clocks in C like P. Notice that $\varnothing \mapsto P$ behaves like P. Let I be as before. Then the syntax of $\spadesuit$ is:

$$P ::= \sum_{i \in I} C_i \mapsto P \;\Big|\; a\,.\,P \;\Big|\; \{\!|\, C \,|\!\}\, P \;\Big|\; P\,||_A\,P \;\Big|\; P[f] \;\Big|\; X \;\Big|\; recX.P$$

9.2 Semantics

The semantics of $\spadesuit$ is defined in terms of stochastic automata, a kind of timed automata where clocks are initialised according to probability distribution functions. States are equipped with a finite set of clocks that are set according to their distribution function as soon as the state is reached. For state s this set is denoted by $\kappa(s)$. Transitions are labelled with pairs of actions and sets of clocks: $s \xrightarrow{a,C} s'$ denotes that the system can move from state s to s' while performing action a if all clocks in C have expired (i.e. have reached value 0).

To associate a stochastic automaton to a given term P in the language, we define its different components[8]. In order to define the automaton associated to a parallel composition, we introduce the operation $\overline{\mathsf{ck}}$. $\overline{\mathsf{ck}}(P)$ is a process that behaves like P except that no clock is set at the very beginning. The clock setting function κ is defined as the smallest set satisfying: $\kappa(\mathbf{0}) = \kappa(a\,.\,P) = \kappa(\overline{\mathsf{ck}}(P)) = \varnothing$, $\kappa(\sum_i C_i \mapsto P_i) = \bigcup_i \kappa(P_i)$, $\kappa(P \,||_A\, Q) = \kappa(P) \cup \kappa(Q)$, $\kappa(\{\!|\, C \,|\!\}P) = C \cup \kappa(P)$ and $\kappa(recX.P) = \kappa(P)$. $\longrightarrow$ is defined as the smallest relation satisfying the rules in Table 10.

Table 10
Operational semantics for ♤

$$a\,.\,P \xrightarrow{a,\varnothing} P \qquad\qquad \frac{P_j \xrightarrow{a,C} P_j'}{\sum_{i\in I} C_i \mapsto P_i \xrightarrow{a,C\cup C_j} P_j'} \quad (j \in I)$$

$$\frac{P \xrightarrow{a,C'} P'}{\{\!|\, C \,|\!\}P \xrightarrow{a,C'} P'} \qquad\qquad \frac{P \xrightarrow{a,C} P'}{\begin{array}{c} P \,||_A\, Q \xrightarrow{a,C} P' \,||_A\, \overline{\mathsf{ck}}(Q) \\ Q \,||_A\, P \xrightarrow{a,C} \overline{\mathsf{ck}}(Q) \,||_A\, P' \end{array}} \quad (a \notin A)$$

$$\frac{P \xrightarrow{a,C} P',\; Q \xrightarrow{a,C'} Q'}{P \,||_A\, Q \xrightarrow{a,C\cup C'} P' \,||_A\, Q'} \quad (a \in A) \qquad\qquad \frac{P \xrightarrow{a,C} P'}{\overline{\mathsf{ck}}(P) \xrightarrow{a,C} P'}$$

$$\frac{P \xrightarrow{a,C} P'}{P[f] \xrightarrow{f(a),C} P'[f]} \qquad\qquad \frac{P\{\, recX.P/X \,\} \xrightarrow{a,C} P'}{recX.P \xrightarrow{a,C} P'}$$

Clearly, since arbitrary probability distributions are allowed, the underlying stochastic process does not have to satisfy the Markov property (cf. Section 3.2). Due to the presence of immediate actions we cannot even guarantee in general that a stochastic process is obtained anyway — like for IMAC and IMC. In case non-determinism is resolved (or absent), it can be shown [26] that the underlying stochastic process is a homogeneous, mono-rated[9] *generalised semi-Markov process* (GSMP), a model that is suited as a mathematical frame-

[8] Here we assume that P does not contain any name clashes of clock variables. This not a severe restriction since terms that suffer from such name clash can always be properly renamed into a term without such name clash [25].
[9] Since all clocks run at the same speed.

work for the study of stochastic discrete-event systems [95].

Example 31 *To exemplify the operational semantics we consider a queueing system similar to the one of Example 27, where the inter-arrival time of jobs and the service time of jobs are governed by an arbitrary distribution. For simplicity we take a single server. To show the regularity we take a buffer of infinite capacity. The resulting system is known as a $G/G/1/\infty$-queueing system. The compositional specification amounts to:*

$$\begin{aligned}
\mathit{Buff}_0 &:= a \,.\, \mathit{Buff}_1 \\
\mathit{Buff}_{i+1} &:= a \,.\, \mathit{Buff}_{i+2} + d \,.\, \mathit{Buff}_i \text{ for } i \geqslant 0 \\
\mathit{Arr} &:= \{\!|\, x_F \,|\!\}\{x_F\} \mapsto a \,.\, \mathit{Arr} \\
\mathit{Proc} &:= d \,.\, \{\!|\, y_H \,|\!\}\{y_H\} \mapsto c \,.\, \mathit{Proc} \\
\mathit{Sys} &:= \mathit{Arr} \,||_a\, (\mathit{Buff}_0 \,||_d\, \mathit{Proc})
\end{aligned}$$

Using the operational semantics of Table 10 it can be shown that the semantics of the Sys-specification boils down to the stochastic automaton depicted in Fig. 10. Here, we represent a state s as a circle containing the clocks that are to be set in s, and denote edges by arrows. Empty sets are omitted; in particular d stands for $d, \varnothing$.

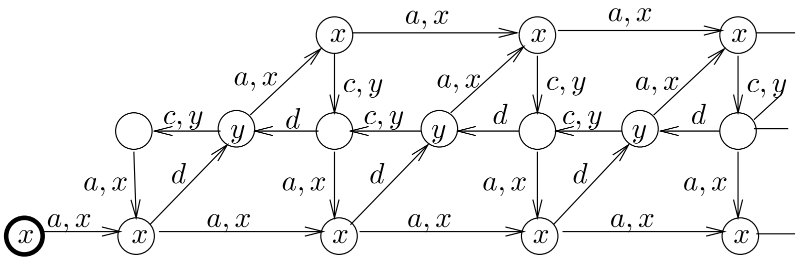

Fig. 10. Stochastic automaton of the compositional $G/G/1/\infty$-queueing system.

9.3 Equivalences and equational theory

Different forms of bisimulation have been proposed for ♤ [25]. Structural bisimulation only considers the structure of a stochastic automaton, is strongly related to strong bisimulation and allows to obtain an expansion law in a neat way. For instance,

$$\{\!|\, x_F \,|\!\}\, P' \,||_\varnothing\, \{\!|\, y_G \,|\!\}\, Q'$$

where $P' = \{\, x_F \,\} \mapsto a \,.\, P$ and $Q' = \{\, y_G \,\} \mapsto b \,.\, Q$, is structurally bisimilar to

$$\{\!|\, x_F, y_G \,|\!\}\Big((\{\, x_F \,\} \mapsto a \,.\, (P \,||_\varnothing\, Q')) + (\{\, y_G \,\} \mapsto b \,.\, (P' \,||_\varnothing\, Q))\Big).$$

The reader is invited to compare this to equation (2). Structural bisimulation does not equate terms that might be equal from the probabilistic point of view — like Markovian bisimulation and lumping observation congruence do — but which are not structural bisimilar. For instance, an equivalent of

$$(\lambda) \,.\, P + (\mu) \,.\, P = (\lambda{+}\mu) \,.\, P$$

is not preserved under structural bisimulation. In order to consider such equivalences and axioms, the interpretation of stochastic automata in terms of (highly infinite) probabilistic transition systems must be considered. Such systems can be considered as a continuous variant of the alternating model [42] and are similar to simple probabilistic systems [94]. By defining a notion of probabilistic bisimulation (a continuous variant of Larsen and Skou's probabilistic bisimulation [74]) on these infinite transition systems, one obtains

$$(\{\!|\, x_F \,|\!\}\{\, x_F \,\} \mapsto P) + (\{\!|\, y_G \,|\!\}\{\, y_G \,\} \mapsto P) = \{\!|\, z_H \,|\!\}\{\, z_H \,\} \mapsto P,$$

where H equals the minimum of F and G.[10] For more details see [25].

10 Conclusions and challenges

10.1 Summary of results

In this paper we have summarised the basic theoretical results in the field of stochastic process algebras. We have concentrated on the different ways in which random durations can be incorporated into a classical process algebra, and have extensively treated notions of equivalence, and sound and complete axiomatisations. We conclude this survey by briefly considering the relationships between the various Markovian calculi discussed in this paper, see Fig. 11.

Let PA denote classical process algebra (over the signature $a \,.\, \_$, $\_ + \_$, $\_ ||_A \_$, $recX.\_$, $\_[f]$) with observational congruence as notion of equivalence. A line connects two calculi, whenever the upper one is a conservative algebraic extension of the lower, in the sense of [29]. For instance, IMAC conservatively extends IMC since IMC's prefix $(\lambda) \,.\, P$ corresponds to IMAC's prefix $(\tau, \lambda) \,.\, P$ (cf. footnote 7 on page 31). Due to the non-standard semantics of prefixing

[10] Recall that in case of exponential distributions this means that if F and G are exponential distributions with rates λ and μ, respectively, then H is an exponential distribution with rate $\lambda{+}\mu$.

(that invalidates idempotence for choice) MAC inherits from MC that it fails to be a conservative algebraic extension of PA. Note that, even though IMAC is the top element in this lattice, we prefer the framework of IMC, since the additional expressiveness of IMAC is lacking an intuitive interpretation, because it cannot represent the maximum of two exponential distributions.

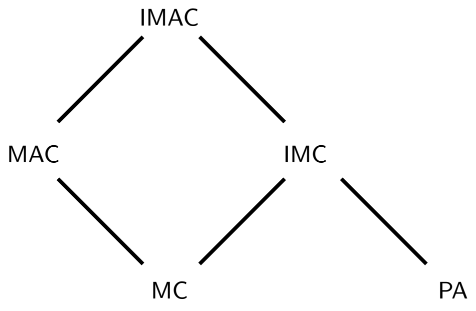

Fig. 11. Relation between the discussed Markovian calculi.

We have restricted to (strongly) guarded expressions just for the sake of keeping the presentation concise. To extend the algebras to unguarded expressions is feasible, but involves some non-standard axioms to capture the interplay of maximal progress and divergence [52].

10.2 Trends and future work

Non-Markovian process algebra

The main issues in defining semantics and equivalences for Markovian process algebras have been settled, as explained in this paper. Although for arbitrary distributions some promising developments are ongoing (like the approach for ♤ discussed in Section 9), several questions are to be considered for this case. For instance, it has not been investigated yet how a notion like Markovian observational congruence can be adapted to the general case. In addition, it is unclear how the various semantical models proposed so far — like event structures [18], stochastic task graphs [59], step and ST-semantics [16] and stochastic automata [25] — are interrelated.

Compact representation of state spaces

One of the main problems is the state-space explosion. In Section 8 we discussed some approaches to circumvent this problem to a certain extent: model reduction using equivalences, compositional model reduction, and exploiting the structure for analysis purposes. Various case studies show that despite these techniques there is a need to combat the state space explosion problem even more. An interesting trend is to consider variants of multi-terminal binary decision diagrams (MTBDDs) to encode Markovian transition systems in an efficient way [41], and to calculate equivalence classes on the basis of this representation [53,57].

Specification of performance measures

Traditionally, performance measures are obtained from Markov chains by including real-valued weights (called rewards) into the model and calculating the desired performance measures using a weighted sum over steady state probabilities. There are currently (more or less) two approaches for the specification of performance measures, like throughput, utilisation, in process algebras. In the first approach rewards are included as “first class citizens” of the Markovian process algebra. The basic idea of this approach is to extend the prefix $(a, \lambda) \,.\, P$ into a triple $(a, \lambda, r) \,.\, P$ where r denotes a real number. Accordingly, semantics and equivalence relations are extended with rewards [9,72]. An alternative approach is to specify the measure of interest in a temporal logic extended with rewards. Such an approach for Hennessy-Milner logic has been proposed in [21]. A challenging direction for future research is to combine such logical approach with model checking techniques [23,8].

Acknowledgements

The remarkable level of todays stochastic process algebra research has only been possible, since scientists of both communities, theoretical computer science and performance evaluation, worked closely together. Stimulating was also the close international cooperation of research groups with different background. We are particularly grateful to N. Götz, M. Rettelbach and V. Mertsiotakis, the hardcore of early research years and L. Kerber, U. Klehmet, J. Meyer-Kayser, and M. Siegle, our current colleagues in Erlangen. Many students contributed also significantly to our theoretical investigations as well as to the tool development. In addition, we would like to acknowledge the valuable comments and suggestions of Pedro R. D’Argenio on an earlier version of this paper and like to thank the reviewers for their constructive comments.

References

[1] M. Ajmone Marsan, A. Bianco, L. Ciminiera, R. Sisto and A. Valenzano. A LOTOS extension for the performance analysis of distributed systems. *IEEE/ACM Trans. on Networking*, **2**(2), 151–164, 1994.

[2] M. Ajmone Marsan, G. Balbo and G. Conte. *Performance Models of Multiprocessor Systems.* MIT Press, 1986.

[3] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets.* John Wiley, 1995.

[4] R. Alur and D.L. Dill. A theory of timed automata. *Th. Comp. Sc.*, **126**: 183–235, 1994.

[5] J.C.M. Baeten and J.A. Bergstra. Real time process algebra. *Form. Asp. of Comp.*, **3**(2):142–188, 1991.

[6] J.C.M. Baeten, J.A. Bergstra and S.A. Smolka. Axiomatizing probabilistic processes: ACP with generative probabilities. *Inf. & Comp.*, **121**: 234–255, 1995.

[7] C. Baier and H. Hermanns. Weak bisimulation for fully probabilistic processes. In O. Grumberg, ed, *Computer-Aided Verification*, LNCS 1254, pages 119–130, 1997.

[8] C. Baier, J.-P. Katoen and H. Hermanns. Approximate symbolic model checking of continuous-time Markov chains. In J.C.M. Baeten and S. Mauw, eds, *Concurrency Theory*, LNCS 1664, pages 146–162, 1999.

[9] M. Bernardo. An algebra-based method to associate rewards with EMPA terms. In P. Degano and R. Gorrieri, eds, *Automata, Languages and Programming*, LNCS 1256, pages 358–368, 1997.

[10] M. Bernardo, L. Donatiello and R. Gorrieri. Modeling and analyzing concurrent systems with MPA. In [61], pages 71–88.

[11] M. Bernardo and R. Gorrieri. A tutorial on EMPA: a theory of concurrent processes with nondeterminism, priorities, probabilities and time. *Th. Comp. Sc.*, **202**: 1–54, 1998.

[12] M. Bernardo, R. Gorrieri and M. Roccetti. Formal performance modelling and evaluation of an adaptive mechanism for packetised audio over the Internet. *Form. Asp. of Comp.*, **10**: 313–337, 1999.

[13] M. Bernardo, R. Cleaveland, S. Sims and W. Stewart. TwoTowers: a tool integrating functional and performance analysis of concurrent systems. In S. Budkowski, A. Cavalli and E. Najm, eds, *Proc. Joint Int. Conf. on Formal Description Techniques for Distributed Systems and Communication Protocols and Protocol Spec., Testing, and Ver.*, pages 457–467, Kluwer, 1998.

[14] H.C. Bohnenkamp and B.R. Haverkort. Semi-numerical solution of stochastic process algebra models. In [69], pages 228–244.

[15] T. Bolognesi, F. Lucidi and S. Trigila. Converging towards a timed LOTOS standard. *Comp. Standards & Interfaces*, **16**: 87–118, 1994.

[16] M. Bravetti, M. Bernardo and R. Gorrieri. Towards performance evaluation with general distributions in process algebras. In D. Sangiorgi and R. de Simone, eds, *Concurrency Theory*, LNCS 1466, pages 405–422, 1998.

[17] E. Brinksma, J.-P. Katoen, R. Langerak and D. Latella. Performance analysis and true concurrency semantics. In T. Rus and C. Rattray, eds, *Theories and Experiences for Real-Time System Development*, pages 309–337, 1994.

[18] E. Brinksma, J.-P. Katoen, R. Langerak and D. Latella. A stochastic causality-based process algebra. *The Comp. J.*, **38**(7): 552–565, 1995.

[19] P. Buchholz. Exact and ordinary lumpability in finite Markov chains. *J. of Appl. Prob.*, **31**: 59–75, 1994.

[20] P. Buchholz. Markovian process algebra: Composition and equivalence. In [61], pages 11–30.

[21] G. Clark. Formalising the specification of rewards with PEPA. In [89], pages 139–160.

[22] D.R. Cox. A use of complex probabilities in the theory of stochastic processes. *Proc. Cambridge Philosophy Society*, **51**: 313–319, 1955.

[23] L. de Alfaro. How to specify and verify the long-run average behavior of probabilistic systems. In *LICS*, 1998.

[24] P.R. D'Argenio and E. Brinksma. A calculus for timed automata (extended abstract). In B. Jonsson and J. Parrow, eds, *Formal Techniques in Real-Time and Fault Tolerant Systems*, LNCS 1135, pages 110-129, 1996.

[25] P.R. D'Argenio, J.-P. Katoen and E. Brinksma. An algebraic approach to the specification of stochastic systems (extended abstract). In D. Gries and W.-P. de Roever, eds, *Programming Concepts and Methods*. Chapman & Hall, pages 126–147, 1998.

[26] P.R. D'Argenio, J.-P. Katoen and E. Brinksma. General purpose discrete-event simulation using ♤. In [88], pages 85–102.

[27] P.R. D'Argenio, J.-P. Katoen and E. Brinksma. A compositional approach to generalised semi-Markov processes. In A. Guia, M. Spathopoulos and R. Smedinga, eds, *Proc. 4th Int. Workshop on Discrete-Event Systems*, pages 391–397. IEE Press, 1998.

[28] P.R. D'Argenio, H. Hermanns and J.-P. Katoen. On generative parallel composition. *Electr. Notes on Th. Comp. Sc.*, **22**, 1999.

[29] P.R. D'Argenio and C. Verhoef. A general conservative extension theorem in process algebras with inequalities. *Th. Comp. Sc.*, **177**(2): 351–380, 1997.

[30] C. Derman. *Finite-State Markovian Decision Processes.* Academic Press, 1970.

[31] D. Ferrari. Considerations on the insularity of performance evaluation. *IEEE Trans. on Soft. Eng.*, **12**(6): 678–683, 1986.

[32] A.J. Field, P.G. Harrison, and K. Kanani. Automatic generation of verifiable cache coherence simulation models from high-level specifications. *Australian Comp. Sc. Comms.*, **20**(3):261–275, 1998.

[33] S. Gilmore and J. Hillston. The PEPA Workbench: A tool to support a process algebra-based approach to performance modelling. In G. Haring and G. Kotsis, eds, *Modelling Techniques and Tools for Computer Performance Evaluation*, LNCS 794, pages 353–368, 1994.

[34] S. Gilmore, J. Hillston, R. Holton, and M. Rettelbach. Specifications in stochastic process algebra for a robot control problem. *Int. J. of Production Res.*, **34**(4):1065–1080, 1996.

[35] R.J. van Glabbeek, S.A. Smolka, and B. Steffen. Reactive, generative, and stratified models of probabilistic processes. *Inf. and Comp.*, **121**: 59–80, 1995.

[36] R.J. van Glabbeek and W.P. Weijland. Branching time and abstraction in bisimulation semantics. *J. ACM*, **43**(3): 555–600, 1996.

[37] J.C. Godskesen and K.G. Larsen. Real-time calculi and expansion theorems. In R.K. Shyamasundar, ed, *Foundations of Software Technology and Theoretical Computer Science*, LNCS 652, pages 302–316, 1992.

[38] N. Götz. *Stochastische Prozeßalgebren–Integration von Funktionalem Entwurf und Leistungsbewertung Verteilter Systeme.* PhD. thesis, U. Erlangen-Nürnberg, 1994.

[39] N. Götz, U. Herzog and M. Rettelbach. Multiprocessor and distributed system design: The integration of functional specification and performance analysis using stochastic process algebras. In L. Donatiello and R. Nelson, eds, *Performance Evaluation of Computer and Communication Systems*, LNCS 729, pages 121–146. Springer-Verlag, 1993.

[40] J.F. Groote and M.A. Reniers. Algebraic process verification. In *Handbook of Process Algebra*, 1999 (to appear).

[41] G. Hachtel, E. Macii, A. Padro and F. Somenzi. Markovian analysis of large finite-state machines. *IEEE Trans. on Comp. Aided Design of Integr. Circ. and Sys.*, **15**(12): 1479–1493, 1996.

[42] H. Hansson. *Time and Probability in Formal Design of Distributed Systems.* Elsevier, 1994.

[43] H. Hansson and B. Jonsson. A calculus for communicating systems with time and probabilities. In *Proc. 11th IEEE Real-Time Systems Symposium*, pages 278–287, 1990.

[44] P.G. Harrison and J. Hillston. Exploiting quasi-reversible structures in Markovian process algebra models. *The Comp. J.*, **38**(7): 510–520, 1995.

[45] P.G. Harrison and B. Strulo. Stochastic process algebra for discrete event simulation. In F. Baccelli, A. Jean-Marie and I. Mitrani, eds, *Quantitative Methods in Parallel Systems*, pages 18–37. Springer, 1995.

[46] C. Harvey. Performance engineering as an integral part of system design. *Br. Telecom Technol. J.*, **4**(3): 142–147, 1986.

[47] H. Hermanns. *Interactive Markov Chains.* PhD. thesis, U. Erlangen-Nürnberg, 1998.

[48] H. Hermanns, U. Herzog, U. Klehmet, V. Mertsiotakis, and M. Siegle. Compositional performance modelling with the TIPP-tool. In R. Puigjaner, N.N. Savio and B. Serra, eds, *Computer Performance Evaluation*, LNCS 1469, pages 51–63, 1998.

[49] H. Hermanns, U. Herzog and V. Mertsiotakis. Stochastic process algebras as a tool for performance and dependability modelling. In *Proc. IEEE Int. Computer Performance and Dependability Symposium*, pages 102–111, 1995.

[50] H. Hermanns, U. Herzog and V. Mertsiotakis. Stochastic process algebras: between LOTOS and Markov chains. *Comp. Netw. and ISDN Syst.*, **30**(9/10): 901–924, 1998.

[51] H. Hermanns and J.-P. Katoen. Automated compositional Markov chain generation for a plain-old telephone system. *Sci. of Comp. Programming*, 1999 (to appear).

[52] H. Hermanns and M. Lohrey. Priority and maximal progress are completely axiomatisable (extended abstract). In D. Sangiorgi and R. de Simone, eds, *Concurrency Theory*, LNCS 1466, pages 237–252, 1998.

[53] H. Hermanns, J. Meyer-Kayser and M. Siegle. Multi terminal binary decision diagrams to represent and analyse continuous time Markov chains. In B. Plateau and W.J. Stewart, eds, *Proc. 3rd Int. Workshop on Numerical Solution of Markov Chains*, LNCS, 1999.

[54] H. Hermanns and M. Rettelbach. Syntax, semantics, equivalences, and axioms for MTIPP. In [61] pages 71–88.

[55] H. Hermanns and M. Rettelbach. Towards a superset of LOTOS for performance prediction. In [89], pages 77–94.

[56] H. Hermanns, M. Rettelbach and T. Weiss. Formal characterisation of immediate actions in SPA with non-deterministic branching. *The Comp. J.*, **38**(7): 530–542, 1995.

[57] H. Hermanns and M. Siegle. Bisimulation algorithms for stochastic process algebras and their BDD-based implementation. In [69], pages 244–265.

[58] U. Herzog. Formal description, time and performance analysis: a framework. In T. Härder, H. Wedekind and G. Zimmermann, eds, *Entwurf und Betrieb Verteilter Systeme*, Informatik Fach-Berichte 264, Springer, 1990.

[59] U. Herzog. A concept for graph-based stochastic process algebras, generally distributed activity times, and hierarchical modelling. In [89], pages 1–20.

[60] U. Herzog and V. Mertsiotakis. Stochastic process algebras applied to failure modelling. In [61], pages 107–126.

[61] U. Herzog and M. Rettelbach, eds. *Proc. 2nd Workshop on Process Algebra and Performance Modelling.* Arbeitsbericht **27**(4), U. Erlangen-Nürnberg, 1994.

[62] J. Hillston. PEPA: Performance Enhanced Process Algebra. Tech. Rep. CSR-24-93, U. Edinburgh, 1993.

[63] J. Hillston. *A Compositional Approach to Performance Modelling.* Cambridge University Press, 1996.

[64] J. Hillston. Exploiting symmetry in solution: decomposing composed models. In [88], pages 1–15.

[65] J. Hillston and V. Mertsiotakis. A simple time scale decomposition technique for stochastic process algebras. *The Comp. J.*, **38**(7): 566–577, 1995.

[66] J. Hillston and N. Thomas. Product-form solutions for a class of PEPA models. *Performance Evaluation*, **35**(3): 171–192, 1999.

[67] J. Hillston and N. Thomas. A syntactic analysis of reversible PEPA processes. In [88], pages 37–50.

[68] K. Kanani. *A Unified Framework for Systematic Quantitative and Qualitative Analysis of Communicating Systems.* Imperial College, Univ. of London, 1998.

[69] J.-P. Katoen, ed. *Formal Methods for Real-Time and Probabilistic Systems.* LNCS 1601, 1999.

[70] J.-P. Katoen, E. Brinksma, D. Latella and R. Langerak. Stochastic simulation of event structures. In [89], pages 21–40.

[71] J.G. Kemeny and J.L. Snell. *Finite Markov Chains.* Van Nostrand, 1960.

[72] U. Klehmet. Extensions of stochastic process algebras for performance and dependability modelling. *Proc. 12th Eur. Simulation Multiconf.*, pages 771–775, 1998.

[73] H. Kobayashi. *Modeling and Analysis: An Introduction to System Performance Evaluation Methodology.* Addison-Wesley, 1978.

[74] K.G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Inf. and Comp.*, **94**(1): 1–28, 1992.

[75] L. Léonard and G. Leduc. An introduction to ET-LOTOS for the description of time-sensitive systems. *Comp. Netw. & ISDN Sys.* **29**(3): 271–292, 1997.

[76] G. Lowe. Probabilistic and prioritized models of timed CSP. *Th. Comp. Sci.*, **138**: 315–352, 1995.

[77] V. Mertsiotakis. *Approximate Analysis Methods for Stochastic Process Algebras.* PhD. thesis, U. Erlangen-Nürnberg, 1998.

[78] V. Mertsiotakis and M. Silva. Throughput approximation of decision-free processes using decomposition. In *Proc. 7th Int. Workshop on Petri Nets and Performance Models*, pages 174–182. IEEE CS-Press, 1997.

[79] C. Miguel, A. Fernández and L. Vidaller. LOTOS extended with probabilistic behaviours. *Form. Asp. of Comp.*, **5**: 253–281, 1993.

[80] F. Moller and C. Tofts. A temporal calculus of communicating systems. In J.C.M. Baeten and J-W. Klop, eds, *Concur'90: Theories of Concurrency – Unification and Extension*, LNCS 458, pages 401–415, 1990.

[81] M.F. Neuts. *Matrix-geometric Solutions in Stochastic Models – An Algorithmic Approach.* The Johns Hopkins University Press, 1981.

[82] X. Nicollin and J. Sifakis. An overview and synthesis on timed process algebras. In J.W. de Bakker et al, eds, *Real-Time: Theory in Practice*, LNCS 600, pages 526–548, 1992.

[83] N.M. Nounou and Y. Yemini. Algebraic specification-based performance analysis of communication protocols. In Y. Yemini, R. Strom and S. Yemini, eds, *Protocol Spec., Testing, and Ver. IV*, pages 541–560. North-Holland, 1985.

[84] N.M. Nounou. *A Methodology for Specification-based Performance Analysis of Protocols.* PhD. thesis, Columbia University, 1986.

[85] B. Plateau and K. Atif. Stochastic automata networks for modelling parallel systems. *IEEE Trans. on Soft. Eng.*, **17**(10): 1093-1108, 1991.

[86] C. Priami. Stochastic π-calculus. *The Comp. J.*, **38**(7): 578–589, 1995.

[87] C. Priami. Stochastic π-calculus with general distributions. In [89], pages 41–57.

[88] C. Priami, ed. *Proc. 6th Int. Workshop on Process Algebra and Performance Modelling.* Tech. Rep., Università di Verona, 1998.

[89] M. Ribaudo, ed. *Proc. 4th Int. Workshop on Process Algebra and Performance Modelling.* C.L.U.T. Press, 1996.

[90] N. Rico and G. von Bochmann. Performance description and analysis for distributed systems using a variant of LOTOS. In B. Jonsson, J. Parrow and B. Pehrson, eds, *Protocol Spec., Testing, and Ver. IX*, pages 199–213. North-Holland, 1991.

[91] R.A. Sahner and K.S. Trivedi. Performance and reliability analysis using directed acyclic graphs. *IEEE Trans. on Softw. Eng.*, **13**(10): 1105–1114, 1987.

[92] S. Schneider. An operational semantics for timed CSP. *Inf. & Comp.*, **116**:193–213, 1995.

[93] J. Schot. Addressing performance requirements in FDT-based design of distributed systems. *Comp. and Comm.*, 1992.

[94] R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic J. of Computing*, **2**(2): 250–273, 1995.

[95] G.S. Shedler. *Regenerative Stochastic Simulation.* Academic Press, Inc., 1993.

[96] M. Sereno. Towards a product-form solution of stochastic process algebras. *Comp. J.*, **38**(7): 622–632.

[97] W.J. Stewart. *Introduction to the Numerical Solution of Markov Chains.* Princeton University Press, 1994.

[98] R. von Stieglitz. Messung und Bewertung des dynamischen Verhaltens eines Kommunikationssystem-Prototypen für Breitband-ISDN (BERKOM). MSc. thesis, U. Erlangen-Nürnberg, 1990.

[99] B. Strulo. *A Process Algebra for Discrete-Event Simulation.* PhD. thesis, Imperial College, U. London, 1993.

[100] C. Tofts. Processes with probabilities, priorities and time. *Form. Asp. of Comp.*, **6**(5): 536–564, 1994.

[101] C. Tofts. Compositional performance analysis. In E. Brinksma, ed, *Tools and Algorithms for the Construction and Analysis of Systems*, LNCS 1217, pages 290–305, 1997.

[102] A. Valderrutten, O. Hjiej, A. Benzekri, and D. Gazal. Deriving queuing networks performance models from annotated LOTOS specifications. In R. Pooley and J. Hillston, eds, *Computer Performance Evaluation – Modelling Techniques and Tools*, pages 167–178, 1992.

[103] M.Y. Vardi. Automatic verification of probabilistic concurrent finite state programs. In *Proc.* 26th *Symp. on Foundations of Computer Science*, pages 327–338, 1985.

[104] Y. Wang. Real-time behaviour of asynchronous agents. In J.C.M. Baeten and J-W. Klop, eds, *Concur'90: Theories of Concurrency – Unification and Extension*, LNCS 458, pages 502–520, 1990.

[105] I. Weigel. *Modelle für Entwurf und Bewertung eines dynamischen Lastverbunds lose gekoppelter Rechensysteme* PhD. thesis, U. Erlangen-Nürnberg, 1994.

[106] S.-H. Wu, S.A. Smolka and E.W. Stark. Composition and behaviors of probabilistic I/O automata. *Th. Comp. Sc.*, **176**(1/2): 1–38, 1997.